Distributed Photometric Bundle Adjustment

Nikolaus Demmel, Maolin Gao, Emanuel Laude, Tao Wu and Daniel Cremers
Technical University of Munich
{nikolaus.demmel, maolin.gao, emanuel.laude, tao.wu, cremers}@tum.de

Abstract

In this paper we demonstrate that global photometric bundle adjustment (PBA) over all past keyframes can significantly improve the global accuracy of a monocular SLAM map compared to geometric techniques such as pose-graph optimization or traditional (geometric) bundle adjustment. However, PBA is computationally expensive in runtime, and memory usage can be prohibitively high. In order to address this scalability issue, we formulate PBA as an approximate consensus program. Due to its decomposable structure, the problem can be solved with block coordinate descent in parallel across multiple independent workers, each having lower requirements on memory and computational resources. For improved accuracy and convergence, we propose a novel gauge aware consensus update. Our experiments on real-world data show an average error reduction of 62% compared to odometry and 33% compared to intermediate pose-graph optimization, and that compared to the central optimization on a single machine, our distributed PBA achieves competitive pose-accuracy and cost.

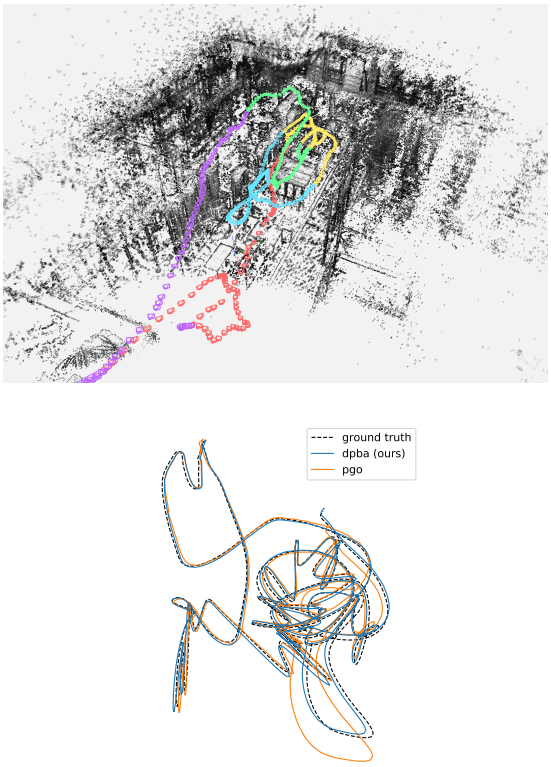


Figure 1. *Top:* Reconstructed sparse point cloud of machine hall (*MH01*) after global PBA. Keyframes are colored according to their partitioning into subproblems. *Bottom:* Estimated camera trajectory before and after global PBA for sequence *V103*.

1. Introduction

Bundle Adjustment (BA) — the joint optimization of camera poses and 3D landmark positions — is a central component of many Visual SLAM and Structure from Motion systems (SfM) [28, 19]. In Photometric Bundle Adjustment (PBA), instead of the more traditional geometric reprojection error based on keypoint matches, we work directly with image pixel values and minimize the intensity differences of reprojected points [7, 2, 8]. Minimizing the photometric error is also known as a *direct* method, as opposed to *indirect*, feature-based approaches with fixed point associations. PBA is paramount for the high accuracy of direct odometry and SLAM systems [8, 35].

While the photometric error is often used with a dense depth representation, this is not inherent for direct methods. In fact, more recent works [2, 8, 35] instead optimize over a *sparse* set of 3D points and with the term *photometric bundle adjustment* we implicitly refer to this sparse formulation. Therein, landmarks are not directly related by residuals, and thus the Schur complement trick — which is crucial for efficiently solving large BA problems — can also be applied. However, PBA is still computationally more expensive both in runtime and memory usage compared to its geometric counterpart: We make use not only of corner features but also other points with sufficient image gradient, like edges, and thus optimize over more landmarks. Each point observation has higher dimension, as it consists of a whole patch of intensity differences instead of just a difference in pixel location. And finally, the optimized cost is

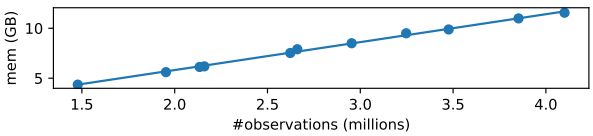


Figure 2. Memory used by our central PBA implementation with growing problem size on the *EuRoC* sequences.

more expensive to evaluate, requiring image interpolation and gradient computation during the optimization. This also necessitates keeping keyframe images in memory. Therefore, PBA in a SLAM context is so far limited to a small set of local keyframes. To correct the accumulated drift over time, direct SLAM systems employ geometric techniques such as keypoint-based loop-closure detection and pose-graph optimization (PGO) [12].

In this paper, we for the first time demonstrate that global photometric bundle adjustment over all past keyframes can significantly improve the global map accuracy compared to PGO alone (see Fig. 1). However, even moderately-sized problems with thousands of keyframes and millions of landmarks can exceed the memory of a typical desktop computer. Fig. 2 shows that the memory usage of our (central) PBA solver grows linearly with the problem size. We hence propose to make the optimization feasible by distributing the computation across multiple parallel workers, each having lower requirements on memory.

In order to achieve this, our key idea is to reformulate the optimization cost in a decomposable way, such that each subproblem is of the same structure as PBA and can be solved independently with standard techniques. We propose a distributed approximate consensus optimization scheme to synchronize between the parallel workers on a master (assuming a star-shaped network topology) and thus (approximately) minimize the same cost as in the original centralized formulation. One important innovation is: unlike previous consensus-optimization schemes for BA [10, 33], we explicitly consider the problem's gauge freedom when combining the individual workers' results on the master.

Our decomposition strategy has several advantages. The workers have to exchange only camera variables with the master, while landmarks are treated as purely local, limiting communication overhead [33]. For each worker, only the subset of the keyframe images in overlapping regions are shared with a small subset of neighboring clusters, and not at all with the master. And lastly, the updates on the master are efficient, and all computationally expensive work is done in parallel on the workers.

The **contributions** of this paper include: (i) We for the first time demonstrate the effectiveness of *global* photometric bundle adjustment in obtaining highly accurate maps of sparse landmarks. (ii) We propose a novel distributed approximate consensus optimization that takes into account the specifics of PBA (every residual relates one landmark with *two* keyframes; additional per-frame optimization variables for brightness transfer, . . .). (iii) Unlike previous work [10, 33], we evaluate not only the optimized cost, but also the pose-accuracy of solutions. (iv) We explicitly consider the gauge freedom of monocular PBA in the distributed formulation and show that our novel gauge aware consensus update leads to faster convergence and more accurate solutions. (v) We release our global PBA pipeline as open source: https://go.vision.in.tum.de/dpba.

2. Related Work

Photometric Bundle Adjustment While feature-based methods have long been dominant for SfM and SLAM, they are ultimately limited by the accuracy of feature detection and matching. In the last decade, direct methods relying on the photometric error have become popular for pose estimation (typically frame-to-frame) [18, 9] and (semi-)dense depth estimation [9, 20, 25]. Those methods show great accuracy and robustness in real-time, but don't attempt to optimize structure and motion jointly.

The joint optimization with photometric residuals has been proposed with a variety of dense structure parameterizations, such as meshes [7], surfels [22] or learnt low-dimensional depth map representations [4]. However, their joint optimization is limited to a small number of frames [7, 4, 27]. Impressive performance is demonstrated in [22], but the formulation relies heavily on depth measurements from an RGB-D camera and to achieve real-time, they *alternate* between the refinement of poses and structure.

More recently, the idea of *sparse* photometric bundle adjustment has emerged [2, 8, 14]. Representing the map as a sparse set of landmarks allows use of the Schur complement trick, unlocking efficient joint optimization at large scale [28]. These odometry systems have demonstrated real-time performance in a sliding window of 5-7 keyframes and thousands of points. Extensions to SLAM are either also limited to joint refinement in a small window of covisible keyframes [35] or use features-based loop-closure estimation and subsequent pose-graph optimization (PGO), *i.e.* globally optimize only for poses [12]. There is no attempt at global refinement of all keyframes and landmarks, as is common in feature-based SLAM systems [19], presumably due to the substantial computational cost.

While [2] parameterizes landmarks as 3D positions in map coordinates, we base our model on [8], which represents points by the inverse depth w.r.t. a fixed pixel position in a host frame. This allows residuals to take into account rotation and scale changes between host and target frames — important for larger baselines — but it also means, that unlike in traditional BA, every residual now relates *two* frames. We later show that this is important to consider when clustering the map into subproblems.

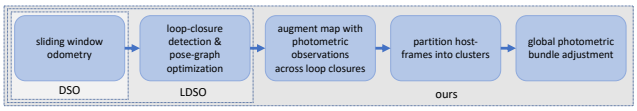


Figure 3. Our pipeline to initialize the global PBA problem for refinement of the map created by LDSO [12].

Distributed Optimization The need for parallel computation for solving BA problems has long been recognized [21, 31]. Ni *et al*. [21] highlight the computational feasibility with limited memory on a single machine and propose an out-of-core solver that alternates between optimizing independent clusters and the so called separator variables in overlapping regions. Unlike our consensus-based approach, the optimization over these boundary regions has to be done jointly and it inevitably grows with growing problem size. Wu *et al*. [31] avoid explicit instantiation of the Schur complement matrix by using a specific iterative solver for the normal equations: *preconditioned conjugate gradient* (PCG). Their implementation computes each PCG step in parallel, and thus has to synchronize in every PCG iteration. Our consensus optimization parallelizes on a higher level, *i.e*. the formulation of BA itself, and is somewhat agnostic to the subproblem solver.

A numerical method that has received considerable attention in the field of distributed optimization is the consensus *alternating direction method of multipliers* (ADMM) [15] originally due to [13, 11]. More recently, in computer vision, the method has been applied to large scale feature-based BA [10, 33]. Eriksson *et al*. [10] propose a distributed BA model with point consensus. To avoid large communication overhead, [33] instead propose a camera consensus formulation which is optimized by ADMM. We follow their conclusion and also impose consensus only on camera variables. However, we propose an algorithmic variation that leaves out the Lagrange multipliers (the dual variables) in ADMM. Due to the Lagrange multiplier updates, ADMM generally suffers from a lack of convergence in the non-convex setting, unless the cost function is strongly smooth [15]. We found this to be the case in our experiments with ADMM and infer that it is due to lack of smoothness of the photometric error, inherited from the spatial non-smoothness of image intensities.

Distributed optimization has recently also been proposed for the related problems of PGO [6] and motion averaging [34]. Optimizing over camera poses only, they show impressive results on very large scale problems (by skipping the joint structure and motion refinement), but sacrifice some of the potential accuracy.

3. Pipeline Overview

Global PBA can be useful in a variety of applications. While it is feasible to perform on top of feature-based initializations (SLAM or SfM), it would require careful selection and initialization of additional non-corner points to exploit one of the key advantages of PBA over BA. Doing this well is a whole separate research topic. To keep the setup simple and focus on the optimization, we therefore choose the direct SLAM system LDSO [12] and evaluate the gained global map-accuracy after offline post-processing its output with PBA. As PBA is a non-convex problem where successful optimization highly depends on accurate initialization, we rely on LDSO to select a meaningful set of keyframes, landmarks and observations, and correct accumulated drift after detecting loop closures. In the following we briefly review LDSO and discuss how on top of that we augment the map with additional observations of existing landmarks where loop closures are detected and how we partition the resulting observations graph into the desired number of clusters using on-graph k-means (see Fig. 3). We then devote Section 4 to our distributed PBA formulation — the main focus of this paper.

3.1. Odometry and Pose-Graph Optimization

The monocular odometry [8] localizes sequential images by maintaining a sliding window of up to 7 keyframes. For each keyframe, up to 2000 sparse points are selected in high-gradient regions of the image, and their depth is initialized with epipolar-line search in following frames. Using well-initialized points, a set of active landmarks is maintained in the sliding window. In order to cope with unknown exposure times, missing photometric calibration or changes in lighting conditions, each keyframe maintains affine brightness transfer parameters. Keyframe poses, brightness transfer parameters, as well as landmark positions are refined by local photometric bundle adjustment. We refer to the keyframe-poses at this stage as *odom* in our evaluation (Section 5).

Since the odometry inevitably accumulates drift, global mapping [12] attempts to correct this. Candidate loop-closures are proposed with visual bag-of-words using ORB-descriptors, and after geometric verification, relative pose constraints are inserted into a pose-graph as loop-closure edges. A subsequent $\mathrm{Sim}(3)$ optimization corrects all keyframe poses. It should be noted that while we optimize over frames only, the corrected scale of each keyframe is used to rescale its hosted landmarks accordingly. We use this optimized map to add photometric observations in the following stage and we initialize our global PBA with the corrected state, which we refer to as *pgo* in the evaluation. For further details, please refer to the respective works.

3.2. Augmenting the Map

In order to benefit from global PBA, we need to associate temporally distant keyframes when a loop closure has occured. For this, we rely on the undirected *observation graph*

of the initial map, where cameras are connected if they share observations. For each keyframe we consider the set of all keyframes connected via loop-closure edges, and their direct neighbors in the observation graph, which gives a set of new candidate edges. For each candidate edge, we reproject all landmarks from host to target frame (both ways), and include a new observation, if the reprojection falls within the image bounds of the target frame and the relative scale (ratio of distance in host and target frame) is sufficiently close to 1. Note that more elaborate schemes are possible: detecting outliers, occlusions and duplicate points. However, robust norms make our optimization somewhat robust to outliers and the evaluation shows that even this simple scheme results in great accuracy. This is in contrast to traditional BA, where great attention has to be given to filtering outlier matches to get accurate results.

3.3. Clustering into Subproblems

Different approaches have been proposed to partition a BA graph into subproblems, *e.g.* (normalized) graph cuts [34, 21, 33]. As we are proposing camera consensus [33], we need to partition the set of landmarks and assign them to unique clusters. Here we want to highlight an important difference to traditional BA: In our PBA formulation every residual involves *two* frames, hence all landmarks sharing the same host frame, as well as all residuals involving these landmarks, should be assigned to the same cluster. Thus, we start by partitioning the set of host frames, which then induces an assignment of landmarks and observations and in turn the required camera copies by the target-frame relation.

This keyframe partitioning should satisfy several criteria. It should be balanced, *i.e.* comprise clusters of similar size. Communication overhead during optimization should be low, *i.e.* we desire a low number of camera copies. Lastly, the clusters should be connected and well constrained. In our experiments we found on-graph k-means clustering on the observation graph with pairwise Euclidean distances of initial camera positions as edge weights to be effective in balancing these conflicting goals. Intuitively, relying on the initial camera positions as well as the connectivity of the observation graph results in spatially compact, connected clusters (see Fig. 1). For further details on how we implement this as an integer linear program to enforce hard-constraints on cluster balance [3], please refer to the supplementary material. Note that to ensure all camera copies are well-constrained, we do in fact also duplicate some landmarks and observations (see Section 4.1), but we neither enforce consensus on any duplicate landmarks, nor do they require additional camera copies.

3.4. Global Photometric Bundle Adjustment

We refine the final map from LDSO by optimizing F_{full} over the states Θ_i of all keyframes $i \in F$ (comprising world-to-camera SE(3) transformations $T_i \simeq (\mathbf{R}_i, \mathbf{t}_i)$ and brightness transfer parameters $a_i,\ b_i$) and inverse distances $d_{\mathbf{q}}$ of all landmarks $\mathbf{q}$:

$$F_{\text{full}} := \sum_{i\in\mathcal{F}} \sum_{\mathbf{q}\in\mathcal{P}_i} \sum_{j\in\mathcal{Q}_\mathbf{q}} F_{i,\mathbf{q},j}(\Theta_i, d_\mathbf{q}, \Theta_j), \tag{1}$$

$$F_{i,\mathbf{q},j}(\cdot) := \sum_{\mathbf{p}\in\mathcal{N}_\mathbf{q}} \left\| (I_j(\mathbf{p}') - b_j) - e^{a_j - a_i}(I_i(\mathbf{p}) - b_i) \right\|_\gamma, \tag{2}$$

$$\mathbf{p}' := \Pi(\mathbf{R}_{j,i}\,\Pi^{-1}(\mathbf{p}) + d_\mathbf{q}\mathbf{t}_{j,i}). \tag{3}$$

We sum over all host frames $i \in \mathcal{F}$ and the hosted points $\mathcal{P}_i$ therein. Each point $\mathbf{q} \in \mathcal{P}_i$ is observed by target frames $\mathcal{Q}_\mathbf{q}$ and in each $j \in \mathcal{Q}_\mathbf{q}$ we compute the brightness-adjusted photometric error $F_{i,\mathbf{q},j}$. Specifically, $F_{i,\mathbf{q},j}$ is evaluated using a sum of Huber norms $\|\cdot\|_\gamma$ over the 8-pixel neighborhood pattern $\mathcal{N}_\mathbf{q}$ proposed in [8]. Each reprojected pixel $\mathbf{p}'$ depends on the relative keyframe pose $(\mathbf{R}_{j,i}, \mathbf{t}_{j,i}) :\simeq \mathbf{T}_j\mathbf{T}_i^{-1}$, and the target frame image intensity $I_j(\mathbf{p}')$ is computed by bilinear interpolation. In addition, Π and Π^{-1} are camera projection and back-projection (to unit-vector) with fixed intrinsic parameters.

In the centralized case, we minimize (1) with Levenberg-Marquardt (LM) — as is common for global BA [28, 2, 8] — and refer to the result as *pba* in the evaluation. Similar to the related work on consensus optimization for BA [10, 33], we employ the same solver for the local subproblems in our distributed optimization (see Section 4.2). We build our implementation on the popular Ceres Solver [1].

4. Distributed Global PBA

4.1. Model Formulation

We propose a *distributed* formulation of the global photometric bundle adjustment (1). As discussed in Section 3.3, we start by partitioning the host frames $\mathcal{F}$ into L subsets, *i.e.*, $\mathcal{F} = \cup_{l=1}^{L}\mathcal{F}^l$. Based on this partition, we split the sum of residuals in (1) into the L clusters, which become independent with the introduction of cluster-wise *local* duplicates to the *global* variables.

In order to assign residuals to clusters and determine which variables need to be duplicated, we define the *scope* of cluster l, denoted by $\mathcal{C}^l$, as the union of all host frames and their associated targets frames:

$$\mathcal{C}^l := \mathcal{F}^l \cup \left(\cup_{i\in\mathcal{F}^l} \cup_{\mathbf{q}\in\mathcal{P}_i} \mathcal{Q}_\mathbf{q} \right). \tag{4}$$

Whenever $i \in \mathcal{C}^l$, we introduce θ_i^l as the local duplicate of the frame variable Θ_i in cluster l. We also define for each point $\mathbf{q}$ the set of target frames limited to cluster l as $\mathcal{Q}_\mathbf{q}^l$ and for each frame $i \in \mathcal{C}^l$ we limit the set of hosted points $\mathcal{P}_i^l$ to ensure it has at least one observation in cluster l:

$$\mathcal{Q}_\mathbf{q}^l := \mathcal{Q}_\mathbf{q} \cap \mathcal{C}^l, \qquad \mathcal{P}_i^l := \{\mathbf{q} \in \mathcal{P}_i : \mathcal{Q}_\mathbf{q}^l \neq \emptyset\}. \tag{5}$$

With that, a residual $F_{i,\mathbf{q},j}$ occurs in cluster l if $i \in \mathcal{C}^l$, $\mathbf{q} \in \mathcal{P}_i^l$, $j \in \mathcal{Q}_\mathbf{q}^l$. Notice that we also include points hosted in frames $i \in \mathcal{C}^l \setminus \mathcal{F}^l$ that are not in the initial partition $\mathcal{F}^l$, as long as they have at least one observation in cluster l. We do this, because otherwise we observed some local *pure target* frames $i \in \mathcal{C}^l \setminus \mathcal{F}^l$ to be underconstrained.

Since the same residual $F_{i,\mathbf{q},j}$ may appear in multiple clusters, we normalize it with the number of appearances:

$$n_{i,\mathbf{q},j} := \left|\left\{l : i \in \mathcal{C}^l,\ \mathbf{q} \in \mathcal{P}_i^l,\ j \in \mathcal{Q}_\mathbf{q}^l\right\}\right|. \tag{6}$$

While also some points may occur in multiple clusters, we choose to not enforce any consensus and keep $\{d_\mathbf{q}^l\}$ as internal variables for cluster l. This limits communication overhead [33]. For comparison to the central solver we need to evaluate the cost (1) with global variables, for which we pick $d_\mathbf{q}^l$ from the cluster where the point's host frame was initially assigned to by partition $\mathcal{F}^l$.

We are now ready to formulate the distributed PBA as an *approximate consensus program*:

$$\begin{aligned} F_{\text{split}} := &\sum_{l=1}^{L} \sum_{i \in \mathcal{C}^l} \sum_{\mathbf{q} \in \mathcal{P}_i^l} \sum_{j \in \mathcal{Q}_\mathbf{q}^l} \frac{1}{n_{i,\mathbf{q},j}} F_{i,\mathbf{q},j}(\theta_i^l, d_\mathbf{q}^l, \theta_j^l) \\ &+ \sum_{l=1}^{L} \sum_{i \in \mathcal{C}^l} P_i^l(\Theta_i, \theta_i^l). \end{aligned} \tag{7}$$

Notice how the summands over clusters l are independent w.r.t. local variables. To relate local and global frame variables, we add quadratic penalties:

$$\begin{aligned} P_i^l(\Theta_i, \theta_i^l) := \Big(&\frac{\rho_\mathbf{R}}{2} \left\|\mathbf{R}_i - \mathbf{R}_i^l\right\|_F^2 + \frac{\rho_\mathbf{t}}{2} \left\|\mathbf{t}_i - \mathbf{t}_i^l\right\|_2^2 \\ &+ \frac{\rho_a}{2} |a_i - a_i^l|^2 + \frac{\rho_b}{2} |b_i - b_i^l|^2 \Big). \end{aligned} \tag{8}$$

Here $\rho_\mathbf{R}, \rho_\mathbf{t}, \rho_a, \rho_b$ are the weights of respective consensus penalties on the discrepancy between global variables and local duplicates. We start with low weights and increase them during the optimization to enforce consensus. In particular, we choose to penalize the difference between $\mathbf{R}_i, \mathbf{R}_i^l \in \mathrm{SO}(3)$ with respect to the Frobenius norm $\|\cdot\|_F$ in the embedding space $\mathbb{R}^{3\times3}$. See Fig. 4 for an illustration of the approximate consensus program.

Our design follows the principle of camera consensus similar to [33] — only the frame variables are duplicated and enforced with consensus constraints. The main differences to the previous work are two-fold:

1. The factor graph is structurally different. Unlike traditional bundle adjustment [28], the PBA here parameterizes its residual using the relative camera pose, hence involving two frame variables for each residual. As a result, the distributed formulation inevitably involves duplicates of both frame variables (poses and brightness transfer parameters) and point variables (inverse distances), unless some local *pure target* frames be left poorly constrained.

2. The distributed BA in [10, 33] aims for perfect consensus between global variables and their duplicates. In our formulation, the consensus between global and local frame variables is enforced by quadratic penalties, which can be optimized by purely primal alternating minimization (see Section 4.2). This avoids numerically unstable dual updates (see the related work in Section 2) which we found to be particularly challenging for the non-smooth photometric residuals.

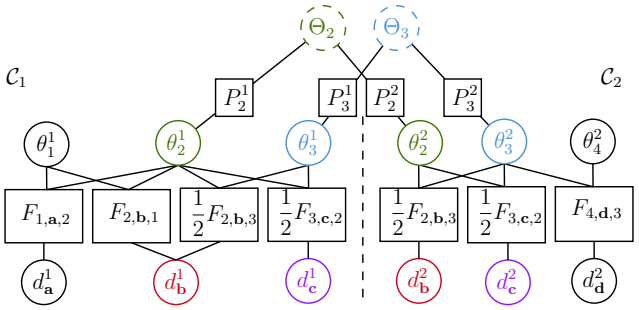


Figure 4. Factor graph of the distributed photometric bundle adjustment (7). The host frames are initially partitioned into two clusters $\mathcal{F}_1 = \{1, 2\}$, $\mathcal{F}_2 = \{3, 4\}$, inducing local duplicates of variables and photometric residuals F in $\mathcal{C}_1, \mathcal{C}_2$. The consensus is imposed on the frame variables θ_i^l with quadratic penalties P, but not on the point variables $d_\mathbf{q}^l$. Local duplicates (solid) of the same global variable (dashed) are shown in equal colors. Duplicated residuals F have weights that sum up to 1. It can be seen that optimizing over local variables with fixed globals can be done independently for each cluster and optimizing global variables with fixed locals only involves frame variables.

These two differences make our consensus program *approximate*. Despite this, we show in Section 5 that the distributed optimization approaches the performance of the central solver both in cost and accuracy.

4.2. Solution Algorithm

Our solution algorithm for the distributed global PBA (7) deploys an alternating minimization strategy switching between local update and (global) consensus update, which we elaborate in the following. At the end of each outer iteration, we increase the penalty weights with rate $\beta > 1$ in order to tighten the consensus.

Local Update as Bundle Adjustment The local update takes place in parallel over the L clusters. Within each cluster, the minimization of the distributed model (7) over local

variables $\{\theta_i^l : i \in \mathcal{C}^l\} \cup \{d_{\mathbf{q}}^l : \mathbf{q} \in \cup_{i\in\mathcal{C}^l} \mathcal{P}_i^l\}$ reduces to:

$$\sum_{i\in\mathcal{C}^l} \sum_{\mathbf{q}\in\mathcal{P}_i^l} \sum_{j\in\mathcal{Q}_{\mathbf{q}}^l} \frac{1}{n_{i,\mathbf{q},j}} F_{i,\mathbf{q},j}(\theta_i^l, d_{\mathbf{q}}^l, \theta_j^l) + \sum_{i\in\mathcal{C}^l} P_i^l(\Theta_i, \theta_i^l). \tag{9}$$

Since the quadratic consensus penalties fit the nonlinear least squares framework, this subproblem has in essence the same structure as our central PBA and we implement it with the same solver (see Section 3.4).

Naive Consensus Update We turn to the minimization of model (7) over the global variables $\{\Theta_i : i \in \mathcal{F}\}$ (with all local variables fixed), which we refer to as the global or consensus update. Naively, it reduces to minimizing:

$$\sum_{l=1}^{L} \sum_{i\in\mathcal{C}^l} P_i^l(\Theta_i, \theta_i^l). \tag{10}$$

A simple closed form solution exists as arithmetic means for each $\mathbf{t}_i, a_i, b_i$, and the Fréchet mean under $\|\cdot\|_F$ for each $\mathbf{R}_i \in \mathrm{SO}(3)$. The consensus algorithm that alternates the local update (9) with the naive consensus update (10) is referred to as *nogauge* in the evaluation.

Gauge Freedom The naive consensus update (10) causes slow convergence of the overall solution algorithm or, worse, convergence towards some undesirable local minimizer (see Fig. 5). The reason for this pitfall is the presence of *gauge freedom* in any bundle adjustment problem [28, 16]. In essence, any solution of a bundle adjust problem is *invariant* (in terms of the objective function) under a *gauge transform*. In monocular BA this is typically a similarity transformation with 7 degrees of freedom. In our case, we have one additional degree of freedom corresponding to the brightness transfer variables a_i. All invariant solutions form the so-called *gauge orbit* [28].

In the distributed PBA model (7), each residual $F_{i,\mathbf{q},j}(\theta_i^l, d_{\mathbf{q}}^l, \theta_j^l)$ is invariant under the gauge transform:

$$\mathbf{R}_i^l \leftarrow \mathbf{R}_i^l \widetilde{\mathbf{R}}^l, \quad \mathbf{t}_i^l \leftarrow \mathbf{R}_i^l \widetilde{\mathbf{t}}^l + \widetilde{s}^l \mathbf{t}_i^l, \tag{11}$$

$$d_{\mathbf{q}}^l \leftarrow d_{\mathbf{q}}^l / \widetilde{s}^l, \quad a_i^l \leftarrow a_i^l + \widetilde{a}^l, \tag{12}$$

parameterized by the (cluster-wise) gauge variables $\xi^l = (\widetilde{\mathbf{R}}^l, \widetilde{\mathbf{t}}^l, \widetilde{s}^l, \widetilde{a}^l) \in \mathrm{SO}(3) \times \mathbb{R}^3 \times \mathbb{R}_{>0} \times \mathbb{R}$. Given a gauge transform ξ^l, we denote the transformed local variables by $\widetilde{\theta}_i^l(\theta_i^l, \xi^l)$ and $\widetilde{d}_{\mathbf{q}}^l(d_{\mathbf{q}}^l, \xi^l)$.

Consensus Update assisted by Gauge Alignment If we replace the local variables $\{\theta_i^l\}, \{d_{\mathbf{q}}^l\}$ in the distributed photometric bundle adjustment model (7) with $\{\widetilde{\theta}_i^l(\theta_i^l, \xi^l)\}, \{\widetilde{d}_{\mathbf{q}}^l(d_{\mathbf{q}}^l, \xi^l)\}$, we realize that optimizing only over the gauge variables $\{\xi^l\}$ can be interpreted as a *subspace* local update: We allow changing the local variables only along the gauge orbit (w.r.t. the photometric residuals). Then, by construction, the photometric residuals remain constant and the subspace local update is equivalent to minimising (13) over gauge variables $\{\xi^l : l \in \{1, \dots, L\}\}$ with fixed globals $\{\Theta_i\}$:

$$\sum_{l=1}^{L} \sum_{i\in\mathcal{C}^l} P_i^l(\Theta_i, \widetilde{\theta}_i^l(\theta_i^l, \xi^l)). \tag{13}$$

This optimization involves the same shared frame variables as the naive consensus update (10) and can thus be computed efficiently on the master with a simple closed-form solution without additional communication.

Instead of a single naive consensus update, we initialize $(\widetilde{\mathbf{R}}^l, \widetilde{\mathbf{t}}^l, \widetilde{s}^l, \widetilde{a}^l) = (I_3, 0, 1, 0)$ for each l, followed by an inner loop of alternating minimization of the consensus cost (13) over gauge and global variables. The minimization over the global variables $\{\Theta_i\}$ is structurally the same as (10). Upon completion of the inner loop, with the (near-)optimal gauge variables $\{\xi^l\}$, we apply the gauge transforms on the local variables using (11)–(12) and reset $\{\xi^l\}$ for the next inner loop. The next outer iteration with the (full) local update follows.

As a remark, the gauge in the local update (9) of each cluster is fixed by the penalty terms. However, each cluster may be locked onto a slightly different point along its gauge orbit. So an alternative view on the inner loop of alternating minimization is that we attempt to find the joint minimum of (13) w.r.t. to gauge and global variables to align the gauges of all clusters.

Summary and Convergence Analysis In summary, one outer iteration of our solution (referred to as *dpba* in the evaluation) consists of the following steps:

1. perform local update (9) in parallel
2. gather *shared* local variables on master
3. perform gauge aware consensus update (13)
4. broadcast global and gauge variables to workers
5. apply gauge transform to *all* local variables
6. update consensus weights

Please refer to the supplementary material for pseudo code of the entire algorithm as well as the closed form solutions we use in the gauge aware consensus update.

Note that unlike ADMM our algorithm is a specialization of block coordinate descent. In each algorithmic step we monotonically decrease the bounded energy (7). Under mild assumptions on sufficient descent and sub-problem accuracy, the alternating minimization of (7) converges to a stationary point. For a proof analogous to [19], please refer to the supplementary.

5. Evaluation

Setup In order to evaluate the performance of global PBA we test our implementation on the *EuRoC MAV* dataset [5]. These sequences constitute images from a flying drone in an indoor setting and contain accurate ground truth poses. Since we use LDSO as initialization we choose the same subset of frames as [12] for each sequence[1] and run our monocular pipeline on images from the left camera.

We evaluate the result from odometry before (*odom*) and after (*pgo*) pose-graph optimization and run global photometric bundle adjustment with our central (*pba*) and distributed (*dpba*) implementation. Unlike LDSO, we do not undistort images in the global optimization. Instead, we use the Kannala-Brandt camera model [17] for radial distortion. Intrinsics are pre-calibrated using the *basalt* calibration tools [29, 30].

We give a detailed account of our choices for hyperparameters in the supplementary material, but unless indicated otherwise, they are all kept constant across all experiments. By default, we run our distributed *dpba* for 250 outer iterations, with initial penalty weights $\rho_{\mathbf{R}} = \rho_{\mathbf{t}} = \rho_a = 10^2$ and $\rho_b = 1$, while increasing the ρ in every iteration by a factor of $\beta = 1.06$. For our central *pba* we let LM converge.

Accuracy We employ the commonly used Absolute Trajectory Error (ATE) in meters [26], which measures the root-mean-squared camera position error, after $\mathrm{Sim}(3)$-alignment of the estimated keyframe trajectory to the ground truth:

$$\mathrm{ATE}(\{\mathbf{t}_i, \mathbf{R}_i\}) = \min_{\mathbf{R},\mathbf{t},s} \sum_i \left\| \widehat{\mathbf{x}}_i + s\mathbf{R}(\mathbf{R}_i^\top \mathbf{t}_i) - \mathbf{t} \right\|^2, \tag{14}$$

where $(\mathbf{R}, \mathbf{t}, s) \in \mathrm{SO}(3) \times \mathbb{R}^3 \times \mathbb{R}_{>0}$ is the alignment-transform, $\widehat{\mathbf{x}}_i$ are ground truth camera positions and $-\mathbf{R}_i^\top \mathbf{t}_i$ are the corresponding estimates from global variables. Ground truth poses are interpolated at image time stamps.

Table 1 shows the ATE for all sequences. While PGO usually reduces the odometry drift noticeably, subsequent global PBA can significantly improve the accuracy even further. As a baseline, we also compare to running traditional (geometric) BA with the same PGO initialization, after feature matching and outlier removal (*gba*). Note that for *MH04* and *V203* the odometry has large scale/rotation drift and no verified loop closures were detected.[2] In this failure case of the initialization, global PBA does not help. Please refer to the supplementary material for further illustrations of these failure cases. Despite this, for central and distributed implementation, the average error reduction is, respectively, 64% and 62% compared to *odom* and 36% and 33% compared to *pgo*, as can be seen from the first two columns in Table 2. Qualitative results for optimized landmarks and camera poses are shown in Fig. 1.

[1] Initial rapid rotations (intended for IMU initialization) are skipped. Effectively, the part where the MAV is in the air is used.

[2] Our results for *pgo* are for the most part in line with the resulte reported in [12], with the exception of *MH04*, where [12] reports a low error, but we consistently see large rotation drift.

	[8]	[12]	ours		
	odom	pgo	gba	pba	dpba
MH01	0.059	0.034	0.019	**0.013**	*0.014*
MH02	0.037	*0.019*	*0.019*	**0.014**	**0.014**
MH03	0.237	0.087	**0.039**	**0.039**	*0.043*
MH04	*3.111*	3.122	**3.080**	3.208	3.223
MH05	0.101	*0.074*	0.075	**0.058**	**0.058**
V101	0.116	0.040	0.035	**0.029**	*0.030*
V102	0.256	0.058	0.043	*0.017*	**0.016**
V103	*0.134*	*0.134*	0.162	**0.085**	**0.085**
V201	0.052	0.026	0.019	**0.013**	*0.016*
V202	0.088	*0.053*	**0.041**	0.059	0.063
V203	1.318	1.318	1.319	*1.312*	**1.279**

Table 1. Accuracy for all *EuRoC* sequences. The final ATE in meters for odometry (*odom*), pose-graph optimization (*pgo*), global PBA in central (*pba*) and distributed (*dpba*) implementation with 5 clusters shows that global PBA significantly improves over PGO — more so than traditional bundle adjustment (*gba*). The performance of central and distributed implementation is competitive.

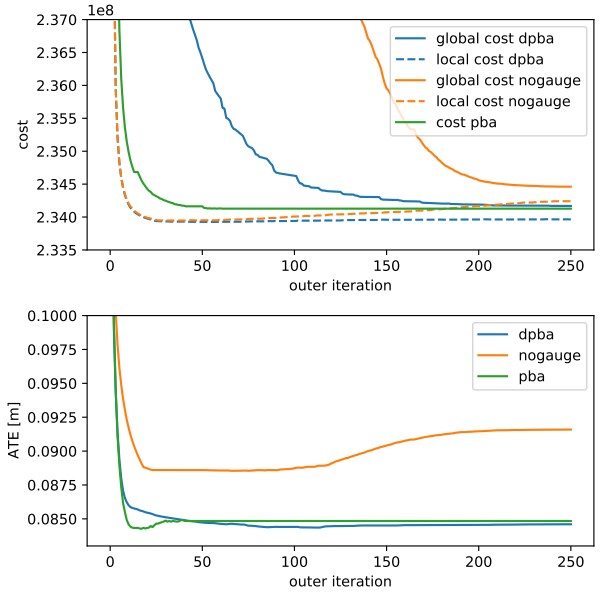


Figure 5. Convergence analysis for sequence *V103*. *Top:* With increasing penalty weight ρ in our distributed *dpba*, global and local cost of the photometric residuals approach the final cost achieved by our central *pba* from above and below, as expected. *Bottom:* Unlike the global cost, the pose accuracy of the central solver is already matched after around 50 iterations. *Both:* Our gauge aware consensus update leads to faster convergence and lower final cost and ATE compared to the naive consensus update (*nogauge*).

Convergence To analyse the convergence behaviour of the distributed algorithm (*dpba*), we compare the final cost of the photometric residuals F_{full} (1), evaluated with local

	pba	dpba			nogauge
ρ-increase rate β		1.06	1.1	1.2	1.06
# iterations (avg)	148	250	150	75	250
fraction of *odom*-ATE	0.363	0.376	0.379	0.380	0.386
fraction of *pgo*-ATE	0.644	0.667	0.673	0.673	0.685
cost global	1.0000	1.0022	1.0038	1.0076	1.0085
cost local	1.0000	0.9987	1.0003	1.0040	1.0040

Table 2. ATE reduction as a fraction of initial (*odom* and *pgo*) error, and final cost as a fraction of the final cost the central solver (*pba*, ours) achieved. We show geometric averages over all *EuRoC* sequences. All distributed variations (*dpba*, ours) are for 5 clusters and share the same initial and final penalty weights ρ. Increasingly higher ρ-increase rates β finish with fewer outer iterations. While the final accuracy is similar, higher β leads to slightly worse solutions in terms of cost. The *cost gap* of around 0.36% remains the same. Thanks to our novel gauge aware consensus update, even the more aggressive optimization of *dpba* over 75 iterations with $\beta = 1.2$ still results in better results than *nogauge* over 250 iterations with $\beta = 1.06$, that takes more than $3\times$ as long in runtime.

and global variables, to the cost achieved by our central solver (*pba*). For different residuals, the local cost may be evaluated with different values for the same frame variables, and thus the cost is expected to be lower than the global cost, while the global cost corresponds to the final result of our optimization, on which ATE is evaluated, so we expect it to be low for good global solutions. Fig. 5 shows the cost evolution for one example sequence, and Table 2 shows the geometric mean of the ratio to the central solver's final cost. The final *cost-gap* is approximately 0.35%. Note that we don't expect the gap to vanish completely, since our consensus optimization is approximate.

We can achieve convergence to the same cost-gap faster with higher ρ-increase rates, but only by sacrificing a little accuracy, as the three middle columns of Table 2 show. Lastly, if we skip our proposed gauge aware consensus update, the convergence can be significantly slower and result in worse solutions, see *nogauge* in Fig. 5 and Table 2.

Number of Clusters, Runtime and Memory Usage The results in Table 3 show that our distributed PBA is robust to different number of clusters. Since the map sizes remain constant, larger number of clusters here means smaller clusters, but also more overlap and more copies per camera. It is therefore natural to expect a slight increase in final error and cost as the number of clusters increases.

Our current implementation runs on a single machine and processes clusters sequentially, so we take the max-time over all clusters for the local update. With 5 clusters we get an average single-threaded runtime (normalized by number of keyframes!) of 43ms per LM-iteration of central *pba*, and 55ms per outer-iteration of *dpba*, on an Intel Xeon W-2133 @ 3.60GHz. Remember that every outer iteration of the distributed PBA involves multiple LM iterations in the local update. In our view, communication bandwidth is

	pba	dpba				
# clusters		3	5	10	15	30
fraction of *odom*-ATE	0.363	0.366	0.376	0.396	0.397	0.401
fraction of *pgo*-ATE	0.644	0.649	0.667	0.702	0.704	0.712
cost global	1.0000	1.0018	1.0022	1.0044	1.0056	1.0067
cost local	1.0000	0.9994	0.9987	0.9991	0.9982	0.9970
time/outer-iteration	1.00	1.60	1.22	0.87	0.77	0.66
memory/worker [GB]	8.0	3.5	2.3	1.4	1.2	0.9

Table 3. Average ATE reduction over all *EuRoC* sequences for our distributed *dpba* with different number of clusters, compared to our central *pba*. As we increase the number of clusters for fixed-size maps, we increase the overlap between clusters and number of copies for each camera. The improvement over the initialization (*pgo*) is slightly reduced, but still substantial. Costs / runtime are averaged fractions of the central solver's final cost / runtime, and accuracy is shown as averaged fractions of initial ATE. Memory usage is averaged over sequences and clusters.

not a major concern, as we follow [33] with consensus only for frame variables. The local update dominates the *dpba* runtime with more than 99%, i.e. the runtime of clustering and consensus/gauge update is largely insignificant. Table 3 shows estimated memory usage based on the number of observations in each cluster and the linear relation from Fig. 2.

6. Conclusion

This paper proposes a novel formulation for decomposable bundle adjustment. In contrast to prior approaches we consider a direct rather than a feature based method which introduces additional challenges: Due to the nonsmoothness of the photometric cost, we resort to a more stable primal alternating minimization scheme rather than a primal-dual iteration as in ADMM. We derive a novel gauge aware consensus update to significantly improve the performance of the overall method and it would be interesting to transfer this to other approaches [10, 33]. Compared to the central optimization on a single machine, the distributed formulation of PBA achieves competitive accuracy and optimized cost and significantly increases the global map accuracy over [12] and over geometric BA. At this stage, the overall runtime is still relatively high, due to a high number of outer iterations required for accurate results and the need to synchronize all workers after every outer iteration (*i.e.* we have to wait for the slowest). The former could be addressed in the future by adaptive per-variable penalty weight schedules (similar to [33]) and the latter by exploring *decentralized* consensus optimization [32, 24, 23]. However, one significant advantage of our distributed algorithm already is, that it can run on a set of workers with limited memory, where the central solver would not fit in any single one.

Acknowledgment This work was supported through the DFG projects WU 959/1-1 and CR 250/20-1 "Splitting Methods for 3D Reconstruction and SLAM".

References

[1] S. Agarwal and K. Mierle. Ceres solver: Tutorial & Reference. *Google Inc.*, 2012. 4, 13

[2] H. Alismail, B. Browning, and S. Lucey. Photometric Bundle Adjustment for vision-based SLAM. In *Asian Conference on Computer Vision (ACCV)*, pages 324–341. Springer, 2016. 1, 2, 4

[3] K. Bennett, P. Bradley, and A. Demiriz. Constrained K-Means clustering. Technical Report MSR-TR-2000-65, Microsoft Research, 2000. 4, 11, 12

[4] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison. Codeslam — learning a compact, optimisable representation for dense visual slam. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2560–2568, 2018. 2

[5] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart. The EuRoC micro aerial vehicle datasets. *The International Journal of Robotics Research*, 2016. 7

[6] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, H. I. Christensen, and F. Dellaert. Distributed mapping with privacy and communication constraints: Lightweight algorithms and object-based models. *The International Journal of Robotics Research*, 36(12):1286–1311, 2017. 3

[7] A. Delaunoy and M. Pollefeys. Photometric Bundle Adjustment for dense multi-view 3d modeling. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1486–1493, 2014. 1, 2

[8] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 40:611–625, 2017. 1, 2, 3, 4, 7, 15

[9] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *European Conference on Computer Vision (ECCV)*, pages 834–849. Springer, 2014. 2

[10] A. Eriksson, J. Bastian, T.-J. Chin, and M. Isaksson. A consensus-based framework for distributed Bundle Adjustment. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1754–1762. IEEE, 2016. 2, 3, 4, 5, 8

[11] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1):17–40, 1976. 3

[12] X. Gao, R. Wang, N. Demmel, and D. Cremers. LDSO: Direct sparse odometry with loop closure. In *International Conference on Intelligent Robots and Systems (IROS)*, 2018. 2, 3, 7, 8

[13] R. Glowinski and A. Marroco. Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires. *Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique*, 9(2):41–76, 1975. 3

[14] C. Ham, S. Lucey, and S. Singh. Proxy templates for inverse compositional photometric Bundle Adjustment. *arXiv preprint arXiv:1704.06967*, 2017. 2

[15] M. Hong, Z.-Q. Luo, and M. Razaviyayn. Convergence analysis of Alternating Direction Method of Multipliers for a family of nonconvex problems. *SIAM J. Optim.*, 26:337–364, 2016. 3

[16] K. Kanatani and D. D. Morris. Gauges and Gauge transformations for uncertainty description of geometric structure with indeterminacy. *IEEE Trans. Inform. Theory*, 47:2017–2028, 2001. 6

[17] J. Kannala and S. S. Brandt. A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 28(8):1335–1340, 2006. 7

[18] C. Kerl, J. Sturm, and D. Cremers. Dense visual slam for rgb-d cameras. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 2100–2106. IEEE, 2013. 2

[19] R. Mur-Artal and J. D. Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *Transactions on Robotics (T-RO)*, 33(5):1255–1262, 2017. 1, 2

[20] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. In *International Conference on Computer Vision (ICCV)*, pages 2320–2327. IEEE, 2011. 2

[21] K. Ni, D. Steedly, and F. Dellaert. Out-of-core bundle adjustment for large-scale 3d reconstruction. In *International Conference on Computer Vision (ICCV)*, pages 1–8. IEEE, 2007. 3, 4

[22] T. Schops, T. Sattler, and M. Pollefeys. Bad slam: Bundle adjusted direct rgb-d slam. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 134–144, 2019. 2

[23] W. Shi, Q. Ling, G. Wu, and W. Yin. EXTRA: An exact first-order algorithm for decentralized consensus optimization. *SIAM J. Optim.*, 25:944–966, 2015. 8

[24] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin. On the linear convergence of the ADMM in decentralized consensus optimization. *IEEE Trans. Signal Process.*, 62:1750–1761, 2014. 8

[25] J. Stühmer, S. Gumhold, and D. Cremers. Real-time dense geometry from a handheld camera. In *Pattern Recognition (DAGM)*, pages 11–20. Springer, 2010. 2

[26] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 573–580. IEEE, 2012. 7

[27] C. Tang and P. Tan. Ba-net: Dense bundle adjustment network. In *International Conference on Learning Representations (ICLR)*, 2019. 2

[28] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle Adjustment — a modern synthesis. In *International Workshop on Vision Algorithms*, pages 298–372, 1999. 1, 2, 4, 5, 6

[29] V. Usenko, N. Demmel, and D. Cremers. The double sphere camera model. In *International Conference on 3D Vision (3DV)*, pages 552–560. IEEE, 2018. 7

[30] V. Usenko, N. Demmel, D. Schubert, J. Stückler, and D. Cremers. Visual-inertial mapping with non-linear factor recovery. *arXiv preprint arXiv:1904.06504*, 5, 2019. 7

[31] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz. Multicore bundle adjustment. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3057–3064. IEEE, 2011. 3

[32] L. Xiao and S. Boyd. Fast linear iterations for distributed averaging. *Syst. Control Lett.*, 53:65–78, 2004. 8

[33] R. Zhang, S. Zhu, T. Fang, and L. Quan. Distributed very large scale Bundle Adjustment by global camera consensus. In *International Conference on Computer Vision (ICCV)*, pages 29–38. IEEE, 2017. 2, 3, 4, 5, 8

[34] S. Zhu, R. Zhang, L. Zhou, T. Shen, T. Fang, P. Tan, and L. Quan. Very large-scale global SfM by distributed motion averaging. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 3, 4

[35] J. Zubizarreta, I. Aguinaga, and J. M. M. Montiel. Direct sparse mapping. *Transactions on Robotics (T-RO)*, pages 1–8, 2020. 1, 2, 16

Distributed Photometric Bundle Adjustment Supplementary Material

Nikolaus Demmel, Maolin Gao, Emanuel Laude, Tao Wu and Daniel Cremers
Technical University of Munich
{nikolaus.demmel, maolin.gao, emanuel.laude, tao.wu, cremers}@tum.de

In this supplementary material we include the following additional content to support the claims of the main paper. While not essential for the understanding, we hope this is useful for practitioners that want to implement our approach and offers more detailed insights into the experiments and theory. We describe our balanced on-graph k-means clustering of host frames in Section A, followed by implementation details of our distributed photometric bundle adjustment including closed-form solutions for the gauge aware consensus update in Section B. Furthermore, we give detailed information about map and cluster sizes for all the sequences of the *EuRoC* dataset in Section C. We expand upon the qualitative results in Section D in the form of trajectory plots for all sequences, as well as a discussion of odometry failure cases. We also include quantitative results expanded to show values for each sequence instead of the averages found in the main paper. Section E presents an additional ablation study regarding different initializations and, finally, Section F provides a convergence proof for our distributed optimization.

A. Balanced On-Graph k-Means Clustering

As discussed in Section 3.3, our distributed PBA model requires a partitioning of host frames into clusters as input, *i.e.* $\mathcal{F} = \cup_{l=1}^{L} \mathcal{F}^l$. In our implementation we rely on balanced on-graph k-means clustering on the observation graph with pairwise Euclidean distances of initial camera positions as edge weights. We summarize this approach in the following for the interested reader, but alternative algorithms to partition the host frames could be employed as well. Clusters must be connected, and should be balanced as well as minimize the number of induced variable copies — three conflicting goals. Our approach in practice guarantees the former two properties, while the latter is approximated.

In the observation graph $\mathcal{G} = (\mathcal{F}, \mathcal{E})$ with keyframes $\mathcal{F}$ as nodes and undirected edges $\mathcal{E}$, two keyframes are connected if they share an observation. Formally, for host frame $i \in \mathcal{F}$ we denote the set of target frames as:

$$\mathcal{T}_i := \cup_{\mathbf{q} \in \mathcal{P}_i} \mathcal{Q}_{\mathbf{q}}, \tag{15}$$

i.e. the union of target frames $\mathcal{Q}_{\mathbf{q}}$ for all hosted points $\mathbf{q}$. With this, the set of edges in the observations graph can be defined as:

$$\mathcal{E} := \{\{i, j\} \subseteq \mathcal{F} : j \in \mathcal{T}_i \vee i \in \mathcal{T}_j\}. \tag{16}$$

If an edge is "cut" by the clustering, a target-frame copy has to be included in one or both clusters (but the number of copies is usually much less than the number of cut edges, since multiple edges can lead to the same required copy). Hence, the compact clusters produced by our k-means algorithm are associated with small overlapping regions and few induced copies.

For the purpose of clustering, we denote the shortest-path distance between nodes $i, j \in \mathcal{F}$ as $D(i, j)$, using the Euclidean distance between initial 3D camera center positions as edge weight. Using the on-graph distance $D(i, j)$, we propose the following extension to classical Euclidean k-means clustering of the 3D camera locations, where we enforce upper and lower bounds on the size of the clusters. We denote the L cluster centers by $\mu_l \in \mathcal{F}$, *i.e.* they are restricted to graph nodes, not arbitrary 3D positions. The cluster assignment is encoded by matrix $y \in \{e^1, \dots, e^L\}^{|\mathcal{F}|} \subset \mathbb{R}^{L \times |\mathcal{F}|}$ [3], where $e^l \in \{0, 1\}^L$ denotes the lth indicator vector. This means that for each frame i the assignment to cluster l is unique and denoted by $y_{li} = 1$.

The balancing can be phrased as integer linear constraints on the cluster assignment:

$$\mathcal{Y} := \Bigg\{ y \in \{e^1, \dots, e^L\}^{|\mathcal{F}|} : \\ b_{\text{low}} \leq \sum_{i \in \mathcal{F}} y_{li} \leq b_{\text{high}}, \forall\, 1 \leq l \leq L \Bigg\}. \tag{17}$$

Here, b_{low} and b_{high} denote hard lower and upper bounds on the number of vertices assigned to each cluster. We optimize the following cost function in an alternating fashion:

$$\min_{\substack{(\mu_1, \dots, \mu_L) \in \mathcal{F}^L \\ y \in \mathcal{Y}}} \sum_{i \in \mathcal{F}} \sum_{l=1}^{L} D^2(i, \mu_l) y_{li}. \tag{18}$$

More precisely, in every iteration we alternate the updates:

$$y^{t+1} = \arg\min_{y \in \mathcal{Y}} \sum_{i \in \mathcal{F}} \sum_{j=1}^{L} D^2(i, \mu_l^t) y_{li}, \tag{19}$$

$$\mu_l^{t+1} = \arg\min_{\mu_l \in \mathcal{F}} \sum_{i \in \mathcal{F}} D^2(i, \mu_l) y_{li}^{t+1}. \tag{20}$$

The y-update (19) yields a simple integer linear program. The problem can be reformulated as a min-cost-flow problem [3], for which efficient algorithms exist. In practice, we solve the LP-relaxation of the integer linear program using the simplex method and round the solution (if required). The centroid update (20) can be solved efficiently by exhaustive search: For each cluster l and each potential centroid $j \in \mathcal{F}$ we compute the sum of (precomputed) squared shortest-path distances to all vertices i assigned to cluster l and then select the vertex j for which this sum is minimal.

Finally, in case the solution contains a cluster whose induced subgraph is not fully connected, we iteratively fuse the smallest connected component of that cluster with a neighboring cluster, as a post-processing. This may result in an infeasible solution (w.r.t. balancing), but in practice we observe that the subgraphs after the k-means clustering are almost always already connected. Intuitively, the reason is that without the balancing constraints, on-graph distances — unlike 3D Euclidean distances — would lead to guaranteed connected clusters.

B. Distributed PBA Implementation Details

In this section we discuss some implementation details going beyond the description in Section 4.2. This is not essential to understand the proposed algorithm, but intended to assist others when attempting to reproduce our results. Algorithm 1 summarizes the procedure in pseudo code and Table 4 lists important hyper-parameters and their default values in our experiments, unless otherwise indicated.

Global Update Remember that we formulate the penalties (8) for rotation matrices in the embedding space $\mathbb{R}^{3\times3}$ and need to ensure that global variables $\mathbf{R}_i$ and gauge variables $\widetilde{\mathbf{R}}^l$ stay in their domain $\mathrm{SO}(3)$ in each update step. To this end, we make use of the metric projection of $\mathbf{M} \in \mathbb{R}^{3\times3}$ onto $\mathrm{SO}(3)$ under the Frobenius norm $\|\cdot\|_F$,

$$\mathrm{Proj}_{\mathrm{SO}(3)}(\mathbf{M}) := \arg\min_{\mathbf{R} \in \mathrm{SO}(3)} \|\mathbf{R} - \mathbf{M}\|_F, \tag{21}$$

which can be efficiently computed using singular value decomposition.

Given gauge variables $\{(\widetilde{\mathbf{R}}^l, \widetilde{\mathbf{t}}^l, \widetilde{s}^l, \widetilde{a}^l)\}$, the minimization of (13) over the global variables $\{(\mathbf{R}_i, \mathbf{t}_i, a_i, b_i)\}$ is solved by either arithmetic or Fréchet means under $\|\cdot\|_F$:

$$\mathbf{R}_i = \mathrm{Proj}_{\mathrm{SO}(3)}\left(\frac{1}{m_i} \sum_{l: i \in \mathcal{C}^l} \mathbf{R}_i^l \widetilde{\mathbf{R}}^l\right), \tag{22}$$

$$\mathbf{t}_i = \frac{1}{m_i} \sum_{l: i \in \mathcal{C}^l} (\mathbf{R}_i^l \widetilde{\mathbf{t}}^l + \widetilde{s}^l \mathbf{t}_i^l), \tag{23}$$

$$a_i = \frac{1}{m_i} \sum_{l: i \in \mathcal{C}^l} (a_i^l + \widetilde{a}^l), \tag{24}$$

$$b_i = \frac{1}{m_i} \sum_{l: i \in \mathcal{C}^l} b_i^l, \tag{25}$$

where m_i is the number of clusters that frame i is copied to, *i.e.* $m_i = \sum_{l: i \in \mathcal{C}^l} 1$.

Gauge Update Given global variables $\{(\mathbf{R}_i, \mathbf{t}_i, a_i, b_i)\}$, the minimization of (13) over the gauge variables $\{(\widetilde{\mathbf{R}}^l, \widetilde{\mathbf{t}}^l, \widetilde{s}^l, \widetilde{a}^l)\}$ leads to the following updates:

$$\widetilde{\mathbf{R}}^l = \mathrm{Proj}_{\mathrm{SO}(3)}\left(\frac{1}{|\mathcal{C}^l|} \sum_{i \in \mathcal{C}^l} (\mathbf{R}_i^l)^\top \mathbf{R}_i\right), \tag{26}$$

$$\widetilde{a}^l = \frac{1}{|\mathcal{C}^l|} \sum_{i \in \mathcal{C}^l} (a_i - a_i^l), \tag{27}$$

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{12}^\top & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} \widetilde{\mathbf{t}}^l \\ \widetilde{s}^l \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}, \tag{28}$$

Algorithm 1 Distributed Photometric Bundle Adjustment

1: Partition the set of keyframes into clusters $\mathcal{F} = \cup_{l=1}^{L} \mathcal{F}_l$. Initialize the global and local variables from the loop-closing odometry, and for each $l \in \{1, ..., L\}$ the gauge variables $(\widetilde{\mathbf{R}}^l, \widetilde{\mathbf{t}}^l, \widetilde{s}^l, \widetilde{a}^l) = (I_3, 0, 1, 0)$.
2: **while** not converged **do**
3: For each $l \in \{1, ..., L\}$, update the local variables $\{(\mathbf{R}_i^l, \mathbf{t}_i^l, a_i^l, b_i^l) : i \in \mathcal{C}^l\} \cup \{d_{\mathbf{q}}^l : \mathbf{q} \in \cup_{i \in \mathcal{C}^l} \mathcal{P}_i^l\}$ by applying non-linear least squares solver to (9).
4: **while** not converged **do**
5: Update the global variables $\{(\mathbf{R}_i, \mathbf{t}_i, a_i, b_i) : i \in \mathcal{F}\}$ by Eq. (22)–(25).
6: Update the gauge variables $\{(\widetilde{\mathbf{R}}^l, \widetilde{\mathbf{t}}^l, \widetilde{s}^l, \widetilde{a}^l) : l \in \{1, ..., L\}\}$ by Eq. (26)–(28).
7: **end while**
8: Perform the gauge transform on the local variables $\{(\mathbf{R}_i^l, \mathbf{t}_i^l, a_i^l, b_i^l) : i \in \mathcal{C}^l\} \cup \{d_{\mathbf{q}}^l : \mathbf{q} \in \cup_{i \in \mathcal{C}^l} \mathcal{P}_i^l\}$ using Eq. (11)–(12). Reset $(\widetilde{\mathbf{R}}^l, \widetilde{\mathbf{t}}^l, \widetilde{s}^l, \widetilde{a}^l) = (I_3, 0, 1, 0)$ for each l.
9: Increase the penalty weights: $\rho_{\mathbf{R}} \leftarrow \beta\rho_{\mathbf{R}}$, $\rho_{\mathbf{t}} \leftarrow \beta\rho_{\mathbf{t}}$, $\rho_a \leftarrow \beta\rho_a$, $\rho_b \leftarrow \beta\rho_b$.
10: **end while**

parameter	description	value
	adding observations (Section 3)	
relative scale	the permissible range of relative scale of inverse depth in host and target frame while adding new photometric observations	$(0.7, 1.3)$
boundary margin	distance in pixels from the image boundary for a projected point to be considered *in bounds* when adding observations	10
	model (Section 4.1)	
γ	huber parameter for photometric residual	5
	algorithm (Section 4.2)	
$N^{\text{outer}}_{\text{max}}$	maximum number of outer iterations (see Algorithm 1)	250
$N^{\text{gauge}}_{\text{max}}$	maximum number of iterations of global/gauge loop (see Algorithm 1)	5
$N^{\text{local}}_{\text{max}}$	maximum number of LM iterations for local solver	10
$N^{\text{local}}_{\text{max success}}$	maximum number of successful LM iterations for local solver	2
$\lambda^{\text{init}}_{\text{max}}$	maximum initial LM-dampening factor for local solver	10^{-4}
β	rate of increase of penalty parameters $\rho_\bullet$	1.06
ρ^0	base-factor to initialize penalty parameters $\rho_\bullet$	10^2
ρ^{250}	final base-factor for penalty parameters $\rho_\bullet$ after 250 iterations	$\sim 2.1 \times 10^8$
$\alpha_{\mathbf{R}}$	scale factor for penalty parameter $\rho_{\mathbf{R}}$	1
$\alpha_{\mathbf{t}}$	scale factor for penalty parameter $\rho_{\mathbf{t}}$	1
α_a	scale factor for penalty parameter ρ_a	1
α_b	scale factor for penalty parameter ρ_b	0.02
	clustering (Section A)	
b_{tol}	tolerance for balancing constraint; how much we allow the cluster size to deviate from a perfectly balanced result, i.e. $b_{\text{low}} = \frac{\lvert\mathcal{F}\rvert}{L}(1 - b_{\text{tol}})$ and $b_{\text{high}} = \frac{\lvert\mathcal{F}\rvert}{L}(1 + b_{\text{tol}})$	0.1

Table 4. Important hyper-parameters and their default values.

with

$$\mathbf{A}_{11} := |\mathcal{C}^l| I_3, \tag{29}$$

$$\mathbf{A}_{12} := \sum_{i \in \mathcal{C}^l} (\mathbf{R}^l_i)^\top \mathbf{t}^l_i, \tag{30}$$

$$\mathbf{A}_{22} := \sum_{i \in \mathcal{C}^l} \left\| \mathbf{t}^l_i \right\|^2_2, \tag{31}$$

$$\mathbf{b}_1 := \sum_{i \in \mathcal{C}^l} (\mathbf{R}^l_i)^\top \mathbf{t}_i, \tag{32}$$

$$\mathbf{b}_2 := \sum_{i \in \mathcal{C}^l} \mathbf{t}_i^\top \mathbf{t}^l_i. \tag{33}$$

The linear system (28) can be efficiently solved using the Schur complement:

$$\mathbf{A}_s := \mathbf{A}_{22} - \mathbf{A}_{12}^\top \mathbf{A}_{11}^{-1} \mathbf{A}_{12}, \tag{34}$$

$$\mathbf{b}_s := \mathbf{b}_2 - \mathbf{A}_{12}^\top \mathbf{A}_{11}^{-1} \mathbf{b}_1, \tag{35}$$

$$\widetilde{s}_l = \frac{\mathbf{b}_s}{\mathbf{A}_s}, \tag{36}$$

$$\widetilde{\mathbf{t}}_l = \mathbf{A}_{11}^{-1}(\mathbf{b}_1 - \mathbf{A}_{12}\widetilde{s}_l). \tag{37}$$

There is a theoretical possibility of getting a negative scale factor $\widetilde{s}_l$ in (36). In practice, this occurs only when the whole optimization diverges due to very poor initialization. In our implementation, we catch this case and assign $\widetilde{s}_l \leftarrow \frac{\widetilde{s}_l}{2}$ instead.

Local Sub-problem Solver To perform the local update we solve (9) using a standard Levenberg-Marquardt (LM) solver as implemented in the Ceres library [1]. For every outer iteration of Algorithm 1, we perform for each cluster up to $N^{\text{local}}_{\text{max}} = 10$ LM-iterations. Those iterations can either be successful, if they decrease the cost, or unsuccessful, otherwise. In the latter case, the dampening factor λ is increased. In our experiments we found that it is not essential to solve the sub-problems as accurately as possible in every outer iteration. Instead, it is sufficient to achieve a number of successful LM steps, and we thus abort the local solver after $N^{\text{local}}_{\text{max success}} = 2$ successful steps. What we did find to be essential is ensuring a sufficiently small initial dampening factor λ to allow progress. At the start of every outer iteration, we therefore take the initial λ as the minimum of $\lambda^{\text{init}}_{\text{max}} = 10^{-4}$ and the smallest λ, that lead to a successful step for this cluster in the previous outer iteration.[3]

Initializing Penalty Parameters The penalty parameters $\rho_\bullet$ are increased by factor β in every iteration (see Algorithm 1). To make the initial penalty parameters independent of the map and cluster size, we initialize $\rho_\bullet = \kappa \alpha_\bullet \rho^0$ for all $\bullet \in \mathbf{R}, \mathbf{t}, a, b$. Here, $\rho^0 = 10^2$ is the chosen initial ρ-base-factor, $\alpha_\bullet$ is a scale factor compensating for different dimensions of the different variable types, and κ is the average number of times each local frame variable appears in a photometric residual, to balance the penalty terms and the photometric cost. For $\beta = 1.06$, after 250 iterations we have a final ρ-base-factor of $\rho^{250} = \rho^0 \beta^{250} \approx 2.1 \times 10^8$.

[3] Note that in Ceres we configure the *initial trust region radius* $\frac{1}{\lambda}$.

	odom				pgo
	#l-c	#kf	#lm	#obs	#obs
eurocMH01	16	481	308039	2291053	114%
eurocMH02	7	382	249158	1919379	111%
eurocMH03	19	650	376060	2992960	116%
eurocMH04	0	410	193002	1477691	100%
eurocMH05	14	400	214937	1682726	116%
eurocV101	21	695	449646	3639173	113%
eurocV102	11	673	350979	2764613	107%
eurocV103	1	949	392339	3228913	101%
eurocV201	7	412	246423	2013929	107%
eurocV202	22	868	429225	3322179	116%
eurocV203	0	1083	325252	2660546	100%

Table 5. Size of the optimization problem for all *EuRoC* sequences. For the initialization from odometry (*odom*) we show: number of loop-closure edges (*#l-c*), number of keyframes (*#kf*), number of landmarks (*#lm*) and number of photometric landmark observations (*#obs*). We also show the total number of observations after pose-graph optimization and augmentation of the map with additional observations across loop-closure edges (see Section 3) as a percentage of the number of observations in odometry (*pgo*). All global PBA variants use the same observations as shown for *pgo*.

Single-copy Cameras For cameras that only have a single copy, i.e. only appear in a single cluster, the global update is an average over one variable and thus effectively assigns the global variable the value of that one local copy (respecting the gauge transform). The penalty terms in the local update thus effectively only act as dampening on these variables. We found that one can treat such frame variables as *purely local*, ignoring the penalty term. It can be seen as setting $\rho = 0$ for these variables. Then, in the local update there is no penalty cost and the gauge update is affected only by variables that have two copies or more.

C. Dataset Details

Table 5 gives an overview of the size of the optimization problem for all sequences in the *EuRoC* dataset, where we have $\sim 10^3$ keyframes, $\sim 10^5$ landmarks and $\sim 10^6$ observations. The two sequences where odometry failed can also be identified as the ones without validated loop-closure edges. Our scheme to augment the map with additional observations across loop-closure edges before global PBA results in an up to 16% increase in the number of observations.

Table 6 gives more detail on the experiments with different number of clusters (see Table 3 and Table 8 for accuracy and convergence results). It shows the average effective cluster size (count of local cameras / landmarks / observations), as resulting from our proposed balanced on-graph k-means clustering, for varying number of clusters. The total map size is always the same for each sequence. As we increase the number of clusters, they naturally become smaller, but the relative overlap and thus the average number of copies per camera / landmark / observation increases. For example, with five clusters, we have an average number of copies of $\frac{0.27}{0.20} \approx 1.4$ per global camera, which also means that more than half of the global cameras only have one copy, i.e. appear only in a single cluster. For 30 clusters however, the average number of copies per global camera is higher, around $\frac{0.09}{0.03} \approx 3$.

	pba	dpba				
# clusters		3	5	10	15	30
no-overlap ratio	1.00	0.33	0.20	0.10	0.07	0.03
# cam / cluster	1.00	0.42	0.27	0.16	0.13	0.09
# lm / cluster	1.00	0.41	0.27	0.16	0.13	0.09
# obs / cluster	1.00	0.41	0.27	0.16	0.12	0.08

Table 6. Average number of cameras (*cam*), landmarks (*lm*) and observations (*obs*) per cluster, as a fraction of the total counts for the whole map (*pba*). Values are geometric averages over all *EuRoC* sequences and shown for different number of clusters (*dpba*), while map size remains constant for each sequence. For comparison, a theoretical fraction assuming no overlap and no duplication between clusters, which equals $\frac{1}{\#\,clusters}$, is shown as *no-overlap ratio*. Note that for successful distributed optimization, we require *some* overlap between the clusters.

D. Additional Results

Visualized Trajectories and Odometry Failure In Table 10 we show estimated trajectories before and after global distributed photometric bundle adjustment for all sequences. It can be seen that the quantitative improvement in ATE (see Table 1) is also obvious in the trajectory alignment. In general, the trajectories after PBA fit the ground-truth much better.

For the failure cases (*MH04*, *V203*) of the odometry (our initialization), it can be seen that while parts of the trajectories fit the ground-truth very well, when the odometry fails, there is significant rotation- and translation-drift (first half of *MH04*) or scale-drift (last half of *V203*). Since the definition of the ATE uses least-squares alignment based on all poses, this results in a large RMS translational error. In this case, global PBA (without additional preprocessing) does not help, since it relies on good-enough initialization and is otherwise prone to get stuck in local minima of the highly non-convex cost function. However, given the initialization, PBA still behaves reasonably: the accurate part of the trajectory shows significant improvement over the odometry poses. To show this, we provide additional plots of these trajectories aligned using only the last (*MH04*) or first (*V203*) half of the trajectory. Computing the ATE for these halfs shows that our distributed global PBA reduces the error substantially from 0.084 to 0.050 meters (*MH04*), and from 0.255 to 0.080 meters (*V203*), even though no loop-closure is detected.

Quantitative Results In Table 7 and Table 8 we expand upon the results presented in Table 2 and Table 3, respectively, by showing values for all individual sequences, in-

	pba	dpba			nogauge
ρ-increase rate β		1.06	1.1	1.2	1.06
# iterations (avg)	148	250	150	75	250
	0.400	**0.416**	*0.425*	0.442	0.445
MH01	1.0000	**1.0012**	*1.0015*	1.0021	1.0027
	1.0000	**0.9960**	*0.9963*	0.9970	0.9974
	0.715	0.734	0.730	**0.705**	*0.708*
MH02	1.0000	**1.0014**	*1.0019*	1.0033	1.0044
	1.0000	**0.9967**	*0.9971*	0.9985	0.9994
	0.447	**0.501**	*0.558*	0.641	0.700
MH03	1.0000	**1.0029**	*1.0039*	1.0078	1.0080
	1.0000	**1.0008**	*1.0019*	1.0058	1.0059
	1.028	*1.032*	*1.032*	**1.030**	1.043
MH04	1.0000	*0.9996*	**0.9995**	0.9999	1.0003
	1.0000	**0.9984**	**0.9984**	*0.9987*	0.9988
	0.791	0.790	0.767	*0.721*	**0.688**
MH05	1.0000	**1.0031**	*1.0050*	1.0131	1.0076
	1.0000	**0.9972**	*0.9993*	1.0073	1.0017
	0.744	0.753	*0.745*	0.747	**0.742**
V101	1.0000	**1.0036**	*1.0059*	1.0084	1.0068
	1.0000	**0.9988**	*1.0008*	1.0037	1.0019
	0.286	**0.275**	**0.275**	*0.278*	0.309
V102	1.0000	**1.0076**	*1.0179*	1.0369	1.0365
	1.0000	**1.0033**	*1.0135*	1.0324	1.0316
	0.635	**0.633**	*0.637*	0.653	0.686
V103	1.0000	**1.0002**	**1.0002**	*1.0005*	1.0014
	1.0000	**0.9993**	*0.9994*	0.9996	1.0005
	0.505	**0.609**	0.618	*0.617*	0.654
V201	1.0000	**1.0036**	*1.0040*	1.0072	1.0194
	1.0000	**0.9986**	*0.9990*	1.0015	1.0082
	1.118	1.199	1.180	*1.055*	**0.950**
V202	1.0000	**1.0024**	*1.0032*	1.0061	1.0035
	1.0000	**0.9981**	*0.9989*	1.0018	0.9991
	0.996	**0.970**	*0.977*	0.984	1.009
V203	1.0000	*0.9992*	**0.9991**	0.9993	1.0031
	1.0000	0.9986	*0.9983*	**0.9980**	1.0002

Table 7. ATE reduction as a fraction of initial (*pgo*) error, and final global and local cost as a fraction of the final cost the central solver (*pba*, ours) achieved. The three metrics shown for each entry are top-to-bottom: "fraction of *pgo*-ATE", "cost global" and "cost local" ("fraction of *odom*-ATE" is not shown, as it is qualitatively similar). We show the same results as in Table 2, but for each individual sequence, so please refer to its caption. Highlights for **smallest** and *second smallest* value in each row exclude the first column (*pba*). For "fraction of *pgo*-ATE" and "cost global" smaller is better. For "cost local" values close to "cost global" show good consensus between local and global variables.

stead of averages over all.

Generally, the same conclusions can be drawn. From Table 7 it can be seen that a higher ρ-increase rate β leads to faster optimization, at a slight expense of final accuracy and achieved cost value, which allows to trade off speed versus accuracy depending on the application. Applying our novel gauge aware consensus update always leads to lower final cost and generally to higher accuracy. Interestingly, in a few cases the naive consensus update (*nogauge*) has lower final ATE and it seems ATE values are not always congruent with the optimized cost. Regardless, enabling the gauge update does lead to more consistent results and behaviour more similar to the central *pba*.

The tradeoff of having more clusters is to down-scale the sub-problem size, while being prone to lower overall accuracy due to introduction of more duplicates with consensus

	pba	dpba			
# clusters		3	5	10	15	30
	0.400	**0.411**	*0.416*	0.644	0.664	0.720
MH01	1.0000	**1.0012**	**1.0012**	*1.0024*	1.0028	1.0026
	1.0000	0.9986	0.9960	0.9969	*0.9958*	**0.9949**
	0.715	**0.676**	0.734	0.718	*0.709*	0.730
MH02	1.0000	**1.0006**	*1.0014*	1.0018	1.0015	1.0016
	1.0000	0.9979	0.9967	0.9962	*0.9951*	**0.9937**
	0.447	*0.545*	**0.501**	0.662	0.722	0.774
MH03	1.0000	*1.0049*	**1.0029**	1.0084	1.0088	1.0089
	1.0000	*1.0011*	**1.0008**	1.0028	1.0020	**1.0008**
	1.028	1.027	1.032	1.035	*1.018*	**0.982**
MH04	1.0000	*0.9997*	**0.9996**	1.0003	1.0013	1.0036
	1.0000	0.9991	0.9984	0.9966	*0.9954*	**0.9936**
	0.791	0.768	0.790	0.729	**0.693**	*0.706*
MH05	1.0000	**1.0030**	*1.0031*	1.0086	1.0089	1.0100
	1.0000	1.0001	**0.9972**	1.0022	0.9993	*0.9981*
	0.744	0.749	0.753	0.750	*0.733*	**0.730**
V101	1.0000	**1.0025**	*1.0036*	1.0047	1.0059	1.0057
	1.0000	0.9993	0.9988	0.9989	*0.9966*	**0.9964**
	0.286	**0.272**	*0.275*	0.282	0.313	0.284
V102	1.0000	**1.0021**	*1.0076*	1.0115	1.0154	1.0205
	1.0000	**0.9986**	*1.0033*	1.0053	1.0072	1.0088
	0.635	**0.630**	*0.633*	0.674	0.698	0.732
V103	1.0000	**1.0001**	*1.0002*	1.0009	1.0012	1.0020
	1.0000	0.9996	0.9993	0.9981	0.9975	**0.9951**
	0.505	**0.534**	0.609	0.635	0.596	0.598
V201	1.0000	1.0035	1.0036	**1.0032**	*1.0033*	**1.0032**
	1.0000	0.9989	0.9986	0.9957	*0.9941*	**0.9915**
	1.118	1.074	1.199	0.985	*0.917*	0.923
V202	1.0000	*1.0053*	**1.0024**	1.0083	1.0112	1.0110
	1.0000	1.0026	**0.9981**	1.0015	1.0013	*0.9998*
	0.996	*0.985*	**0.970**	0.990	0.997	1.002
V203	1.0000	**0.9973**	0.9992	*0.9981*	1.0013	1.0043
	1.0000	0.9970	0.9986	0.9963	0.9964	**0.9939**

Table 8. ATE reduction for our distributed *dpba* with different number of clusters, compared to our central *pba*. The three metrics shown for each entry are top-to-bottom: "fraction of *pgo*-ATE", "cost global" and "cost local" ("fraction of *odom*-ATE" is not shown, as it is qualitatively similar). We show the same results as in Table 3, but for each individual sequence, so please refer to its caption. Highlights for **best** and *second best* value in each row exclude the first column (*pba*).

augment map	–	yes				no			
init state from	–	pgo		odom		pgo		odom	
method	pgo	pba	dpba	pba	dpba	pba	dpba	pba	dpba
odom-ATE frac.	0.56	0.36	0.38	0.50	0.54	0.60	0.57	0.60	0.67

Table 9. Comparing different initializations by ATE reduction (fraction of odometry ATE) averaged over all EuRoC sequences.

constraints. In the experiments (Table 8), we observe that deterioration of accuracy due to increase of number of clusters appears to be mild.

E. Notes on Initialization

In our experience, global photometric BA is very sensitive to good initialization, more so than geometric BA. This is in line with the discussion in [8]. There are two aspects of using the result *after* loop closure as initialization, as opposed to the pure odometry result. First, we use the detected loop closures to augment the map by adding observations between temporally distant keyframes. Second, we use the updated state after PGO as initialization for

global PBA. Both are critical. The first ensures that we can actually correct drift and don't just batch-optimize the sliding window open loop graph. The second is important for the global PBA to find good minima. Table 9 shows additional results from the experiments in the main paper (the first three columns correspond to values in Table 2). While all global optimizations improve over pure odometry, we can clearly see that not augmenting the map doesn't bring the full benefit (results are worse than just running PGO) and that initializing the global PBA with the corrected state from PGO is also essential for convergence (the region of convergence could be increased a bit with a coarse-to-fine approach [35]). Note that the central and distributed variants in each case show similar behaviour.

F. Convergence Proof

In the following we prove that under mild assumptions on sufficient descent and sub-problem accuracy, our proposed distributed photometric BA converges to a stationary point.

Notation Let $\mathcal{P}^l := \cup_{i \in \mathcal{C}^l} \mathcal{P}_i^l$ be the set of all points that appear in cluster l. To streamline the notation, we stack all variables (including the rotation matrices) into column vectors, such that the norms used in the definition of the penalty cost become the regular 2-norm. We refer to the local variables $\{\theta_i^l : i \in \mathcal{C}^l\} \cup \{d_{\mathbf{q}}^l : \mathbf{q} \in \mathcal{P}^l\}$ for cluster l as $(\theta^l, d^l) \in \mathcal{K}^l :\simeq (\mathrm{SO}(3) \times \mathbb{R}^3 \times \mathbb{R} \times \mathbb{R})^{\mathcal{C}^l} \times \mathbb{R}_{\geq 0}^{\mathcal{P}^l}$, to the local variables $(\theta^1, \dots, \theta^L, d^1, \dots, d^L)$ for all clusters as (θ, d), to the global variables $\{\Theta_i : i \in \mathcal{F}\}$ as $\Theta \in \mathcal{B} :\simeq (\mathrm{SO}(3) \times \mathbb{R}^3 \times \mathbb{R} \times \mathbb{R})^{\mathcal{F}}$, and to the local variables under gauge parameterization $\{\widetilde{\theta}_i^l(\theta_i^l, \xi^l) : i \in \mathcal{C}^l\} \cup \{\widetilde{d}_{\mathbf{q}}^l(d_{\mathbf{q}}^l, \xi^l) : \mathbf{q} \in \mathcal{P}^l\}$ for cluster l as $(\widetilde{\theta}^l(\theta^l, \xi^l), \widetilde{d}^l(d^l, \xi^l)) \in \mathcal{K}^l$. We denote the set of all possible gauge transformations as $\mathcal{X} := \mathrm{SO}(3) \times \mathbb{R}^3 \times \mathbb{R}_{>0} \times \mathbb{R}$, *i.e.* $\xi^l \in \mathcal{X}$. The subset of variables in Θ corresponding to θ^l for cluster l is denoted as $A^l\Theta$, where matrix A^l selects the rows from Θ that correspond to variables appearing in cluster l (the rows of A^l are a subset of the rows of the identity matrix $I_{|\mathcal{B}|}$). We define the diagonal matrix W with positive penalty weights $\rho_{\mathbf{R}}$, $\rho_{\mathbf{t}}$, ρ_a, ρ_b on the diagonal in such a way that we can multiply $W\Theta_i$, and likewise diagonal matrix W^l with repeated diagonal blocks W such that we can multiply $W^l\theta^l$.

With that, we can abbreviate the local bundle adjustment cost with:

$$F_{\mathrm{BA}}^l(\theta^l, d^l) := \sum_{i \in \mathcal{C}^l} \sum_{\mathbf{q} \in \mathcal{P}_i^l} \sum_{j \in \mathcal{Q}_{\mathbf{q}}^l} \frac{1}{n_{i,\mathbf{q},j}} F_{i,\mathbf{q},j}(\theta_i^l, d_{\mathbf{q}}^l, \theta_j^l), \tag{38}$$

and the penalty terms with:

$$P^l(\Theta, \theta^l) := \frac{1}{2} \|W^{l\,\frac{1}{2}}(A^l\Theta - \theta^l)\|^2 \tag{39}$$

$$= \frac{1}{2} \sum_{i \in \mathcal{C}^l} \|W^{\frac{1}{2}}(\Theta_i - \theta_i^l)\|^2. \tag{40}$$

Then, the splitting cost function reads:

$$F_{\mathrm{split}}^l(\theta^l, d^l, \Theta) := F_{\mathrm{BA}}^l(\theta^l, d^l) + P^l(\Theta, \theta^l), \tag{41}$$

$$F_{\mathrm{split}}(\theta, d, \Theta) := \sum_{l=1}^{L} F_{\mathrm{split}}^l(\theta^l, d^l, \Theta). \tag{42}$$

Algorithm Note that Algorithm 1 is procedural, similar to how one would implement it in practice. In particular, it factors the alternating minimization of (13) into an inner loop (lines 4–7), as it can be performed on the master without intermediate communication with the workers. However, as discussed earlier, the optimization of gauge variables can also be viewed as a subspace local update, where the local variables are constrained to the gauge orbit (of the bundle adjustment residuals). For the convergence analysis we use a more suitable view onto the same algorithm and make the "subspace update" interpretation explicit: In this modified formulation each iteration $t \in \{0, 1, ...\}$ comprises one local update followed by one global update. Let $\mathcal{T} \subset \{0, 1, ...\}$ be the iterations where we perform the full local update. In the proof we reason about the convergence of iterates in this subsequence, but this is not a restriction, since the distance between two consecutive $t \in \mathcal{T}$ is bounded by $N_{\max}^{\text{gauge}}$ (the inner loop in Algorithm 1 has at most $N_{\max}^{\text{gauge}}$ iterations). Note that in particular $0 \in \mathcal{T}$. For all other iterations $t \notin \mathcal{T}$ the subspace local update with gauge parameterization is used.

Specifically, for each cluster l, the local update without gauge constraint ($t \in \mathcal{T}$) reads:

$$(\theta^{l,t+1}, d^{l,t+1}) \in \underset{(\theta^l, d^l) \in \mathcal{K}^l}{\arg\min} \; F_{\mathrm{split}}^l(\theta^l, d^l, \Theta^t), \tag{43}$$

and with gauge constraint ($t \notin \mathcal{T}$) reads:

$$\xi_l^{t+1} \in \underset{\xi_l \in \mathcal{X}}{\arg\min} \; P^l(\Theta^t, \widetilde{\theta}^l(\theta^{l,t_k(t)+1}, \xi^l)), \tag{44}$$

and

$$(\theta^{l,t+1}, d^{l,t+1}) = \big(\widetilde{\theta}^l(\theta^{l,t_k(t)+1}, \xi_l^{t+1}), \widetilde{d}^l(d^{l,t_k(t)+1}, \xi_l^{t+1})\big), \tag{45}$$

where $t_k(t) = \max\{t_k \in \mathcal{T} : t_k \leq t\}$ denotes for a given iteration t the most recent iteration $t_k(t)$ where a full local update was performed. Note that we parameterize the gauge orbit with the local variables from iteration $t_k(t) + 1$ for

Algorithm 2 Distributed Photometric Bundle Adjustment Convergence Analysis

1: Partition the set of keyframes into clusters $\mathcal{F} = \cup_{l=1}^{L} \mathcal{F}_l$. Initialize the global and local variables at $t = 0$ from the loop-closing odometry.
2: **for** $t = 0, 1, \dots$ **do**
3: **for** $l \in \{1, ..., L\}$ **do**
4: **if** $t \in \mathcal{T}$ **then**
5: Update the local variables $(\theta^{l,t+1}, d^{l,t+1})$ by applying non-linear least squares solver to (43).
6: **else**
7: Update the gauge variables $\xi^{l,t+1}$ with the closed-form solution to Eq. (44).
8: Update the local variables $(\theta^{l,t+1}, d^{l,t+1})$ with (45).
9: **end if**
10: **end for**
11: Update the global variables Θ^{t+1} with the closed-form solution to (46).
12: **end for**

the optimization (44), since in Algorithm 1 we only apply the gauge normalization once after the inner loop (line 8). Finally, the global update in each iteration is:

$$\Theta^{t+1} \in \operatorname*{arg\,min}_{\Theta \in \mathcal{B}} \sum_{l=1}^{L} P^l(\Theta, \theta^{l,t+1}). \tag{46}$$

The above steps are summarized in Algorithm 2. One minor difference to Algorithm 1 is that there, at the end of each inner iteration, a gauge update is directly followed by the full local update of the subsequent outer iteration. While in practice this is useful to do, since the gauge update is cheap to compute, we skip this intermediate gauge update in Algorithm 2, as it does not change the optimality conditions of the full local update and hence has no influence on convergence. Moreover, we only consider the case of constant penalty weights ρ, since in practice, for a continuation scheme with increasing ρ like ours, one performs only a maximum number of steps of increasing ρ, leaving it constant thereafter if full convergence is desired.

Assumptions We assume that the subproblems are solved possibly inexactly, but such that there always hold relative error conditions w.r.t. the limiting subdifferential [36, Definition 8.3(b)]. For $t \in \mathcal{T}$ we have after the full local update

$$\begin{aligned}\epsilon_\theta^{l,t+1} \in \nabla F_{\mathrm{BA}}^l(\theta^{l,t+1}, d^{l,t+1}) + N_{\mathcal{K}^l}(\theta^{l,t+1}, d^{l,t+1}) \\ + (W^l(\theta^{l,t+1} - A^l\Theta^t), \mathbf{0}),\end{aligned} \tag{47}$$

for every cluster l, where $N_{\mathcal{K}^l}(\theta^{l,t+1}, d^{l,t+1})$ denotes the (limiting) normal cone of $\mathcal{K}^l$ at $(\theta^{l,t+1}, d^{l,t+1})$, and we assume bound

$$\|\epsilon_\theta^{l,t+1}\| \leq \gamma_1 \|(\theta^{l,t+1}, d^{l,t+1}) - (\theta^{l,t}, d^{l,t})\| \tag{48}$$

for some $\gamma_1 \in \mathcal{R}_{>0}$. After the global update, for any iteration t, we have:

$$\epsilon_\Theta^{t+1} = -\sum_{l=1}^{L} {A^l}^\top W^l(\theta^{l,t+1} - A^l\Theta^{t+1}) + N_{\mathcal{B}}(\Theta^{t+1}), \tag{49}$$

where similarly $N_{\mathcal{B}}(\Theta^{t+1})$ denotes the (limiting) normal cone of $\mathcal{B}$ at Θ^{t+1}, and we assume bound

$$\|\epsilon_\Theta^{t+1}\| \leq \gamma_1 \|\Theta^{t+1} - \Theta^t\|. \tag{50}$$

Furthermore, we assume the following descent conditions: For any $t \in \mathcal{T}$ we have:

$$\begin{aligned}F_{\mathrm{split}}(\theta^{t+1}, d^{t+1}, \Theta^t) - F_{\mathrm{split}}(\theta^t, d^t, \Theta^t) \\ \leq -\gamma_2 \|(\theta^{l,t+1}, d^{l,t+1}) - (\theta^{l,t}, d^{l,t})\|^2,\end{aligned} \tag{51}$$

for some $\gamma_2 \in \mathcal{R}_{>0}$ and for all iterations $t \in \{0, 1, \dots\}$:

$$\begin{aligned}F_{\mathrm{split}}(\theta^{t+1}, d^{t+1}, \Theta^{t+1}) - F_{\mathrm{split}}(\theta^{t+1}, d^{t+1}, \Theta^t) \\ \leq -\gamma_2 \|\Theta^{t+1} - \Theta^t\|^2.\end{aligned} \tag{52}$$

We remark that conditions (47)–(52) typically hold when our algorithm is applied in practice. Additionally, they could easily be enforced by solving quadratically perturbed problems in place of (43) and (46), obtained by adding proximal terms $\epsilon\|(\theta^l, d^l) - (\theta^{l,t}, d^{l,t})\|^2$ resp. $\epsilon\|\Theta - \Theta^t\|^2$ for some $\epsilon \in \mathcal{R}_{>0}$. Then, Fermat's rule [36, Theorem 10.1] yields the four conditions above.

If $t \notin \mathcal{T}$, we have by the properties of the gauge orbit:

$$\begin{aligned}F_{\mathrm{split}}(\theta^{t+1}, d^{t+1}, \Theta^t) - F_{\mathrm{split}}(\theta^t, d^t, \Theta^t) \\ = P^l(\Theta^t, \theta^{l,t+1}) - P^l(\Theta^t, \theta^{l,t}) \leq 0.\end{aligned} \tag{53}$$

Proposition F.1. *Let $t \in \{0, 1, \dots\}$ be the sequence of iterates generated by Algorithm 2 and assume conditions (47)–(52) hold. Then the cost function $F_{\mathrm{split}}(\theta^t, \Theta^t, d^t)$ is monotonically decreasing and as a result converges to some finite value. Furthermore every limit point of the sequence $\{(\theta^{t+1}, \Theta^{t+1} d^{t+1})\}_{t \in \mathcal{T}}$ is a stationary point of F_{split}.*

Proof. Let $[\![\cdot]\!]$ denote the Iverson bracket. We obtain by summing Inequalities (52) with (51) or (53) as

$$\begin{aligned}-\infty < F_{\mathrm{split}}(\theta^{t+1}, d^{t+1}, \Theta^{t+1}) - F_{\mathrm{split}}(\theta^t, d^t, \Theta^t) \\ \leq -\gamma_2 \|\Theta^{t+1} - \Theta^t\|^2 \\ - \gamma_2 [\![t \in \mathcal{T}]\!] \sum_{l=1}^{L} \|(\theta^{l,t+1}, d^{l,t+1}) - (\theta^{l,t}, d^{l,t})\|^2,\end{aligned} \tag{54}$$

that $F_{\text{split}}(\theta^t, d^t, \Theta^t)$ is monotonically decreasing and since $F_{\text{split}} \geq 0$ is bounded from below the energies converge to a finite value $F_{\text{split}}(\theta^t, d^t, \Theta^t) \to F^*_{\text{split}}$. Then summing from $t = 0$ to $t = T$ yields:

$$\begin{aligned}
-\infty < C &\leq F_{\text{split}}(\theta^{T+1}, d^{T+1}, \Theta^{T+1}) - F_{\text{split}}(\theta^0, d^0, \Theta^0) \\
&\leq \sum_{t=0}^{T} -\gamma_2 \|\Theta^{t+1} - \Theta^t\|^2 \\
&\qquad - \gamma_2 [\![t \in \mathcal{T}]\!] \cdot \sum_{l=1}^{L} \|(\theta^{l,t+1}, d^{l,t+1}) \\
&\qquad\qquad - (\theta^{l,t}, d^{l,t})\|^2, \qquad (55)
\end{aligned}$$

showing that $\|\Theta^{t+1} - \Theta^t\| \to 0$ and $\|(\theta^{l,t_k+1}, d^{l,t+1}) - (\theta^{l,t_k}, d^{l,t_k})\| \to 0$ for the subsequence $t_k \in \mathcal{T}$ and via the relative error bounds also $\epsilon_\Theta^{t+1} \to 0$ and $\epsilon_\theta^{l,t_k+1} \to 0$. Furthermore, using the shorthand notation

$$\nabla F_{\text{BA}}^{l,t} := \nabla F_{\text{BA}}^{l}(\theta^{l,t}, d^{l,t}), \qquad (56)$$

$$N_{\mathcal{K}^l}^{t} := N_{\mathcal{K}^l}(\theta^{l,t}, d^{l,t}), \qquad (57)$$

we have in view of [36, Proposition 10.5] and [36, Exercise 8.8(c)]:

$$\begin{aligned}
&\begin{pmatrix}
N_{\mathcal{B}}(\Theta^{t+1}) - \sum_{l=1}^{L} {A^l}^\top W^l(\theta^{l,t+1} - A^l\Theta^{t+1}) \\
\nabla F_{\text{BA}}^{1,t+1} + N_{\mathcal{K}^1}^{t+1} + (W^1(\theta^{1,t+1} - A^1\Theta^{t+1}), \mathbf{0}) \\
\vdots \\
\nabla F_{\text{BA}}^{L,t+1} + N_{\mathcal{K}^L}^{t+1} + (W^L(\theta^{L,t+1} - A^L\Theta^{t+1}), \mathbf{0})
\end{pmatrix} \\
&\qquad = \partial F_{\text{split}}(\theta^{t+1}, d^{t+1}, \Theta^{t+1}) \qquad (58)
\end{aligned}$$

By the individual optimality conditions we have for any $t_k \in \mathcal{T}$:

$$\begin{aligned}
\omega^{t_k} &:= \begin{pmatrix}
\epsilon_\Theta^{t_k+1} \\
\epsilon_\theta^{1,t_k+1} + (W^1 A^1(\Theta^{t_k} - \Theta^{t_k+1}), \mathbf{0}) \\
\vdots \\
\epsilon_\theta^{L,t_k+1} + (W^L A^L(\Theta^{t_k} - \Theta^{t_k+1}), \mathbf{0})
\end{pmatrix} \\
&\in \partial F_{\text{split}}(\theta^{t_k+1}, d^{t_k+1}, \Theta^{t_k+1}). \qquad (59)
\end{aligned}$$

Now consider a convergent sub-sequence $(\theta^{t_{k_j}+1}, d^{t_{k_j}+1}, \Theta^{t_{k_j}+1}) \to (\theta^*, d^*, \Theta^*)$ of the iterates $(\theta^{t_k+1}, d^{t_k+1}, \Theta^{t_k+1})$ indexed by j. Then we have in view of the inclusion above that $\omega^{t_{k_j}} \to 0$ with

$$\omega^{t_{k_j}} \in \partial F_{\text{split}}(\theta^{t_{k_j}+1}, d^{t_{k_j}+1}, \Theta^{t_{k_j}+1}), \qquad (60)$$

and

$$F_{\text{split}}(\theta^{t_{k_j}+1}, d^{t_{k_j}+1}, \Theta^{t_{k_j}+1}) \to F^*_{\text{split}}, \qquad (61)$$

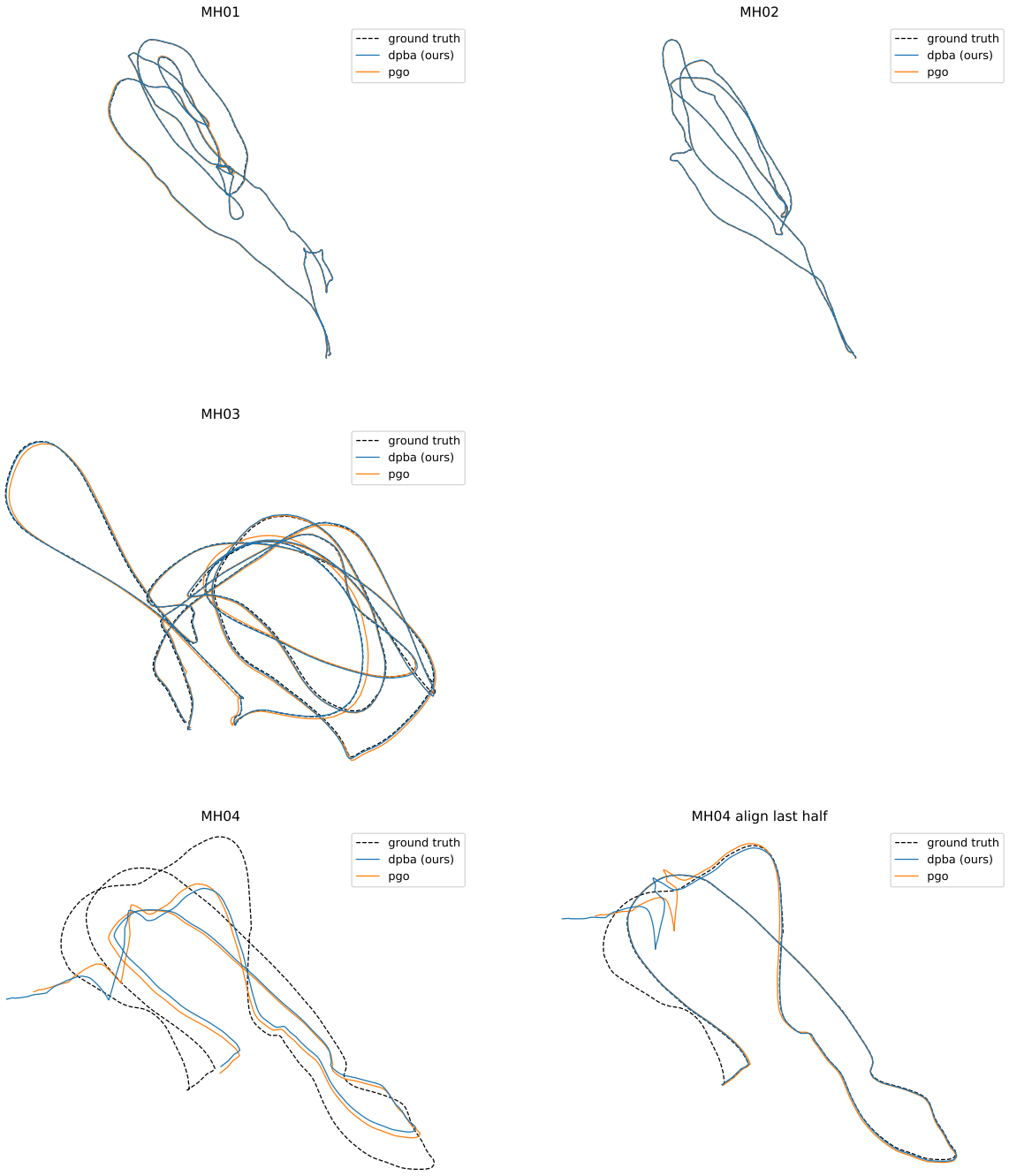
$$F^*_{\text{split}} = F_{\text{split}}(\theta^*, d^*, \Theta^*). \qquad (62)$$

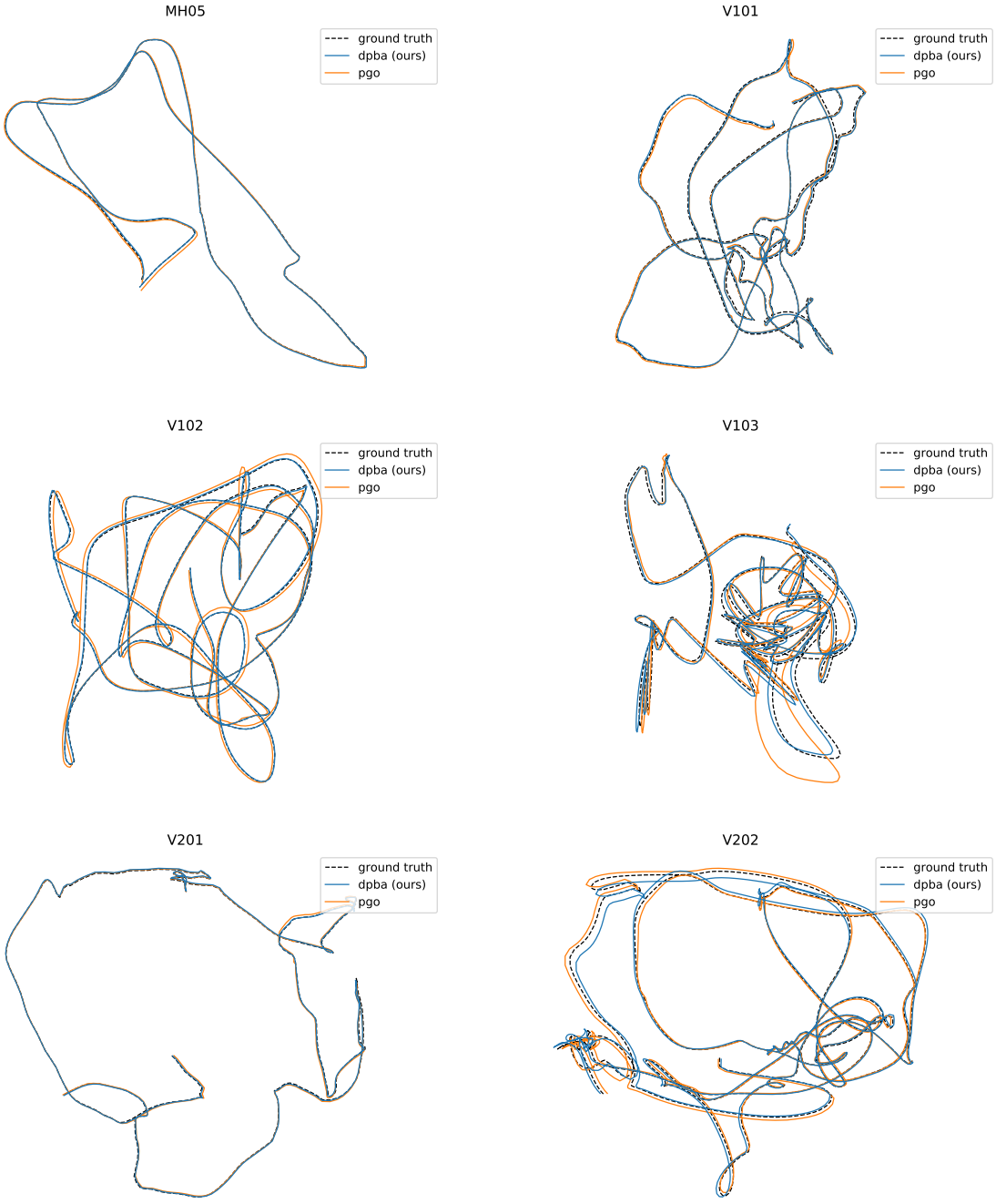
The equality in (62) holds, since the iterates are feasible and the domain $\operatorname{dom} F_{\text{split}}$ is closed and F_{split} is continuous on its domain. Since the graph of the subdifferential, see [36, Definition 8.3(b)], $\operatorname{gph} \partial F_{\text{split}} = \{(x, y) : y \in \partial F_{\text{split}}(x)\}$ is a closed set under the F_{split}-attentive topology and $((\theta^{t_{k_j}+1}, d^{t_{k_j}+1}, \Theta^{t_{k_j}+1}), \omega^{t_{k_j}}) \in \operatorname{gph} \partial F_{\text{split}}$, we obtain by passing to the limit $((\theta^*, d^*, \Theta^*), 0) \in \operatorname{gph} \partial F_{\text{split}}$, i.e., $0 \in \partial F_{\text{split}}(\theta^*, d^*, \Theta^*)$. □

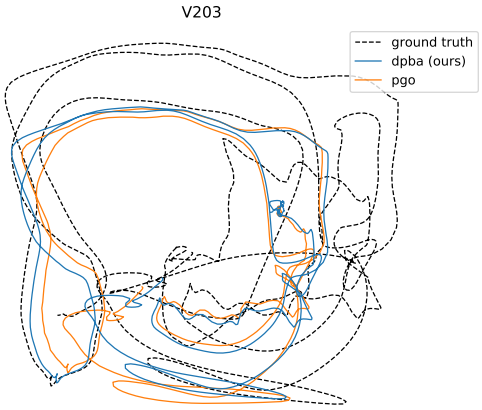
References

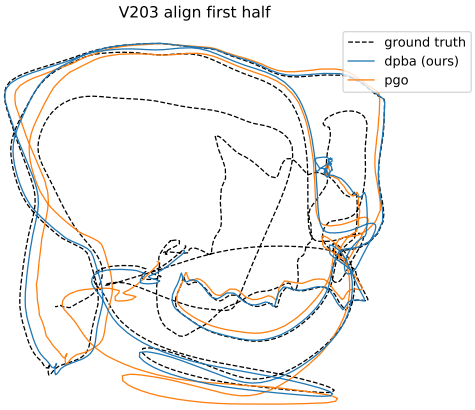
[36] R. T. Rockafellar and R. J. B. Wets. *Variational Analysis*. Springer, Berlin, 1998.

Table 10. Estimated camera trajectories before (*pgo*) and after (*dpba*) our global distributed photometric bundle adjustment for all *EuRoC* sequences, after $\mathrm{Sim}(3)$ alignment to the ground-truth poses.




MH05
ground truth
dpba (ours)
pgo
V101
ground truth
dpba (ours)
pgo
V102
ground truth
dpba (ours)
pgo
V103
ground truth
dpba (ours)
pgo
V201
ground truth
dpba (ours)
pgo
V202
ground truth
dpba (ours)
pgo


V203
ground truth
dpba (ours)
pgo


V203 align first half
ground truth
dpba (ours)
pgo