DOI: 10.1109/LRA.2021.3140129

DM-VIO: Delayed Marginalization Visual-Inertial Odometry

Lukas von Stumberg[1] and Daniel Cremers[1]

***Abstract*— We present DM-VIO, a monocular visual-inertial odometry system based on two novel techniques called delayed marginalization and pose graph bundle adjustment. DM-VIO performs photometric bundle adjustment with a dynamic weight for visual residuals. We adopt marginalization, which is a popular strategy to keep the update time constrained, but it cannot easily be reversed, and linearization points of connected variables have to be fixed. To overcome this we propose delayed marginalization: The idea is to maintain a second factor graph, where marginalization is delayed. This allows us to later readvance this delayed graph, yielding an updated marginalization prior with new and consistent linearization points. In addition, delayed marginalization enables us to inject IMU information into already marginalized states. This is the foundation of the proposed pose graph bundle adjustment, which we use for IMU initialization. In contrast to prior works on IMU initialization, it is able to capture the full photometric uncertainty, improving the scale estimation. In order to cope with initially unobservable scale, we continue to optimize scale and gravity direction in the main system after IMU initialization is complete. We evaluate our system on the EuRoC, TUM-VI, and 4Seasons datasets, which comprise flying drone, large-scale handheld, and automotive scenarios. Thanks to the proposed IMU initialization, our system exceeds the state of the art in visual-inertial odometry, even outperforming stereo-inertial methods while using only a single camera and IMU. The code will be published at `vision.in.tum.de/dm-vio`**

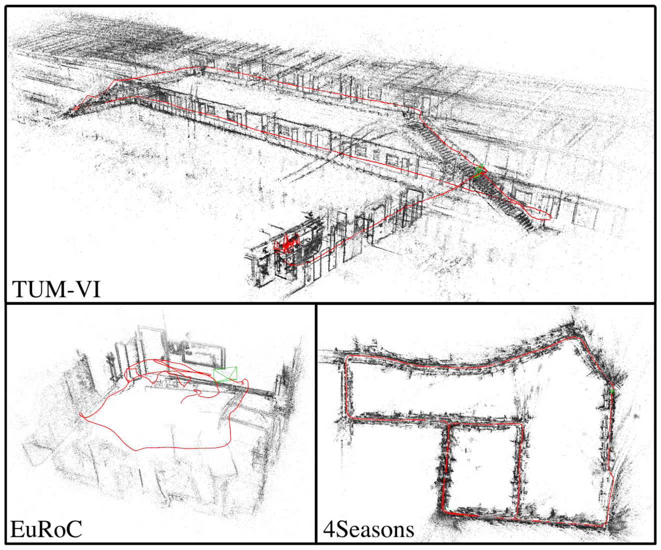


Fig. 1: In this paper, we propose a novel method for monocular visual-inertial odometry. It provides state-of-the-art performance on three different benchmarks. Here we show pointclouds and trajectories (red) for magistrale5, V203_difficult, and neighbor_2020-03-26_13-32-55_0.

I. Introduction

Visual-(inertial) odometry is an increasingly relevant task with applications in robotics, autonomous driving, and augmented reality. A combination of cameras and inertial measurement units (IMUs) for this task is a popular and sensible choice, as they are complementary sensors, resulting in a highly accurate and robust system [1]. In the minimal configuration of a single camera, the IMU can also be used to recover the metric scale. However, the scale is not always observable, the most common degenerate case being movement with a constant velocity [2]. Hence, initialization of such system can take arbitrarily long, depending on the trajectory. Even worse, when initialized prematurely the IMU can in fact worsen the performance. The difficulty of IMU initialization is why stereo-inertial methods have outperformed mono-inertial ones in the past.

Most prior systems [3] [4] [5] initially run visual-only odometry and an IMU initialization in parallel. Once finished, the visual-inertial system is started. This introduces a trade-off for the duration of the initialization period: It should be as short as possible, as no IMU information is used in the main system in the meantime. But when too short, the scale estimate will be inaccurate, leading to bad performance.

VI-DSO [6] instead initializes immediately with an arbitrary scale, and explicitly optimizes the scale in the main system. This yields highly accurate scale estimates, but it can significantly increase the time until the scale is correctly estimated. Also, it can fail in cases where the initial scale error is very high, like in large-scale outdoor environments.

We propose a combination of the two strategies: Similar to the former, we start with a visual-only system and run an IMU initializer in parallel. But after IMU initialization we still estimate scale and gravity direction as explicit optimization variables in the main system. This results in a quickly converging and highly accurate system.

This initialization strategy can lead to three questions: 1) How can the visual uncertainty be properly captured in the IMU initializer. 2) How can information about scale and IMU variables be transferred from the IMU initializer to the main system? 3) If the scale estimate changes, how can a consistent marginalization prior be maintained? VI-DSO [6] tried to address 3 by introducing dynamic marginalization, which does keep the marginalization factor consistent, but loses too much information in the process.

In this work we propose delayed marginalization, which provides a meaningful answer to all three of these questions.

[1]All authors are with the Computer Vision Group, Technical University of Munich, Germany. `lukas.stumberg@tum.de`, `cremers@in.tum.de`

The idea is to maintain a second, *delayed* marginalization prior, which has very little overhead, but enables three core techniques:

1) We can populate the delayed factor graph with new IMU factors to perform the proposed pose graph bundle adjustment (PGBA). This is the basis of an IMU initialization which captures the full photometric uncertainty, leading to increased accuracy.
2) The graph used for IMU initialization can be *re-advanced*, providing a marginalization prior with IMU information for the main system.
3) When the scale changes significantly in the main system we can trigger *marginalization replacement*.

The combination of these techniques makes for a highly accurate initializer, which is robust even to long periods of unobservability. Based on it we implement a visual-inertial odometry (VIO) system featuring a photometric front-end integrated with a new dynamic photometric weight.

We evaluate our method on three challenging datasets (Fig. 1), capturing three domains: The EuRoC dataset [7] recorded by a flying drone, the TUM-VI dataset [8] captured with a handheld device, and the 4Seasons dataset [9] representing the automotive scenario. The latter features long stretches of constant velocity, posing a particular challenge for mono-inertial odometry.

We show that our system exceeds the state of the art in visual-inertial odometry, even outperforming stereo-inertial methods. In summary our contributions are:

- Delayed marginalization compensates drawbacks of marginalization while retaining the advantages.
- Pose graph bundle adjustment (PGBA) combines the efficiency of pose graph optimization with the full uncertainty of bundle adjustment.
- A state-of-the-art visual-inertial odometry system with a novel multi-stage IMU initializer and dynamically weighted photometric factors.

The full source code for our approach will be released.

II. Related Work

Initially, most visual odometry and SLAM systems have been feature-based [10], either using filtering [11] or non-linear optimization [12] [13]. More recently, direct methods have been proposed, which optimize a photometric error function and can operate on dense [14] [15], semi-dense [16], or sparse point clouds [17].

Mourikis and Roumeliotis [1] have shown that a tight integration of visual and inertial measurements can greatly increase accuracy and robustness of odometry. Afterwards, many tightly-coupled visual-inertial odometry [18] [19] and SLAM systems [20] [21] [3] [5] have been proposed.

Initialization of monocular visual-inertial systems is not trivial, as sufficient motion is necessary for the scale to become observable [22] [2]. Most systems [4] [3] [5] start with a visual-only system and use its output for a separate IMU initialization. In contrast to these systems, we continue optimizing the scale explicitly in the main system. We note that ORB-SLAM3 [5] also continues to refine the scale after initialization, but this is a separate optimization fixing all poses and only performed until 75 seconds after initialization. [23] also continues to optimize the scale in the main system, but in contrast to us they do not transfer covariances between the main system and the initializer, thus they do not achieve the same level of accuracy. Different to all these systems, the proposed delayed marginalization allows our IMU initializer to capture the full visual uncertainty and continuously optimize the scale in the main system.

VI-DSO [6] initializes immediately with an arbitrary scale and explicitly optimizes the scale in the main system. It also introduced dynamic marginalization to handle the consequential large scale changes in the main system. Compared to it we propose a separate IMU initializer, delayed marginalization as a better alternative to dynamic marginalization, a dynamic photometric error weight, and more improvements, resulting in greatly improved accuracy and robustness.

III. Method

A. Notation

We denote vectors as bold lowercase letters $\mathbf{x}$, matrices as bold upper-case letter $\mathbf{H}$, scalars as lowercase letters λ, and functions as uppercase letters E. $\mathbf{T}^{V}_{\text{w\_cam}_i} \in \mathbf{SE}(3)$ represents the transformation from camera i to world in the visual coordinate frame V, and $\mathbf{R}^{V}_{\text{w\_cam}_i} \in \mathbf{SO}(3)$ is the respective rotation. Poses are represented either in visual frame $\mathbf{P}^{V}_{i} := \mathbf{T}^{V}_{\text{cam}_i\text{\_w}}$, or in inertial frame $\mathbf{P}^{I}_{i} := \mathbf{T}^{I}_{\text{w\_imu}_i}$. If not mentioned otherwise we use poses in visual frame $\mathbf{P}_i := \mathbf{P}^{V}_{i}$. We also use states $\mathbf{s}$, which can contain transformations, rotations, and vectors. For states we define the subtraction operator $\mathbf{s}_i \boxminus \mathbf{s}_j$, which applies $\log(\mathbf{R}_i \mathbf{R}_j^{-1})$ for rotations and other Lie group elements, and a regular subtraction for vector values.

B. Direct Visual-Inertial Bundle Adjustment

The core of DM-VIO is the visual-inertial bundle adjustment performed for all keyframes. As commonly done, we jointly optimize visual and IMU variables in a combined energy function. For the visual part we choose a direct formulation based on DSO [17], as it is a very accurate and robust system. For integrating IMU data into the bundle adjustment we perform preintegration [24] between keyframes.

We optimize the following energy function using the Levenberg-Marquardt algorithm:

$$E(\mathbf{s}) = W(e_{\text{photo}}) \cdot E_{\text{photo}} + E_{\text{imu}} + E_{\text{prior}} \tag{1}$$

E_{prior} contains added priors on the first pose and the gravity direction, as well as the marginalization priors explained in section III-C. In the following we describe the individual energy terms and the optimized state.

Photometric error: The photometric energy is based on [17]. We optimize a set of active keyframes $\mathcal{F}$, each of which hosts a set of points $\mathcal{P}_i$. Every point $\mathbf{p}$ is projected into all keyframes $\text{obs}(\mathbf{p})$ where it is visible, and the photometric energy is computed:

$$E_{\text{photo}} = \sum_{i \in \mathcal{F}} \sum_{\mathbf{p} \in \mathcal{P}_i} \sum_{j \in \text{obs}(\mathbf{p})} E_{\mathbf{p}j} \tag{2}$$

$$E_{\mathbf{p}j} = \sum_{\mathbf{p}\in\mathcal{N}_\mathbf{p}} \omega_\mathbf{p} \left\| (I_j[\mathbf{p}'] - b_j) - \frac{t_j e^{a_j}}{t_i e^{a_i}} (I_i[\mathbf{p}] - b_i) \right\|_\gamma \quad (3)$$

For details regarding the variables we refer the reader to [17].

Dynamic photometric weight: In cases of bad image quality, the system should rely mostly on the inertial data. However due to the photometric cost function used, bad image quality will often lead to very large photometric residuals, effectively increasing the photometric weight compared to the IMU. To counteract this we propose a dynamic photometric weight $W(e_{\text{photo}})$. We compute it using the root mean squared photometric error $e_{\text{photo}} = \sqrt{E_{\text{photo}}/n_{\text{residuals}}}$.

$$W(e_{\text{photo}}) = \lambda \cdot \begin{cases} (\theta/e_{\text{photo}})^2, & \text{if } e_{\text{photo}} \geq \theta \\ 1, & \text{otherwise} \end{cases} \quad (4)$$

where λ is a static weight component, and θ is the threshold from which the error-dependent weight is activated. This effectively normalizes the root mean squared photometric error to be $\sqrt{\lambda}\theta$ at maximum, similar to a threshold robust cost function [25]. In contrast to the Huber norm in Equation (3), which downweights individual points that violate the photometric assumption, this weight addresses cases where the overall image quality is bad and increases the relative weight of the IMU. In our experiments we choose $\theta = 8$.

Optimized variables: We optimize scale and gravity direction as explicit variables. While bundle adjustment can in principle also change the scale and global orientation, convergence is improved when optimizing them explicitly instead [6]. To facilitate this, we represent poses for the visual factors in visual frame V and poses for the IMU factors in IMU frame I. Whereas the IMU frame has a metric scale and a z-axis aligned with gravity direction, the visual frame can have an arbitrary scale and rotation, which is defined during initialization of the visual system. To model this we optimize the scale s and the rotation $\mathbf{R}_{V\_I}$. As yaw is not observable using an IMU, we fix the last coordinate of $\mathbf{R}_{V\_I}$. We convert between the coordinate frames using:

$$\begin{aligned} \mathbf{P}_i^I := \mathbf{T}^I_{\text{w\_imu}_i} = \Omega(\mathbf{P}_i^V, \mathbf{S}, \mathbf{R}_{V\_I}) = \\ \mathbf{R}_{V\_I}^{-1}\mathbf{S}_{I\_V}(\mathbf{P}_i^V)^{-1}\mathbf{S}_{I\_V}^{-1}\mathbf{T}_{\text{cam\_imu}} \end{aligned} \quad (5)$$

where $\mathbf{S}_{I\_V}$ is the $\mathbf{Sim}(3)$ element with identity rotation and translation, and scale s. The other variables are converted to $\mathbf{Sim}(3)$, but note that the result has scale 1 and is in $\mathbf{SE}(3)$.

The full state optimized is

$$\mathbf{s} = \{s, \mathbf{R}_{V\_I}\} \cup \bigcup_{i\in\mathcal{F}} \mathbf{s}_i \quad (6)$$

with $\mathbf{s}_i$ being the states for all active keyframes defined as:

$$\mathbf{s}_i = \{\mathbf{P}_i^V, \mathbf{v}_i, \mathbf{b}_i, a_i, b_i, d_i^0, d_i^2, ...d_i^j\} \quad (7)$$

where $\mathbf{v}_i$ is the velocity, $\mathbf{b}_i$ the bias, a_i and b_i are affine brightness parameters, and d_i^j are the inverse depths of active points hosted in the keyframe. Optimization is performed with a custom integration of the SIMD-accelerated code from [17] for photometric residuals and GTSAM for other factors.

IMU Error: We apply the well-known IMU preintegration first proposed in [26], implemented as smart factors in [27], and further improved in [24]. For this energy we use the IMU state $\mathbf{s}_i^I := \{\mathbf{P}_i^I, \mathbf{v}_i, \mathbf{b}_i\}$, which contains poses in IMU frame and is computed from the optimized state $\mathbf{s}_i$ using Equation (5). Given the previous state $\mathbf{s}_i^I$, the preintegration data provides us with a prediction $\widehat{\mathbf{s}}_j^I$ for the following state $\mathbf{s}_j^I$ as well as a covariance matrix $\widehat{\mathbf{\Sigma}}_j$. The resulting inertial error function penalizes deviations of the current state estimate from the predicted state.

$$E_{\text{imu}}(\mathbf{s}_i^I, \mathbf{s}_j^I) := \left(\widehat{\mathbf{s}}_j^I \boxminus \mathbf{s}_j^I\right)^T \widehat{\mathbf{\Sigma}}_j^{-1} \left(\widehat{\mathbf{s}}_j^I \boxminus \mathbf{s}_j^I\right) \quad (8)$$

C. Partial Marginalization using the Schur Complement

We marginalize old variables using the Schur complement. When marginalizing a set β of variables, we gather all factors dependent on them as well as the connected variables α, which form the Markov blanket. These factors are linearized at the current state estimate, yielding the linear system:

$$\begin{bmatrix} \mathbf{H}_{\alpha\alpha} & \mathbf{H}_{\alpha\beta} \\ \mathbf{H}_{\beta\alpha} & \mathbf{H}_{\beta\beta} \end{bmatrix} \begin{bmatrix} \mathbf{s}_\alpha \\ \mathbf{s}_\beta \end{bmatrix} = \begin{bmatrix} \mathbf{b}_\alpha \\ \mathbf{b}_\beta \end{bmatrix} \quad (9)$$

We apply the Schur-complement, which results in the new linear system $\widehat{\mathbf{H}_{\alpha\alpha}} \boldsymbol{s}_\alpha = \widehat{\boldsymbol{b}_\alpha}$ with

$$\widehat{\mathbf{H}_{\alpha\alpha}} = \mathbf{H}_{\alpha\alpha} - \mathbf{H}_{\alpha\beta}\mathbf{H}_{\beta\beta}^{-1}\mathbf{H}_{\beta\alpha} \quad (10)$$

$$\widehat{\boldsymbol{b}_\alpha} = \boldsymbol{b}_\alpha - \mathbf{H}_{\alpha\beta}\mathbf{H}_{\beta\beta}^{-1}\boldsymbol{b}_\beta \quad (11)$$

This linear system forms a marginalization prior connecting all variables in α (Fig. 2a).

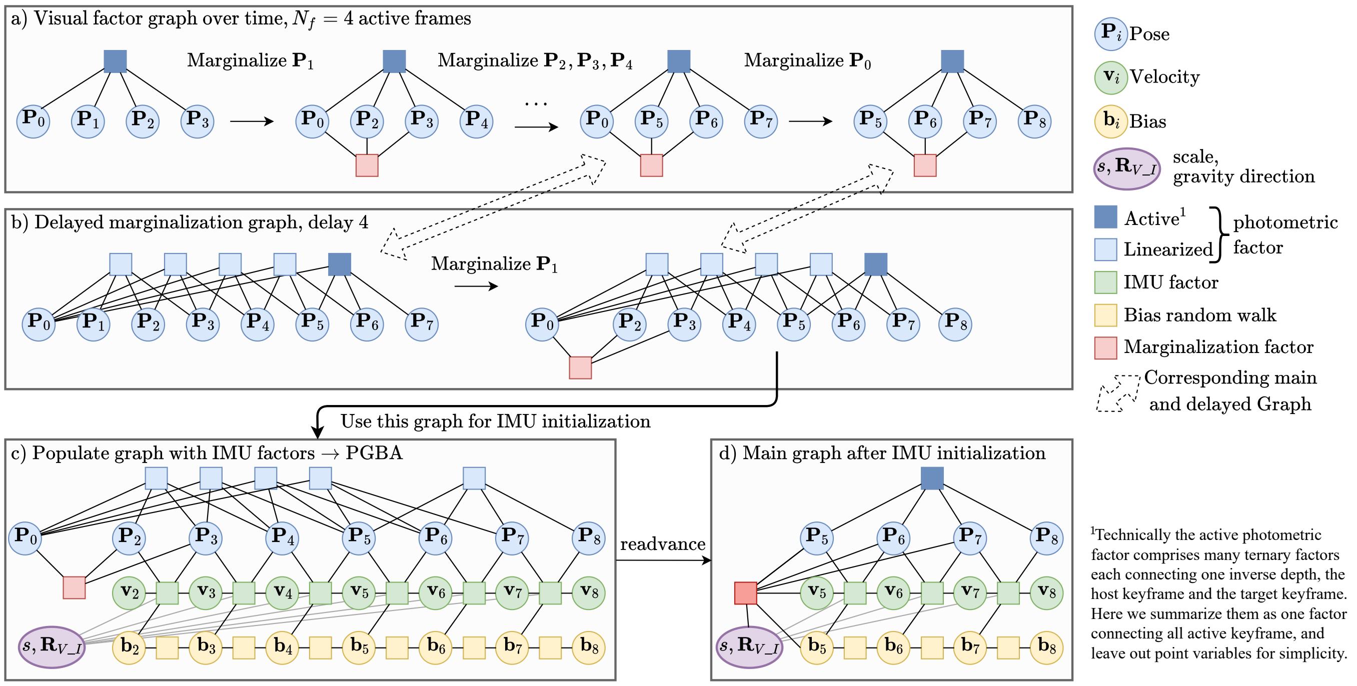
We keep a maximum of $N_f = 8$ keyframes during the bundle adjustment[1]. The marginalization strategy is taken over from [17]: This means that different from a fixed-lag smoother, we do not always marginalize the oldest pose, but instead keep a combination of newer and older poses, as long as they do not leave the field of view. As shown in [17] this is superior to a fixed-lag smoother for visual odometry. When marginalizing a pose, first all remaining points hosted in the frame are marginalized and residuals with remaining active points are dropped. This retains sparsity of the Hessian while preserving enough information.

D. Delayed Marginalization

The concept of marginalization explained in the previous section has the advantage of capturing the full probability distribution. In fact, solving the resulting smaller system is equivalent to solving the much larger original system, as long as the marginalized factors are not relinearized.

However, it also comes with severe drawbacks: Reverting the marginalization of a set of variables is not possible without redoing the whole marginalization procedure. Also, to keep the marginalization prior consistent, First-Estimates Jacobians (FEJ) [28] have to be applied. This means that the linearization point of all connected variables has to be fixed as soon as they are connected to a marginalization prior. This is especially problematic for visual-inertial odometry, where the scale is connected to the marginalization prior as

[1]We define N_f as the maximum number of frames during bundle adjustment, whereas in [17] it is the number of frames after marginalization.



a) Visual factor graph over time, $N_f = 4$ active frames
Marginalize $\mathbf{P}_1$
Marginalize $\mathbf{P}_2, \mathbf{P}_3, \mathbf{P}_4$
Marginalize $\mathbf{P}_0$
b) Delayed marginalization graph, delay 4
Marginalize $\mathbf{P}_1$
Use this graph for IMU initialization
c) Populate graph with IMU factors → PGBA
readvance
d) Main graph after IMU initialization
$\mathbf{P}_i$ Pose
$\mathbf{v}_i$ Velocity
$\mathbf{b}_i$ Bias
$s, \mathbf{R}_{V\_I}$ scale, gravity direction
Active[1]
Linearized
photometric factor
IMU factor
Bias random walk
Marginalization factor
Corresponding main and delayed Graph
[1]Technically the active photometric factor comprises many ternary factors each connecting one inverse depth, the host keyframe and the target keyframe. Here we summarize them as one factor connecting all active keyframe, and leave out point variables for simplicity.

Fig. 2: Delayed Marginalization and PGBA: a) Normal marginalization in the visual graph. Note that not always the oldest pose is marginalized. b) Delayed marginalization: We marginalize all variables in the same order as the main graph, but with a delay d (in practice $d = 100$). Marginalization in this graph is equally fast as marginalizing in the main graph. c) For the pose graph bundle adjustment (PGBA) we populate the delayed graph with IMU factors. This optimization leverages the full photometric uncertainty. d) We readvance the marginalization in the graph used for PGBA to obtain an updated marginalization prior for the main system. This transfers inertial information from the initializer to the main system.

soon as the first keyframe is marginalized, but might change significantly. In [6] dynamic marginalization was introduced to combat this, but it is limited in its application to a single one-dimensional variable, namely the scale, and loses most prior inertial information when the scale changes quickly.

Here we introduce delayed marginalization which circumvents the drawbacks of marginalization while retaining the advantages. It enables us to:

- Effectively undo part of the marginalization to capture the full photometric probability distribution for the pose graph bundle adjustment (section III-E).
- Update the initially visual-only marginalization prior with IMU information after the IMU initialization.
- Relinearize variables in the Markov blanket while keeping all visual and most inertial information.

The idea of delayed marginalization is that marginalization cannot be undone, but it can be delayed: *In addition* to the normal marginalization prior we also maintain a second, *delayed* marginalization prior and corresponding factor graph. In this delayed graph, marginalization of frames is performed with a delay of d. Points are still marginalized at the same time in the delayed graph, resulting in linearized photometric factors. We note that the same marginalization order as in the original graph is preserved. Switching to a fixed-lag smoother for this graph would immediately lead to a much larger Markov blanket jeopardizing the runtime of the system. E.g. in Fig. 2b we depict the delayed marginalization of $\mathbf{P}_1$. The Markov blanket only contains $\mathbf{P}_0$, $\mathbf{P}_2$, and $\mathbf{P}_3$. If we instead marginalized the oldest frame $\mathbf{P}_0$, the Markov blanket would contain $\mathbf{P}_1 - \mathbf{P}_7$, leading to higher runtime. **Marginalization in the delayed graph has the same runtime as marginalization in the original graph.** The delayed graph contains the same photometric factors as the original graph, and points are marginalized at the same time. This means that each linearized photometric factor in the delayed graph is connected to exactly the $N_f = 8$ keyframes which were active when the respective factor was generated. By keeping the marginalization order, the Markov blanket in the delayed graph always has the same size as the one in the original graph. Thus, the runtime of the Schur complement is the same. This means that the overhead of Delayed Marginalization is very small even for arbitrarily large delays, as it only amounts to an additional marginalization procedure per delayed graph.

E. Pose Graph Bundle Adjustment for IMU Initialization

PGBA utilizes delayed marginalization for IMU initialization. The idea is to populate the delayed graph with IMU factors and optimize all variables (Fig. 2c).

Populating the graph: Let a frame $\mathbf{P}_i$ be directly connected to the newest pose $\mathbf{P}_k$ iff all poses $\mathbf{P}_j, i < j < k$ have not been marginalized yet. We determine the first frame $\mathbf{P}_{\text{conn}}$ in the delayed graph which is still directly connected to the newest frame. In Fig. 2c this is $\mathbf{P}_2$. From there, we insert IMU factors and bias factors to all successive frames.

We cannot start before $\mathbf{P}_{\text{conn}}$ because we do not want to insert IMU factors between non-successive keyframes. As the marginalization order is not fixed-lag, this means that we have to optimize poses without corresponding IMU variables.

It can be shown that there can be at most $N_f - 2$ poses without IMU variables. The reason is that all non-connected poses were at some point active at the same time. This means that in practice we have at least $d - N_f + 2$ poses for which we can add IMU data. In practice, we choose $N_f = 8$ and delay $d = 100$, meaning that even in the worst case there will be 93 IMU factors in the optimization. As explained previously, fixed-lag smoothing would either result in a dense Hessian or in suboptimal performance of the visual system, so this is a very good trade-off.

Optimization: We optimize the graph with the GTSAM [29] library using the Levenberg-Marquardt optimizer with the provided Ceres-default settings. In this optimization all points are marginalized. We call it pose graph bundle adjustment because it is a combination of regular pose graph optimization (PGO) and bundle adjustment (BA). In contrast to BA, we do not update our estimates for point depths and do not relinearize photometric error terms. Different from PGO we do not use binary constraints between poses, but instead use ”octonary” constraints, which connect N_f frames and capture the full probability distribution of BA. Compared to PGO our solution is thus more accurate while being much faster than full BA. By using a fixed delay it is also constrained in runtime even though it can be performed at any time without losing any prior visual information.

Readvancing: Another advantage of delayed marginalization and our PGBA is that we can obtain a marginalization prior for the main system, capturing all the visual and inertial information. For this, we readvance the graph used for the PGBA. This works by successively marginalizing all the variables which have been marginalized in the main graph. Again, this is done preserving the marginalization order, which means that in each marginalization step the Markov blanket has a fixed maximum size. Hence, marginalizing step by step is significantly faster than marginalizing all variables at once, which would involve a much larger matrix inversion. Fig. 2d shows the result of readvancing.

F. Robust Multi-Stage IMU Initialization

Our initialization strategy is based on three insights:

1) When some variables are unknown (in our case scale, gravity direction, and biases) and others are close to the optimum, it is most efficient to first optimize only the unknown variables and fix the others.
2) The most accurate result can be obtained by optimizing all variables jointly, capturing the full covariance.
3) When marginalizing, connected variables have to be close to the optimum, otherwise the marginalization prior becomes inconsistent.

These observations inspire 1) the Coarse IMU Initialization, 2) the PGBA, and 3) the Marginalization Replacement (Fig. 3). Note that after ”Initialize main VIO”, the main VIO system III-B (green box) is already running in parallel.

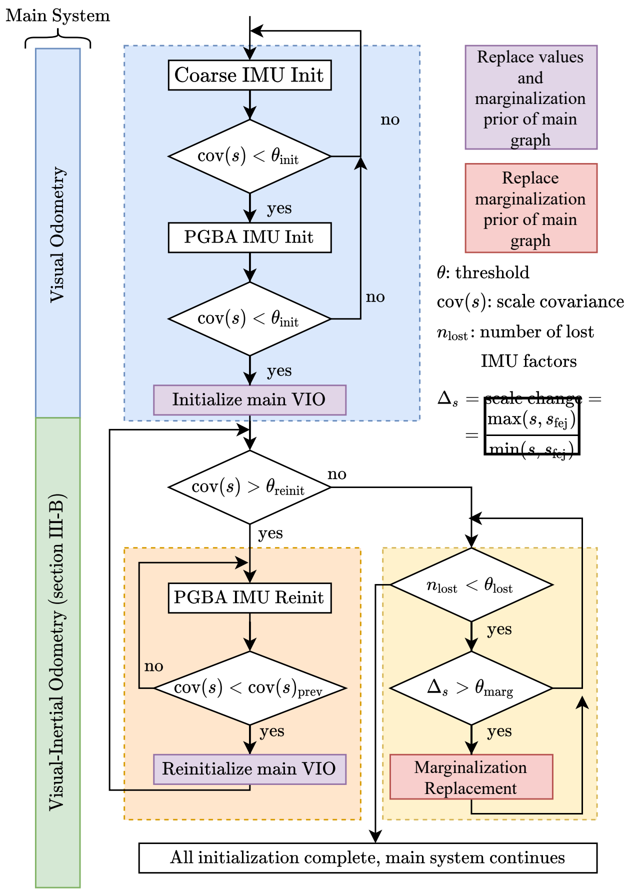

Main System
Visual Odometry
Visual-Inertial Odometry (section III-B)
Coarse IMU Init
$\text{cov}(s) < \theta_{\text{init}}$
no
yes
PGBA IMU Init
$\text{cov}(s) < \theta_{\text{init}}$
no
yes
Initialize main VIO
$\text{cov}(s) > \theta_{\text{reinit}}$
no
yes
PGBA IMU Reinit
$\text{cov}(s) < \text{cov}(s)_{\text{prev}}$
no
yes
Reinitialize main VIO
$n_{\text{lost}} < \theta_{\text{lost}}$
yes
$\Delta_s > \theta_{\text{marg}}$
yes
Marginalization Replacement
All initialization complete, main system continues
Replace values and marginalization prior of main graph
Replace marginalization prior of main graph
θ: threshold
$\text{cov}(s)$: scale covariance
n_{lost}: number of lost IMU factors
$\Delta_s = \text{~~scale change~~} = \frac{\max(s, s_{\text{fej}})}{\min(s, s_{\text{fej}})}$

Fig. 3: Our multi-stage IMU initialization. First we perform a coarse IMU initialization, which provides initial values for the PGBA. The PGBA captures the full visual covariances, achieving very accurate initial estimates for scale, gravity direction, and biases. It also provides an updated marginalization prior for the main graph. By also optimizing the scale in the main VIO system (green box), we can initialize early (purple box) and later reinitialize or perform marginalization replacement, if new information about the scale becomes available. The proposed delayed marginalization is what enables both, the PGBA, and the marginalization replacement.

For this initializer we use a single delayed graph with a delay of $d = 100$. This delayed graph will always contain only visual factors and no IMU factors, even after the first initialization, to facilitate the marginalization replacement.

Coarse IMU Initialization: For this we only consider the last $d = 100$ keyframes and connect them with IMU factors. Similar to the inertial only optimization used for initialization in ORB-SLAM3 [5], in this optimization we fix the poses and use a single bias. We only optimize velocities, bias, the gravity direction and the scale. Gravity direction is initialized by averaging the accelerometer measurements between the first two keyframes, scale is initialized with 1, and bias and velocity with 0. This optimization is less accurate than PGBA but serves as an initialization for it. After optimizing, we compute the marginal covariance for the scale $\text{cov}(s)$ and continue to the PGBA if it is smaller

TABLE I: Evaluation of various mono (M) and stereo (S) visual-inertial odometry systems on EuRoC. Our system provides a notable improvement over the state-of-the art. Please note that a full SLAM system utilizing loop closures can achieve even more accurate results, e.g. ORB-SLAM-VI has a mean error of 0.075, and ORB-SLAM3 has a mean error of 0.043.

Sequence		MH1	MH2	MH3	MH4	MH5	V11	V12	V13	V21	V22	V23	Avg
MCSKF^2 [1] (M)	RMSE	0.42	0.45	0.23	0.37	0.48	0.34	0.20	0.67	0.10	0.16	1.13	0.414
OKVIS^1 [19] (M)	RMSE	0.33	0.37	0.25	0.27	0.39	0.094	0.14	0.21	0.090	0.17	0.23	0.231
ROVIO^2 [18] (M)	RMSE	0.21	0.25	0.25	0.49	0.52	0.10	0.10	0.14	0.12	0.14	0.14	0.224
VINS-Mono [3] (M)	RMSE	0.15	0.15	0.22	0.32	0.30	0.079	0.11	0.18	0.080	0.16	0.27	0.184
Kimera [21] (S)	RMSE	0.11	0.10	0.16	0.24	0.35	0.05	0.08	0.07	0.08	0.10	0.21	0.141
Online VIO [23] (M)	RMSE	0.14	0.13	0.20	0.22	0.20	0.05	0.07	0.16	0.04	0.11	0.17	0.135
VI-DSO [6] (M)	RMSE	**0.062**	**0.044**	0.117	0.132	0.121	0.059	0.067	0.096	0.040	0.062	0.174	0.089
	Scale Error (%)	1.1	0.5	0.4	0.2	0.8	1.1	1.1	0.8	1.2	0.3	0.4	0.7
BASALT [20] (S)	RMSE	0.07	0.06	**0.07**	0.13	0.11	**0.04**	0.05	0.10	0.04	**0.05**	-	0.072
DM-VIO (M)	RMSE	0.065	**0.044**	0.097	**0.102**	**0.096**	0.048	**0.045**	**0.069**	**0.029**	**0.050**	**0.114**	**0.069**
	Scale Error (%)	1.3	0.9	0.4	0.2	0.4	0.4	1.0	0.3	0.02	0.6	0.8	0.6

^{1} results taken from [3].
^{2} results taken from [30], these are $\mathbf{Sim}(3)$-aligned.
All other results are taken from the respective paper.

than a threshold θ_{init}. As shown in [31], taking into account IMU noise parameters is crucial for good IMU initialization, which our coarse IMU initialization satisfies. But for our method it is just an initialization for the PGBA, which in addition models photometric noise properties.

PGBA IMU Init.: We perform PGBA as explained in section III-E. Afterwards, we again threshold on the marginal covariance for the scale to find out if the optimization was successful. When a tighter threshold θ_{reinit} is not also met, we initialize with the result, but will perform another PGBA afterwards to reinitialize with more accurate values. This reinitialization enables us to set θ_{init} to a relatively large value, allowing to use IMU data in the main system earlier.

Marginalization Replacement: After IMU initialization, we monitor how much the scale s changes compared to the First-Estimates scale s_{fej} used in the marginalization prior. If this change exceeds a threshold θ_{marg}, i.e. $\delta_s := \max(s, s_{\text{fej}})/\min(s, s_{\text{fej}}) > \theta_s$, we trigger a marginalization replacement. For the marginalization replacement we rebuild the PGBA graph by populating the delayed graph with IMU factors, Fig. 2c). Different from the PGBA, we do not optimize in this graph but instead just readvance it to obtain an updated marginalization prior. This new prior still contains all visual factors and at least the last $d - N_f + 1 = 93$ IMU factors. We disable the marginalization replacement if more than $\theta_{\text{lost}} = 50\%$ of the IMU factors contained in the previous prior would be lost. This procedure shows how delayed marginalization can be used to update FEJ values, overcoming one of the main problems of marginalization.

In realtime mode we perform the coarse IMU initialization and the PGBA in a separate thread. Note how important the proposed delayed marginalization is for this IMU initialization. It allows the PGBA to capture the full covariance from the photometric bundle adjustment. By readvancing, this also enables us to generate a marginalization prior for the main system, containing all IMU information from the initializer. Lastly, it is used for updating the marginalization prior when the scale changes after the initialization.

IV. Results

We evaluate our method on the EuRoC dataset [7], the TUM-VI dataset [8], and the 4Seasons dataset [9], covering flying drones, handheld sequences, and autonomous driving respectively. We encourage the reader to watch the supplementary video which shows qualitative realtime results on 4Seasons and TUM-VI slides1. We also provide ablation studies and runtime evaluations in the supplementary available at `vision.in.tum.de/dm-vio`.

Unless otherwise stated all experiments are performed in realtime mode on the same MacBook Pro 2013 (i7 at 2.3GHz) which was used for generating the results in [6], without utilizing the GPU. As ORB-SLAM3 is not officially supported on MacOS, we show results for it on a slightly stronger desktop with an Intel Core i7-7700K at 4.2GHz, which is very similar to the PC used in their paper.

All methods are evaluated 10 times for EuRoC and 5 times for the other datasets on each sequence. Following [17], results are presented in cumulative error plots, which show how many sequences (y-axis) have been tracked with an accuracy better than the threshold on the x-axis. We perform $\mathbf{SE}(3)$ alignment of the trajectory with the provided ground-truth and report the root mean squared error (RMSE), also called absolute trajectory error (ATE). On TUM-VI and 4Seasons, trajectory lengths can vary greatly so we report the drift in $\%$, which we compute with drift $= \frac{\text{rmse}\cdot 100}{\text{length}}$. We also show tables to compare to numbers from other papers and report the median result for each sequence for our method.

A. *EuRoC dataset*

The EuRoC dataset [7] is the most popular visual-inertial dataset to date, and many powerful methods have been eval-

TABLE II: RMSE ATE in m on the TUM-VI dataset [8]. Best results in bold, underline is the best result among monocular methods. DM-VIO outperforms even state-of-the-art stereo-inertial methods by a large margin.

Sequence	ROVIO stereo	VINS mono	OKVIS stereo	BASALT stereo	DM-VIO mono	length [m]
corridor1	0.47	0.63	0.33	0.34	**0.19**	305
corridor2	0.75	0.95	0.47	**0.42**	0.47	322
corridor3	0.85	1.56	0.57	0.35	**0.24**	300
corridor4	**0.13**	0.25	0.26	0.21	**0.13**	114
corridor5	2.09	0.77	0.39	0.37	**0.16**	270
magistrale1	4.52	2.19	3.49	**1.20**	2.35	918
magistrale2	13.43	3.11	2.73	**1.11**	2.24	561
magistrale3	14.80	**0.40**	1.22	0.74	1.69	566
magistrale4	39.73	5.12	**0.77**	1.58	1.02	688
magistrale5	3.47	0.85	1.62	**0.60**	0.73	458
magistrale6	X	2.29	3.91	3.23	**1.19**	771
outdoors1	101.95	**74.96**	X	255.04	123.24	2656
outdoors2	21.67	133.46	73.86	64.61	**12.76**	1601
outdoors3	26.10	36.99	32.38	38.26	**8.92**	1531
outdoors4	X	16.46	19.51	17.53	**15.25**	928
outdoors5	54.32	130.63	13.12	7.89	**7.16**	1168
outdoors6	149.14	133.60	96.51	65.50	**34.86**	2045
outdoors7	49.01	21.90	13.61	**4.07**	5.00	1748
outdoors8	36.03	83.36	16.31	13.53	**2.11**	986
room1	0.16	0.07	0.06	0.09	**0.03**	146
room2	0.33	**0.07**	0.11	**0.07**	0.13	142
room3	0.15	0.11	**0.07**	0.13	0.09	135
room4	0.09	0.04	**0.03**	0.05	0.04	68
room5	0.12	0.20	0.07	0.13	**0.06**	131
room6	0.05	0.08	0.04	**0.02**	**0.02**	67
slides1	13.73	0.68	0.86	0.32	**0.31**	289
slides2	0.81	0.84	2.15	**0.32**	0.87	299
slides3	4.68	0.69	2.58	0.89	**0.60**	383
avg drift%	16.83*	1.700	0.815*	0.939	**0.472**	normalized

uated on it. In Table I we compare to the state-of-the art in visual-inertial odometry, all results are without loop-closure. Our method outperforms all other methods clearly in terms of RMSE. The closest competitor is Basalt [20], a stereo-inertial method which achieves a smaller error on 2 sequences. We also observe the lowest average scale error reported on the dataset so far, confirming that our contributions in IMU initialization have a positive impact on performance. In the supplementary we provide runtime evaluations, showing that tracking takes 10.34ms on average, and keyframe processing takes 53.67ms. The delayed marginalization is responsible for an overhead of 0.44ms or 0.8% in the keyframe thread.

B. TUM-VI dataset

The TUM-VI dataset [8] is a very challenging handheld dataset, featuring large-scale indoor and outdoor scenes, and even sequences sliding down a tube, where almost the full image is covered. With long periods of walking in straight lines, stereo methods have an advantage here as they still can observe the scale with constant motion. We compare to the state-of-the-art visual-inertial odometry methods evaluated in [8] in Table II. Our method clearly outperforms the other monocular method in VINS-Mono [3] on most sequences, and even compared to the stereo methods it shows the best result on 16 sequences and a mean drift of 0.472. The closest competitor is again Basalt, which achieves the best result on 8 sequences and a mean drift of 0.939.

On this dataset, we also evaluate against ORB-SLAM3 [5], which is the state-of-the art visual-inertial SLAM system. This is not entirely fair as ORB-SLAM3 uses loop closures

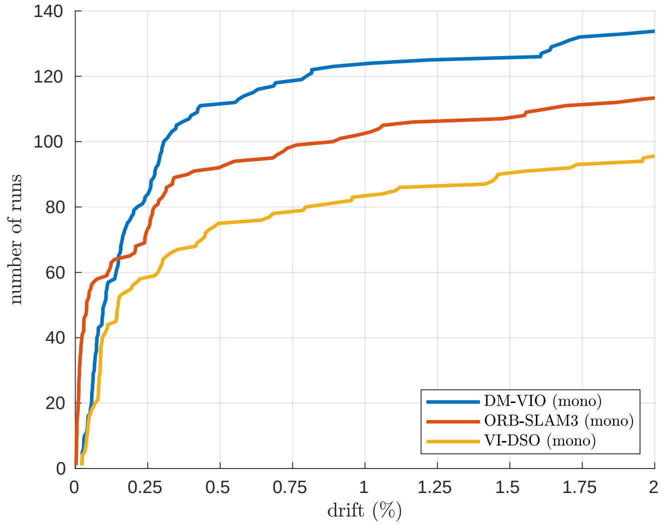


Fig. 4: Cumulative error plot for the TUM-VI dataset (drift in %). Our method clearly outperforms both VI-DSO and ORB-SLAM3 in terms of robustness. Thanks to its powerful loop closure system, ORB-SLAM3 has an advantage in terms of accuracy on some sequences.

(which cannot be disabled), constituting an advantage over the other methods. We find the comparison still helpful as it allows to make conclusions regarding the underlying odometry. We have evaluated ORB-SLAM3 5 times on each sequence and reproduced their results with code and settings provided by the authors. For this comparison we have also evaluated VI-DSO [6], and the results are shown in Fig. 4. We observe that ORB-SLAM3 is more accurate on some sequences thanks to its very strong loop closure system. However, our method is more robust overall. This indicates that an integration of loop closure and map reuse into our system would be an interesting future research direction.

C. 4Seasons dataset

The 4Seasons dataset [9] is a very recent automotive dataset, which, in contrast to most other car datasets, features a well time-synchronized visual-inertial sensor. The lower part of the images is obstructed by the car hood, hence we crop off the bottom 96 pixels, which we do for all methods. As this is the first odometry method to evaluate on the 4Seasons dataset, we make sure to determine IMU noise parameters for all methods the same way to ensure a fair comparison: We have manually read off the accelerometer and gyroscope noise density and bias random walk from the Allan variance plot provided in the data sheet of the IMU. To handle unmodeled effects we follow [8] and inflate noise values by different amounts to determine the best setting for all methods. For each method we tried noise models inflated by 1, 10, 100, 1000 respectively and chose the configuration which gave best results. For VI-DSO and for our method we slightly modified the visual initializer by adding a zero-prior to the translation on the x and y axis, and also added a threshold to stop keyframe creation for translations

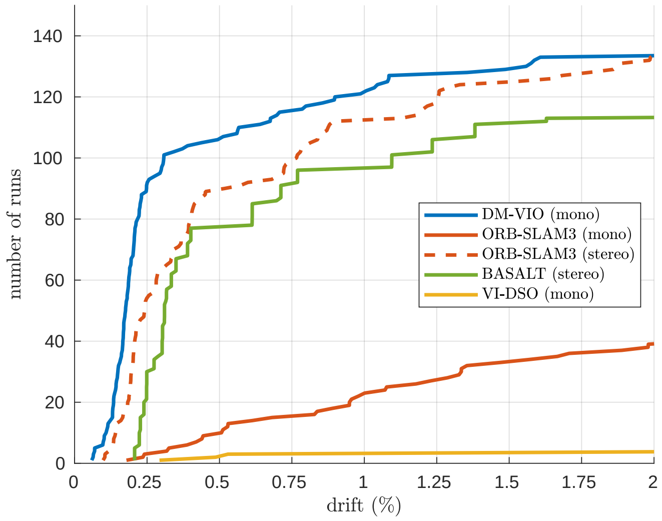


Fig. 5: Cumulative error plot for the 4Seasons dataset (drift in %). With lots of stretches with constant velocity, this dataset is extremely challenging for monocular visual-inertial methods. Thanks to our novel IMU initializer powered by delayed marginalization and PGBA, DM-VIO is able to cope with it and even outperforms stereo-inertial methods.

smaller than 0.01m (the latter was not activated for VI-DSO as it did not improve the results for it). Otherwise, parameters are the same as for the other experiments. For Basalt we tried all three provided default configurations with the optimal noise values to find the best settings. After choosing the configuration for each method, we perform one final evaluation, running all 30 sequences 5 times each.

The results are shown in Fig. 5. It is clear that the automotive scenario is very challenging for monocular methods. This is expected as it naturally features many stretches with constant motion, where scale is not observable, constituting a challenge for IMU initialization. Thanks to our novel IMU initialization, DM-VIO not only works well on the dataset but even outperforms stereo-inertial ORB-SLAM3 and Basalt, while using monocular images and no loop closures.

V. Conclusion and Future Work

We have presented a monocular visual-inertial odometry system which outperforms the state of the art, even stereo-inertial methods. Thanks to a novel IMU initializer, it works well in flying, handheld, and automotive scenarios, extending the applicability of monocular methods. The foundation of our IMU initialization is delayed marginalization, which also enables the pose graph bundle adjustment.

We anticipate that this method will spark further research in this direction. The idea of delayed marginalization could be applied to more use cases, e.g. for reactivating old keyframes in a marginalization setting to enable map reuse. The pose graph bundle adjustment can also be applied to long-term loop closures. Lastly, our open-source system is easily extendible, as all optimizations are integrated with GTSAM, allowing to quickly add new factors. This could be used for GPS integration, wheel odometry, and more.

References

[1] A. I. Mourikis and S. I. Roumeliotis, “A multi-state constraint Kalman filter for vision-aided inertial navigation,” in *ICRA*, 2007.

[2] J. Kaiser, A. Martinelli, F. Fontana, and D. Scaramuzza, “Simultaneous state initialization and gyroscope bias calibration in visual inertial aided navigation,” *RA-L*, vol. 2, no. 1, 2017.

[3] T. Qin, P. Li, and S. Shen, “VINS-Mono: A robust and versatile monocular visual-inertial state estimator,” *T-RO*, vol. 34, no. 4, 2018.

[4] R. Mur-Artal and J. D. Tardós, “Visual-inertial monocular slam with map reuse,” *RA-L*, vol. 2, no. 2, 2017.

[5] C. Campos, R. Elvira, J. J. G. Rodrguez, J. M. M. Montiel, and J. D. Tardós, “ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap slam,” *T-RO*, pp. 1–17, 2021.

[6] L. von Stumberg, V. Usenko, and D. Cremers, “Direct sparse visual-inertial odometry using dynamic marginalization,” in *ICRA*, May 2018.

[7] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, “The EuRoC micro aerial vehicle datasets,” *IJRR*, 2016.

[8] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stueckler, and D. Cremers, “The TUM VI benchmark for evaluating visual-inertial odometry,” in *IROS*, October 2018.

[9] P. Wenzel, R. Wang, N. Yang, Q. Cheng, Q. Khan, L. von Stumberg, N. Zeller, and D. Cremers, “4Seasons: A cross-season dataset for multi-weather SLAM in autonomous driving,” in *GCPR*, 2020.

[10] D. Nister, O. Naroditsky, and J. Bergen, “Visual odometry,” in *CVPR*, vol. 1, June 2004, pp. 652–659.

[11] A. Davison, I. Reid, N. Molton, and O. Stasse, “MonoSLAM: Real-time single camera SLAM,” *TPAMI*, vol. 29, 2007.

[12] G. Klein and D. Murray, “Parallel tracking and mapping for small AR workspaces,” in *ISMAR*, 2007.

[13] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, “ORB-SLAM: A versatile and accurate monocular slam system,” *T-RO*, vol. 31, Oct 2015.

[14] C. Kerl, J. Sturm, and D. Cremers, “Robust odometry estimation for RGB-D cameras,” in *ICRA*, 2013.

[15] R. Newcombe, S. Lovegrove, and A. Davison, “DTAM: Dense tracking and mapping in real-time,” in *ICCV*, 2011.

[16] J. Engel, T. Schöps, and D. Cremers, “LSD-SLAM: Large-scale direct monocular SLAM,” in *ECCV*, 2014.

[17] J. Engel, V. Koltun, and D. Cremers, “Direct sparse odometry,” *TPAMI*, vol. 40, 2018.

[18] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, “Robust visual inertial odometry using a direct EKF-based approach,” in *IROS*, 2015.

[19] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, “Keyframe-based visual-inertial odometry using nonlinear optimization,” *IJRR*, 2014.

[20] V. Usenko, N. Demmel, D. Schubert, J. Stueckler, and D. Cremers, “Visual-inertial mapping with non-linear factor recovery,” *RA-L*, vol. 5, no. 2, pp. 422–429, 2020.

[21] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, “Kimera: an open-source library for real-time metric-semantic localization and mapping,” in *ICRA*, 2020.

[22] A. Martinelli, “Closed-form solution of visual-inertial structure from motion,” *IJCV*, vol. 106, no. 2, 2014.

[23] E. Hong and J. Lim, “Visual-inertial odometry with robust initialization and online scale estimation,” *Sensors*, vol. 18, p. 4287, 12 2018.

[24] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, “IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation,” in *RSS*, 2015.

[25] K. MacTavish and T. D. Barfoot, “At all costs: A comparison of robust cost functions for camera correspondence outliers,” in *CRV*, 2015.

[26] T. Lupton and S. Sukkarieh, “Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions,” *T-RO*, vol. 28, no. 1, pp. 61–76, 2012.

[27] L. Carlone, Z. Kira, C. Beall, V. Indelman, and F. Dellaert, “Eliminating conditionally independent sets in factor graphs: A unifying perspective based on smart factors,” in *ICRA*, 2014.

[28] G. Huang, A. I. Mourikis, and S. Roumeliotis, “A first-estimates jacobian ekf for improving slam consistency,” in *ISER*, 2008.

[29] F. Daellert and Others, “Gtsam,” https://gtsam.org.

[30] J. Delmerico and D. Scaramuzza, “A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots,” in *ICRA*, 2018, pp. 2502–2509.

[31] C. Campos, J. Montiel, and J. D. Tardós, “Inertial-only optimization for visual-inertial initialization,” *ICRA*, pp. 51–57, 2020.

Additional evaluations for DM-VIO: Delayed Marginalization Visual-Inertial Odometry

Lukas von Stumberg and Daniel Cremers

We provide an ablation study on different parts of the IMU initializer in section I, an ablation study on the impact of the dynamic photometric weight in section II, and runtime results in section III.

I. Ablation IMU Initializer

We perform an ablation study on various parts of the IMU initializer. In particular, we compare to the following three baselines, which build on top of each other, meaning that everything removed in baseline n is also removed in baseline $n + 1$.

1) We remove the Reinitialization and the Marginalization replacement, corresponding to the orange and the yellow boxes in Fig. 3 of the main paper. Note that the scale is still optimized in the main system after initialization.
2) No readvancing: We do not replace the marginalization prior of the main graph after the first initialization. This means that in the purple box "Initialize" in Fig. 3 of the main paper we only replace the values and not the marginalization prior of the main graph. Instead we add a constant prior on the initial scale, which is necessary for the scale to not immediately diverge afterwards.
3) No PGBA: We remove the "PGBA IMU Init" and directly initialize with the result of the Coarse IMU Init.

The experiments are performed in non-realtime mode in order to not be dependent on the particular machine.

The results on the 4Seasons dataset are shown in Fig. S1. With each ablation the result becomes significantly worse, showing that all parts of the method contribute to the results. In particular we observe that the proposed delayed marginalization is very important, as it is the foundation of both the pose graph bundle adjustment (PGBA) and the readvancing (and subsequently also the marginalization replacement).

For completeness we should mention that on TUM-VI and EuRoC the contributions do not bring a significant performance improvement, as both datasets are much less challenging for the IMU initialization.

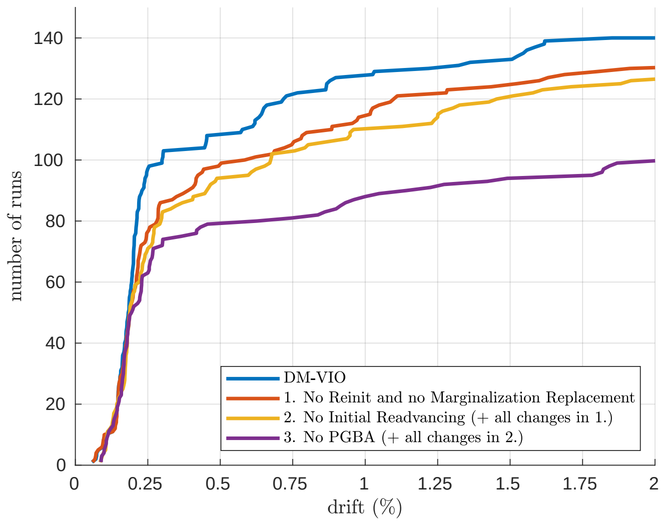


Fig. S1: Ablation study on various parts of the IMU initializer on the 4Seasons dataset (non-realtime). Removing the reinitialization and the marginalization replacement (1.) makes a significant difference, as the marginalization prior can become inconsistent if the initial scale estimate is off. Removing also the initial marginalization replacement powered by readvancing (2.) further deteriorates the performance. Removing the pose graph bundle adjustment and only using the Coarse IMU init (3.) makes the biggest difference. This ablation shows the significance of delayed marginalization which is the foundation of PGBA, readvancing and marginalization replacement.

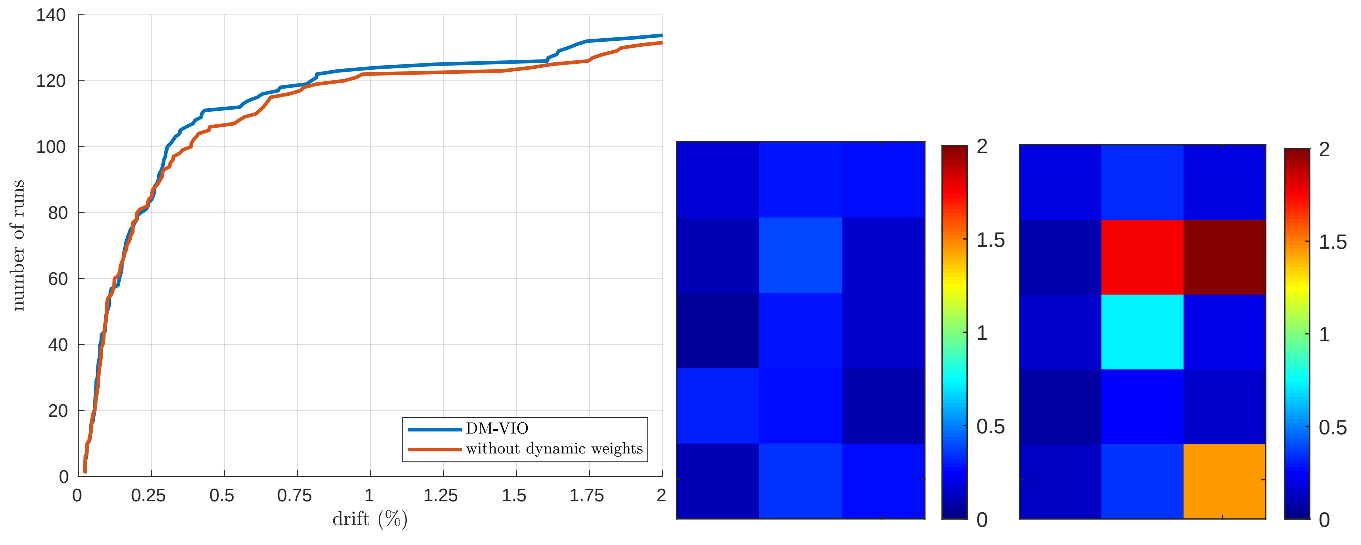
II. Ablation Dynamic Photometric Weight

We provide ablation studies showing the effect of the dynamic photometric weight proposed in the main paper. For this we disable the dynamic part and always use the constant weight $W = \lambda$ (of course λ is set to the same value as for the other results). In Fig. S2a we compare to the results shown in the main paper on TUM-VI, both runs are in realtime mode.

It can be seen that the dynamic weights improve the overall robustness of the method. The effect is most visible on the TUM-VI slides (Fig. S2b, S2c), where the version without dynamic weights does not work well in 4 of the 15 runs whereas our method works well in all runs. The dynamic weighting is designed for situations where the image quality gets really bad. On the other datasets, significant degradation of image quality is rare, hence there is only a marginal difference for them.

III. Runtime

We perform extensive runtime analysis on the same MacBook Pro 2013 (i7 at 2.3GHz) that was used for generating the results in the main paper. For each part of the method we save the mean time and standard deviation it takes while processing a sequence. We run our method in real-time mode 10 times on each sequence of EuRoC dataset, resulting in 110 recorded means and standard deviations for each part.



(a) Cumulative error plot.

(b) DM-VIO on the slides.

(c) Without dynamic weights on the slides

Fig. S2: Realtime results on the TUM-VI dataset with and without the dynamic photometric weights. a) Cumulative error plot. The dynamic weight provides a noticeable improvement in robustness. Only few sequences contain hard enough images to trigger the dynamic weight, hence the overall improvement can appear small. However for these hard sequences, success rate is increased. b) and c): Each colored square represents the drift (%) for one of the 5 runs (rows) on the three slides sequences (columns). With dynamic weights our method works well for all 30 runs whereas the version without them fails once and accumulates a large drift for three executions.

TABLE S1: We save runtime statistics for different parts of the method and show the mean over all 110 runs on the EuRoC dataset (10 times on each sequence). Top 2 sections: Runtime for the two main threads. Bottom 3 sections: Runtime for the parts of the IMU initializer, which are performed occasionally. Note that the overhead introduced by our initializer is small: The only regular overhead is the delayed marginalization, taking 0.44ms which amounts to 0.8% of the keyframe operations. Both, the Coarse IMU initializer and the PGBA initializer perform most operations in a separate thread when running. The only relevant overhead in the keyframe thread is after a successfull PGBA and during a marginalization replacement.

Component	Part	Mean time (ms) Mean of 110 runs	Standard deviation (ms) Mean of 110 runs
Coarse Tracking Performed for every frame.	Total	10.34	10.16
	Direct Image Alignment	4.70	1.31
	Build Image Pyramid	4.78	0.52
Keyframe Operations Performed for every keyframe.	Total	53.67	6.89
	Bundle Adjustment	26.49	5.57
	Create Candidate Points	5.83	0.38
	Trace candidate points	4.37	1.83
	Activate new points	3.56	0.77
	Keyframe Marginalization (full)	3.24	0.5
	— Drop residuals, recompute adjoints, etc.	— 1.71	— 0.39
	— Marginalize main graph	— 0.61	— 0.10
	— **Delayed marginalization**	— 0.44	— 0.20
Coarse IMU Init	Runtime in separate thread	9.08	5.63
	Overhead Coarse Tracking thread	0.05	0.04
PGBA (Re-)init	Runtime in separate thread	74.11	24.77
	— Optimization	— 54.72	— 23.40
	— Readvancing	— 7.61	— 1.37
	Overhead Keyframe thread when active	0.17	0.04
	Overhead Keyframe thread after success	14.50	8.52
	— Include KFs added while PGBA was running	— 13.15	— 8.52
	— Readvancing of the newly added KFs	— 1.35	— 0.19
Marginalization Replacement	Total (Keyframe Thread)	21.02	8.98
	Build Graph	1.15	0.45
	Readvance	19.87	8.97

We present the mean mean runtime and the mean standard deviation over the 110 runs in Table S1.

Even on the relatively old machine the overall runtimes are quite fast. The tracking could run at 97 FPS on average and the keyframe processing could run at 19 FPS. Note that the dataset is recorded at only 20 FPS, and to work well our method needs to process 5-6 keyframes per second.

The only regular overhead introduced by our initializer is the delayed marginalization, which amounts to 0.8% of the total processing time of each keyframe. The Coarse IMU initializer and the PGBA both run in separate threads when active. We do note that after a successful PGBA, some processing has to be done in the keyframe thread in order to include the keyframes, which have been bundle adjusted while the PGBA was running. Similarly the marginalization replacement currently runs in the keyframe thread. However both of these parts are comparably fast (14.50ms and 21.02ms respectively), and are performed only a few times during every run.