2024 IEEE International Conference on Robotics and Automation (ICRA)
May 13-17, 2024. Yokohama, Japan

DVI-SLAM: A Dual Visual Inertial SLAM Network

Xiongfeng Peng[1], Zhihua Liu[1], Weiming Li[1], Ping Tan[2], SoonYong Cho[3], Qiang Wang[1]

2024 IEEE International Conference on Robotics and Automation (ICRA) | 979-8-3503-8457-4/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICRA57147.2024.10610042

***Abstract*— Recent deep learning based visual simultaneous localization and mapping (SLAM) methods have made significant progress. However, how to make full use of visual information as well as better integrate with inertial measurement unit (IMU) in visual SLAM has potential research value. This paper proposes a novel deep SLAM network with dual visual factors. The basic idea is to integrate both photometric factor and re-projection factor into the end-to-end differentiable structure through multi-factor data association module. We show that the proposed network dynamically learns and adjusts the confidence maps of both visual factors and it can be further extended to include the IMU factors as well. Extensive experiments validate that our proposed method significantly outperforms the state-of-the-art methods on several public datasets, including TartanAir, EuRoC and ETH3D-SLAM. Specifically, when dynamically fusing the three factors together, the absolute trajectory error for both monocular and stereo configurations on EuRoC dataset has reduced by 45.3% and 36.2% respectively.**

I. Introduction

State-of-the-art (SOTA) deep learning-based SLAM methods can be roughly categorized into two classes according to the camera pose estimation process. The first class consists of direct regression-based methods such as DEMON [1], DeepTAM [2] and DytanVO [3]. These methods regress the camera pose with a deep network, which often suffer from limited accuracy and generalization problem. The other class consists of optimization-based methods such as CodeSLAM [4], BA-Net [5], DeepFactors [6], and DROID-SLAM [7]. These methods usually include a bundle adjustment (BA) optimizer that minimizes residuals from various related factors corresponding to different aspects of data constraints. CodeSLAM [4] encodes images into compact representations and infers camera pose by minimizing classical photometric and depth residuals with a non-differentiable damped Gauss-Newton algorithm. BA-Net [5] achieves end-to-end learning by integrating a differentiable optimizer into a deep neural network and infers pose by minimizing photometric residual in feature space (also referred to as feature-metric residual). However, these photometric-based methods typically need a good initial guess for the optimizer to work well. DeepFactors [6] proposes to combine photometric, re-projection, and depth factors on CodeSLAM [4], but without considering each factor's reliability. Later, DROID-SLAM [7] learns to optimize camera pose and depth iteratively by minimizing the re-projection residual through a dense bundle adjustment (DBA) optimizer. However, the re-projection-based method heavily relies on correct data association, and noisy correlations could cause severe errors in the estimated flow and result in inaccurate camera pose. The above mentioned methods use only visual factors. It is well-known that including an additional inertial measurement unit (IMU) can enhance the robustness of visual SLAM methods, especially in fast movement. Some direct regression-based deep-learning SLAM methods have tried to integrate IMU factors such as VINet [8] and DeepVIO [9]. But how to effectively fuse all the different visual factors and IMU factors into an end-to-end deep network still poses an open question for the research community.

In this paper, we propose a novel optimization-based deep SLAM method, which is named Dual Visual Inertial SLAM network (DVI-SLAM). DVI-SLAM simultaneously infers camera pose and dense depth by dynamically fusing multiple factors with an end-to-end trainable differentiable structure. Here, the dual visual factors refer to the feature-metric factor and the re-projection factor, which provide complimentary cues to explore the visual information. The network also extends naturally to effectively include the IMU factor or other factors. A key design of our DVI-SLAM is its dynamic multi-factor fusion that learns confidence maps to adjust relative strengths among the different factors for each iteration in the optimization process. In particular, the re-projection factor dominates at the early stage of BA optimization to get a good initialization and avoid getting stuck in local minima. In the later stage, both the feature-metric factor and re-projection factor smoothly steer the joint optimization towards a true minimum. The network also dynamically adjusts the confidence map of IMU factor according to the estimated pose error for a reliable visual-inertial fusion. Our dual visual and IMU constraints are fused and minimized in a tightly-coupled manner. To the best of our knowledge, our DVI-SLAM is the first deep learning-based framework that allows tightly coupling dual visual inertial SLAM with DBA optimization.

Our main contributions are summarized as follows:

- We propose a DVI-SLAM network that dynamically fuses re-projection, feature-metric, and IMU factors with learned confidence maps to improve camera pose estimation in challenging scenes.
- Our proposed end-to-end learnable framework is flexible to include various factors, supporting monocular or stereo, with or without depth or IMU sensors.
- Our method exceeds the SOTA methods on several public datasets: TartanAir, EuRoC and ETH3D-SLAM.

Xiongfeng Peng[1], Zhihua Liu[1], Weiming Li[1] and Qiang Wang[1] are with SAIT-China Lab, Samsung R&D Institute China-Beijing, China {xf.peng, zhihua.liu, weiming.li, qiang.w}@samsung.com

Ping Tan[2] with Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, China

SoonYong Cho[3] with Multimedia System TU, Samsung Advanced Institute of Technology, South Korea

Authorized licensed use limited to: ShanghaiTech University. Downloaded on September 11,2026 at 02:55:40 UTC from IEEE Xplore. Restrictions apply.

II. Related Work

The existing SLAM methods are categorized according to the factors in the optimization process and they are introduced as follows.

a) Re-projection-based SLAM Methods: These methods first extract keypoints and descriptors from images and then build data association [10]–[13], [13]–[17]. The estimated camera pose and 3D map are optimized with BA by minimizing the re-projection error. PTAM [11] is the first real-time optimization-based SLAM system in this category. Later, ORB-SLAM3 [15] combines the re-projection factors and fuses it with an IMU sensor on the basis of [18]–[20]. Recently, DROID-SLAM [7] proposes a deep dense feature-based SLAM network to iteratively optimize the camera pose and dense depth. Based on DROID-SLAM, DPVO [21] proposes a deep visual odometry system built using a sparse patch representation, and PVO [22] proposes a panoptic visual odometry framework to achieve more comprehensive modeling of the scene motion, geometry, and panoptic segmentation information.

b) Photometric-based SLAM Methods: Photometric-based SLAM methods [23]–[28] generally utilize raw pixel intensities and define an objective function with photometric error. Similar to photometric factors, feature-metric factors use deep features extracted by neural networks instead of pixel intensities. CodeSLAM [4] encodes observed images into a compact and optimizable representation that is used in a keyframe-based SLAM system to estimate both camera poses and depths. BA-Net [5] minimizes the feature-metric residual with the combined learned depth basis, which has poor generalization and does not converge well. One advantage of photometric-based SLAM methods is that they allow the inclusion of abundant image information. This empowers these methods to work well as long as the projection residual is reasonably small instead of requiring correct explicit data association.

c) IMU-related SLAM Methods: IMU sensor is widely used in traditional SLAM systems such as ORB-SLAM3 [15], ICE-BA [16], VINS [17], and OKVIS [13]. For deep learning-based SLAM methods, there are several IMU-related works. VINet [8] is the first deep visual-inertial SLAM network that directly concatenates the deep features with IMU feature and directly regresses the camera pose. DeepVIO [9] fuses the visual and inertial features with a designed fully connected layer without considering the reliability of different sensors. Closely related to the above three categories of methods, our proposed DVI-SLAM method dynamically fuses re-projection, feature-metric, and IMU factors in an end-to-end differentiable network.

III. Two-view Reconstruction

In this section, we introduce our DVI-SLAM with two-view reconstruction for clarity. The complete system is described in Section IV. Here, the two-view reconstruction refers to the estimation of camera pose, IMU motion and inverse depth from two views.

Figure 1 illustrates our DVI-SLAM network for the two-view reconstruction. Given inputs of images $\boldsymbol{I}_i$, $\boldsymbol{I}_j$ and the accelerometer and gyroscope measurements of IMU $\{\boldsymbol{\alpha}_k, \boldsymbol{\omega}_k\}$ between the two frames, the network outputs the updated camera pose $\boldsymbol{T}_i$,$\boldsymbol{T}_j$, IMU motion $\boldsymbol{M}_i$,$\boldsymbol{M}_j$ and dense inverse depth $\boldsymbol{d}_i$. Where the camera pose $\boldsymbol{T} = (\boldsymbol{R}, \boldsymbol{p})$ contains rotation $\boldsymbol{R}$ and translation $\boldsymbol{p}$. The IMU motion $\boldsymbol{M} = (\boldsymbol{v}, \boldsymbol{ba}, \boldsymbol{bg})$ includes velocity $\boldsymbol{v}$, accelerator bias $\boldsymbol{ba}$, and gyroscope bias $\boldsymbol{bg}$. Our DVI-SLAM network is built upon DROID-SLAM [7]. It consists of two modules and one layer: (a) a feature extraction module, (b) a multi-factor data association module, and (c) a multi-factor DBA layer. In the following sections, we introduce these modules and layers respectively.

A. Feature Extraction

In the feature extraction module, two kinds of features are extracted. One feature is the correlation feature $\boldsymbol{F}^c$ which is used to estimate a dense flow map and build the correspondence between two frames. The other is the deep appearance feature $\boldsymbol{F}^a$ with more details for computing the feature-metric residual.

Different from DROID-SLAM [7], we add an encoder-decoder branch with a U-Net structure [29] to extract the appearance feature. The encoder has two sub-modules and each contains two convolutional layers followed by a downsampling layer and max pooling operator. The decoder also has two sub-modules and the feature map resolution increases through sequential upsampling until matches the resolution of the input image.

For the module outputs, the resolution of the appearance feature map is eight times that of the correlation feature map. More details are kept in the deep appearance feature map by the skip connection of the U-Net structure.

B. Multi-factor Data Association

The multi-factor data association module learns the feature correspondence and the confidence maps for fusion with re-projection, feature-metric, and IMU. Figure 1 (b) illustrates the detailed process of one iteration. Similar to DROID-SLAM [7], a pyramid 4D correlation volume $\boldsymbol{C}_{ij}$ is built by inputs from the extracted correlation features $\boldsymbol{F}_i^c, \boldsymbol{F}_j^c$. And the projected pixel coordinate $\boldsymbol{x}_{ij}$ is calculated by the updated pose $\boldsymbol{T}_i, \boldsymbol{T}_j$, inverse depth $\boldsymbol{d}_i$ from last iteration. Then the network looks up neighbourhood correlation feature maps from the 4D correlation volume $\boldsymbol{C}_{ij}$ with $\boldsymbol{x}_{ij}$ and feed the correlation feature maps into a ConvGRU network. The $\boldsymbol{x}_{ij}$ represents the projected pixel coordinates from the pixel coordinates $\boldsymbol{x}_i$ of $\boldsymbol{I}_i$ to $\boldsymbol{I}_j$ using pose $\boldsymbol{T}_i, \boldsymbol{T}_j$, inverse depth $\boldsymbol{d}_i$. Specifically, $\boldsymbol{x}_{ij} = \Pi(\boldsymbol{T}_i, \boldsymbol{T}_j, \boldsymbol{d}_i, \boldsymbol{x}_i)$, where Π represents the projection.

Since the deep network has the capability to learn complex priors, we design a multi-head GRU module to learn the uncertainty of different factors for subsequent dynamic residual fusion. Specifically, the multi-head GRU module has four independent output branches: IMU confidence branch, re-projection confidence branch, feature-metric confidence

Authorized licensed use limited to: ShanghaiTech University. Downloaded on September 11,2026 at 02:55:40 UTC from IEEE Xplore. Restrictions apply.

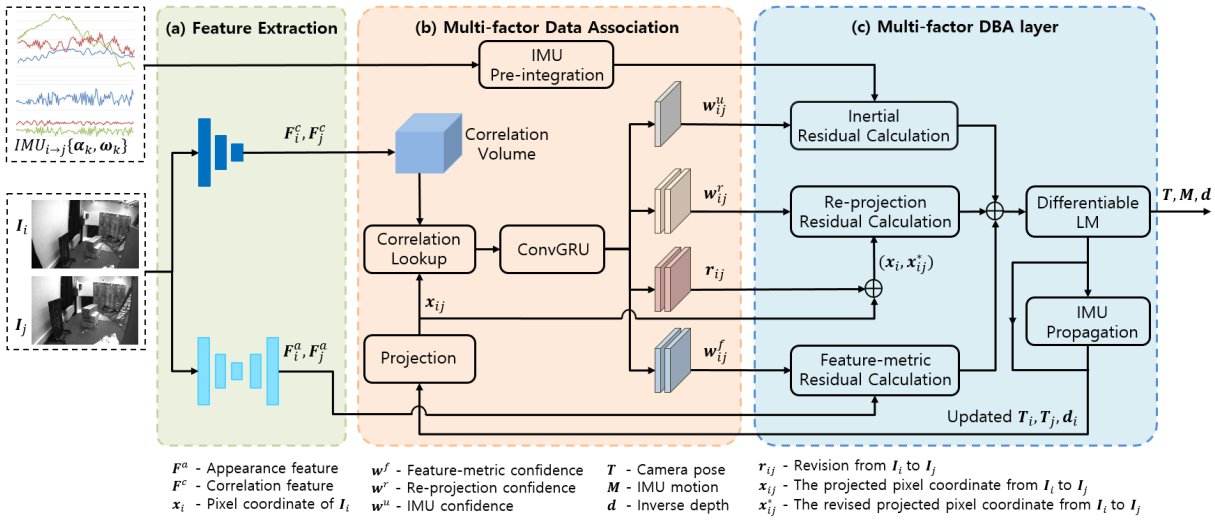

Fig. 1: Overview of our DVI-SLAM structure (the two-views reconstruction only for better clarity). (a) Visual correlation feature and appearance feature are extracted by the input two frames $\boldsymbol{I}_i, \boldsymbol{I}_j$. (b) The multi-factor data association module learns the feature correspondence and confidence maps, and IMU pre-integration results are calculated by the accelerometer and gyroscope measurements of IMU $\{\boldsymbol{\alpha}_k, \boldsymbol{\omega}_k\}$ between the two frames. (c) Those confidence maps dynamically fuse re-projection, feature-metric, and IMU residuals for optimization via a differentiable LM for the pose $\boldsymbol{T}$, IMU motion $\boldsymbol{M}$, and inverse depth $\boldsymbol{d}$. $\boldsymbol{T}$ is also calculated via the IMU propagation. Finally, $\boldsymbol{T}$, $\boldsymbol{M}$, and $\boldsymbol{d}$ are updated for the next iteration.

branch, and flow revision branch. For the IMU confidence branch, it has one 1×1 convolution and one average pooling layer. For the other three branches, two 3×3 convolutional layers are applied. In addition, to predict the confidence maps of re-projection and feature-metric branches, one sigmoid operator is used for each branch respectively.

Figure 2 visualizes our learned re-projection and feature-metric confidence maps. We notice that at the beginning of the optimization iteration, the confidence map of the feature-metric factor is small and the re-projection factor is large. When the iteration continues, the confidence map of feature-metric factor increases gradually, and it will be slightly larger than the re-projection factor at the end of the iteration. In both re-projection and feature-metric confidence maps, the locations with high confidence are well-textured. It's reasonable because the well-textured areas easily build explicit and implicit correspondences. The multi-head GRU module learns IMU confidence as well. The insight is that the pose from the DBA layer is updated again through IMU propagation. In this step, the IMU information is implicitly injected into the multi-head GRU module in the network. Based on the learned confidence, Mahalanobis distance is used to weigh the residual of each factor. The larger the average confidence, the more reliable the factor.

C. Multi-factor DBA Layer

With the multi-factor confidences, the residuals are fused and updated iteratively in the DBA layer. The following section introduces the confidence-based residual definition of each factor in detail.

a) Confidence-based Re-projection Residual: Same as DROID-SLAM [7], according to the correspondence

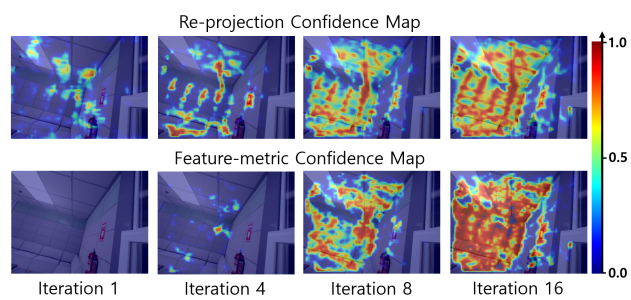

Fig. 2: The confidence map changes iteratively. High confidence in the re-projection confidence map focuses on corner pixels and extends to edge pixels gradually with iteration increasing. For the feature-metric confidence map, the confidence is zero at the beginning of the iteration and focuses on the well-textured regions after several iterations.

$(\boldsymbol{x}_i, \boldsymbol{x}^*_{ij})$ of the pixel coordinates and the learned re-projection confidence map $\boldsymbol{w}^r_{ij}$, the re-projection residual for two-view is defined as in Equation 1.

$$E^r(\boldsymbol{T}, \boldsymbol{d}) = ||\boldsymbol{x}^*_{ij} - \Pi(\boldsymbol{T}_i, \boldsymbol{T}_j, \boldsymbol{d}_i, \boldsymbol{x}_i)||^2_{\sum^r_{ij}}, \quad \sum^r_{ij} = diag\boldsymbol{w}^r_{ij} \tag{1}$$

, where $||\cdot||_{\sum_{ij}}$ is the Mahalanobis distance. The larger the confidences, the more reliable the correspondences. The residual of each correspondence in the re-projection term updates iteratively.

b) Confidence-based Feature-metric Residual: Based on the appearance features $\boldsymbol{F}^a_i, \boldsymbol{F}^a_j$ and the learned feature-metric confidence $\boldsymbol{w}^f_{ij}$, we calculate the two-view feature-metric residual as in Equation 2.

Authorized licensed use limited to: ShanghaiTech University. Downloaded on September 11,2026 at 02:55:40 UTC from IEEE Xplore. Restrictions apply.

$$E^f(\boldsymbol{T},\boldsymbol{d}) = ||\boldsymbol{F}_i^a(\boldsymbol{x}_i) - \boldsymbol{F}_j^a(\Pi(\boldsymbol{T}_i,\boldsymbol{T}_j,\boldsymbol{d}_i,\boldsymbol{x}_i))||^2_{\sum_{ij}^f}, \quad \sum\nolimits_{ij}^f = diag\boldsymbol{w}_{ij}^f \tag{2}$$

c) Confidence-based Inertial Residual: Following the traditional VI-SLAM methods [16], [17], we compute the inertial residual between image $\boldsymbol{I}_i$ and $\boldsymbol{I}_j$. Based on IMU pre-integration [30], the Δ in Equation 4 and Equation 5, and the pre-integration weight $\boldsymbol{w}_{ij}^{pu}$ can be estimated. The bias $\hat{\boldsymbol{b}}_i = (\hat{\boldsymbol{ba}}_i, \hat{\boldsymbol{bg}}_i)$ is the bias estimate at the time of pre-integration. The IMU weight $\boldsymbol{w}_{ij}^{imu}$ is represented by $\boldsymbol{w}_{ij}^{imu} = \boldsymbol{w}_{ij}^{pu}\boldsymbol{w}_{ij}^{u}$, where $\boldsymbol{w}_{ij}^{u}$ is the learned IMU confidence. It should be noted that there is an extrinsic difference between the IMU pose and the camera pose. We calculate the two-view inertial residual as in Equation 3.

$$E^u(\boldsymbol{T},\boldsymbol{M}) = \left|\left|\begin{matrix} e_{ij}^T \\ e_{ij}^M \end{matrix}\right|\right|^2_{\sum_{ij}^u}, \quad \sum\nolimits_{ij}^u = diag\boldsymbol{w}_{ij}^{imu} \tag{3}$$

$$e_{ij}^T = \begin{bmatrix} Log((\Delta\boldsymbol{R}_{ij}Exp(\Delta\boldsymbol{J}_{ij}^r(\boldsymbol{bg}_i - \hat{\boldsymbol{bg}}_i)))^T\boldsymbol{R}_i^T\boldsymbol{R}_j) \\ \boldsymbol{R}_i^T(\boldsymbol{p}_j - \boldsymbol{p}_i - \boldsymbol{v}_i\Delta t_{ij} - \frac{1}{2}\boldsymbol{g}t_{ij}^2) - (\Delta\boldsymbol{p}_{ij} + \Delta\boldsymbol{J}_{ij}^p(\boldsymbol{b}_i - \hat{\boldsymbol{b}}_i)) \end{bmatrix} \tag{4}$$

$$e_{ij}^M = \begin{bmatrix} \boldsymbol{R}_i^T(\boldsymbol{v}_j - \boldsymbol{v}_i - \boldsymbol{g}t_{ij}) - (\Delta\boldsymbol{v}_{ij} + \Delta\boldsymbol{J}_{ij}^v(\boldsymbol{b}_i - \hat{\boldsymbol{b}}_i)) \\ \boldsymbol{ba_j} - \boldsymbol{ba_i} \\ \boldsymbol{bg_j} - \boldsymbol{bg_i} \end{bmatrix} \tag{5}$$

d) Multi-factor Residual Fusion: After getting the re-projection, feature-metric, and IMU residuals of two-view, the total residual $E(\boldsymbol{T},\boldsymbol{M},\boldsymbol{d})$ in Equation 6 is obtained by summing the residuals of each factor and is minimized in the DBA layer. After multiple iterations and updates, the system outputs the optimal camera pose $\boldsymbol{T}$, inverse depth $\boldsymbol{d}$, and IMU motion $\boldsymbol{M}$.

$$E(\boldsymbol{T},\boldsymbol{M},\boldsymbol{d}) = E^r(\boldsymbol{T},\boldsymbol{d}) + E^f(\boldsymbol{T},\boldsymbol{d}) + E^u(\boldsymbol{T},\boldsymbol{M}) \tag{6}$$

IV. DVI-SLAM System

Same as DROID-SLAM [7], for an input video stream, multiple keyframes are extracted to construct a frame graph $G = (\boldsymbol{V}, \boldsymbol{\varepsilon})$ which indicates the co-visibility among these keyframes. Equation 6 is extended into Equation 7 and optimizes the total residual of the entire frame graph via a differentiable Levenberg-Marquardt (LM).

$$E(\boldsymbol{T},\boldsymbol{M},\boldsymbol{d}) = \sum_{(i,j)\in\boldsymbol{\varepsilon}} E^r(\boldsymbol{T},\boldsymbol{d}) + \sum_{(i,j)\in\boldsymbol{\varepsilon}} E^f(\boldsymbol{T},\boldsymbol{d}) + \sum_{(i,j=i+1)\in\boldsymbol{\varepsilon}} E^u(\boldsymbol{T},\boldsymbol{M}) \tag{7}$$

In the following section, we introduce the DVI-SLAM system initialization and the process in detail.

a) Initialization: Same as DROID-SLAM method [7], we initialize DVI-SLAM when there is only visual stream input. When there is both visual and inertial stream inputs, we follow VINS-Mono method [17] for DVI-SLAM initialization. Specifically, there are three steps for initialization. Firstly, multiple keyframes are collected to build a frame graph, and camera pose and inverse depth states are optimized by DBA layer. Then we initialize the gravity direction and IMU motion according to the estimated camera pose and IMU pre-integration result. For the monocular visual and inertial streams, the absolute scale is also initialized. Finally, we minimize re-projection residual, feature-metric residual, and inertial residual with DBA layer to refine camera pose, motion, and inverse depth.

b) VI-SLAM process: After the system initialization is finished, for a new frame and its corresponding IMU data, we first estimate the average flow between the new frame and its latest keyframe. If the new frame is considered a keyframe, it is added to the sliding window and the frame graph which establishes edges with its temporally adjacent keyframes. The new frame pose and IMU motion are optimized in the sliding window with local BA. We use the same strategy as DROID-SLAM [7] to keep a constant sliding window size and get the poses of non-keyframes. The camera pose after optimizing with local BA is called visual-inertial odometry (VIO) output. Based on the VIO result in the sequence, we build a new frame graph in which the keyframes are connected spatially and temporally. After performing an extra global BA, the results are called VI-SLAM output which reflects the accuracy of the reconstructed global map.

V. Experiments

To validate our DVI-SLAM system, we make extensive experiments on both synthetic and real-world datasets with different sensor combinations. Specifically, we use public TartanAir dataset [31], TartanAir SLAM challenge [31], EuRoC dataset [32], and ETH3D-SLAM benchmark [33]. The absolute trajectory error (ATE) [34] metric is used to evaluate the error of the pose. Since there is no absolute scale for monocular input, we additionally optimize the scale (Sim3) when evaluating the trajectory error and name the metric as ATE-scale. In order to better illustrate the effectiveness and robustness of our method, we compare our DVI-SLAM with both traditional and learning-based visual/visual-inertial odometry (VO/VIO) and visual/visual-inertial SLAM (V/VI-SLAM) systems. Due to limited space, we provide more experimental results in the supplementary material.

A. Training and Implementation

a) Training: We train the network with a weighted sum of pose loss L_{pose}, flow loss L_{flow}, re-projection loss L_{repro}, and feature-metric loss L_{metric}. Following DROID-SLAM [7], the pose loss L_{pose} is the distance between ground truth (GT) pose and the predicted pose, and the L_{flow} flow loss is the distance between the predicted flow and GT flow which is converted from GT pose and depth. Re-projection loss L_{repro} and feature-metric loss L_{metric} are the distances of the re-projection error and feature-metric error. The total training loss L is expressed in Equation 8 and the hyperparameters $w1, w2, w3, w4$ are set $10, 0.05, 0.01, 0.01$ respectively:

$$L = w1 * L_{pose} + w2 * L_{flow} + w3 * L_{repro} + w4 * L_{metric} \tag{8}$$

b) Implementation: We train our DVI-SLAM network on four NVIDIA RTX 3090 GPUs with three stages. In the first two stages, our network is trained on monocular video stream with 384×512 resolution from the TartanAir dataset [31]. Each training example consists of a 7-frame

Authorized licensed use limited to: ShanghaiTech University. Downloaded on September 11,2026 at 02:55:40 UTC from IEEE Xplore. Restrictions apply.

TABLE I: The trajectory error comparison of our method with SOTA methods on TartanAir monocular SLAM challenge.

Mono(ATE-scale(m))↓		MH000	MH001	MH002	MH003	MH004	MH005	MH006	MH007	Avg
VO	DeepV2D [35]	6.15	2.12	4.54	3.89	2.71	11.55	5.53	3.76	5.03
	TartanVO [36]	4.88	0.26	2.00	0.94	1.07	3.19	1.00	2.04	1.92
	DROID-SLAM* [7]	0.50	0.08	0.10	0.06	1.33	0.53	0.13	2.18	0.61
	DVI-SLAM(Ours)	**0.19**	**0.04**	**0.04**	**0.03**	**0.93**	**0.11**	**0.08**	**1.19**	**0.33**
SLAM	ORB-SLAM [18]	1.30	0.04	2.37	2.45	-	-	21.47	2.73	-
	DROID-SLAM [7]	**0.08**	0.05	0.04	0.02	**0.01**	1.31	0.30	**0.07**	0.24
	DROID-SLAM* [7]	0.58	0.05	0.03	0.03	1.30	0.52	0.13	0.14	0.35
	DVI-SLAM(Ours)	**0.08**	**0.03**	**0.02**	**0.01**	0.63	**0.12**	**0.06**	0.09	**0.13**

TABLE II: The trajectory error comparison of our method with the SOTA VO and VIO methods on EuRoC dataset.

VO/VIO		MH01	MH02	MH03	MH04	MH05	V101	V102	V103	V201	V202	V203	Avg
Mono ATE-scale(m)↓	DSO [26]	**0.046**	**0.046**	0.172	3.810	**0.110**	0.089	**0.107**	0.903	**0.044**	0.132	1.152	0.601
	SVO [37]	0.100	0.120	0.410	0.430	0.300	0.070	0.210	-	0.110	0.110	1.080	-
	DeepV2D [35]	0.739	1.144	0.752	1.492	1.567	0.981	0.801	1.570	0.290	2.202	2.743	1.298
	TartanVO [36]	0.639	0.325	0.550	1.153	1.021	0.447	0.389	0.622	0.433	0.749	1.152	0.680
	DROID-SLAM [7]	0.163	0.121	0.242	0.399	0.270	0.103	0.165	0.158	0.102	0.115	**0.204**	0.186
	DROID-SLAM* [7]	0.063	0.156	0.207	0.264	0.495	0.092	0.204	**0.087**	0.116	0.080	0.207	0.179
	DVI-SLAM(Ours)	0.120	0.063	**0.076**	**0.161**	0.354	**0.060**	0.207	0.113	0.087	**0.061**	0.268	**0.143**
Mono+IMU ATE(m)↓	MSCKF [12]	0.420	0.450	0.230	0.370	0.480	0.340	0.200	0.670	0.100	0.160	1.130	0.414
	OKVIS [13]	0.160	0.220	0.240	0.340	0.470	0.090	0.200	0.240	0.130	0.160	0.290	0.231
	VINS-Mono [17]	0.270	0.120	0.130	0.230	0.350	**0.070**	**0.100**	0.130	**0.080**	0.080	0.210	0.161
	DVI-SLAM(Ours)	**0.063**	**0.083**	**0.101**	**0.187**	**0.163**	0.074	0.114	**0.083**	0.091	**0.045**	**0.072**	**0.098**
Stereo ATE(m)↓	SVO [37]	**0.040**	0.070	0.270	0.170	0.120	**0.040**	**0.040**	0.070	0.050	0.090	0.790	0.159
	VINS-Fusion [38]	0.540	0.460	0.330	0.780	0.500	0.550	0.230	-	0.230	0.200	-	-
	DROID-SLAM* [7]	0.065	0.042	0.098	0.191	0.133	0.063	0.045	0.043	**0.040**	0.054	0.101	0.080
	DVI-SLAM(Ours)	0.043	**0.041**	**0.064**	**0.148**	**0.114**	0.063	**0.040**	**0.031**	0.049	**0.050**	**0.080**	**0.066**
Stereo+IMU ATE(m)↓	OKVIS [13]	0.188	0.123	0.192	0.185	0.292	**0.045**	0.088	0.123	0.056	0.114	0.199	0.146
	VINS-Fusion [38]	0.159	0.156	0.126	0.279	0.290	0.078	0.068	0.112	0.059	0.091	0.096	0.138
	OKVIS2 [39]	0.049	**0.040**	0.084	0.115	0.156	0.047	0.075	0.037	0.042	0.042	**0.047**	0.076
	DVI-SLAM(Ours)	**0.042**	0.046	**0.081**	**0.072**	**0.069**	0.059	**0.034**	**0.028**	**0.040**	**0.039**	0.055	**0.051**

video sequence. We fix the first two poses to GT poses of each training sequence to remove the 6-DOF and scale gauge freedoms for learning visual-related parameters.

In the first stage, similar to DROID-SLAM [7], the network learns correlation feature module, ConvGRU module, re-projection confidence branch and flow revision branch and is trained for 250k steps. The network is supervised with GT pose, GT flow, and re-projection consistency.

In the second stage, we first freeze the parameters learned from the last stage. Then we learn appearance feature module and feature-metric confidence branch for 50k steps. The network is supervised by GT pose, GT flow, and re-projection consistency for the first 10k steps, and then is jointly supervised with feature-metric consistency for the remaining 40k steps.

In the third stage, the network learns IMU confidence branch. We train it on monocular video stream with 384×512 resolution and IMU stream from the ETH3D-SLAM dataset [33]. We input GT pose and motion of the first frame, without fixing any frame. Since the ETH3D-SLAM dataset has only GT pose annotations, the average flow cannot be calculated, so we dynamically generate videos by sampling paths in the number of frame intervals between 5 and 10. In this stage, we first freeze the parameters learned from the last two stages, then the IMU confidence branch is learned for 10k steps and is supervised by GT pose, re-projection, and feature-metric consistency.

For all the above three-stage training processes, the batch size, frame clips, and data association update iterations are set to 4, 7, and 15 respectively.

B. Comparison with the SOTA Methods

In this section, we compare the VO/VIO and V/VI-SLAM output of our proposed DVI-SLAM method with the SOTA methods. For a fair comparison, we train DROID-SLAM [7] from scratch. Our trained DROID-SLAM model has nearly the same accuracy as the author claimed and is considered as the baseline (DROID-SLAM*).

a) Results on Synthetic Data: Table I shows the trajectory error comparison of our DVI-SLAM with the SOTA methods on the TartanAir monocular SLAM challenge. We compare the trajectory error across all Hard sequences. From the tables, we can see that the results of our method are significantly better than the SOTA methods except for the MH004 sequence. After examining the sequence, we found that a few frames with mostly a white wall lead to larger pose estimation errors.

b) Results on Real-world Data: Besides the synthetic dataset, we also compare our DVI-SLAM with the SOTA methods on real-world visual-inertial datasets. Table II compares the trajectory error of our method with the SOTA VO and VIO methods on EuRoC dataset. As shown in the table, our VO result exceeds the DROID-SLAM's result for most of the sequences. In comparison with other SOTA traditional or learning-based VO and VIO methods, our method also achieves the lowest average ATE.

Table III shows the comparison results on the ETH3D-SLAM benchmark with RGB-D input. Without any finetuning on the dataset, our DVI-SLAM achieves the Top-1 place on the ETH3D-SLAM benchmark.

Authorized licensed use limited to: ShanghaiTech University. Downloaded on September 11,2026 at 02:55:40 UTC from IEEE Xplore. Restrictions apply.

TABLE III: Comparison on ETH3D-SLAM benchmark.

Method	Input	Max error=2cm		Max error=8cm	
		AUC(train)↑	AUC(test)↑	AUC(train)↑	AUC(test)↑
BundleFusion [40]	RGB-D	8.29	2.59	84.10	33.84
ElasticFusion [41]	RGB-D	9.25	2.69	89.06	34.02
DVO-SLAM [42]	RGB-D	26.88	5.83	193.89	71.83
ORB-SLAM3 [15]	RGB-D	9.46	5.82	113.55	73.33
ORB-SLAM2 [19]	RGB-D	23.03	17.84	156.10	104.28
BAD-SLAM [33]	RGB-D	56.24	30.89	280.05	153.47
GO-SLAM [43]	RGB-D	-	33.12	-	197.02
DROID-SLAM [7]	RGB-D	54.61	32.84	340.42	207.79
DVI-SLAM(Ours)	RGB-D	**62.52**	**37.97**	**354.71**	**211.60**

C. Ablation Study

Table IV shows the effectiveness of our dual visual factor design that improves performance on the synthetic TartanAir validation dataset [31], which is defined by DROID-SLAM [7]. From the table, we notice that our DVI-SLAM method achieves about 40% trajectory error reduction in comparison to the baseline method on the output of VO or SLAM regardless of the monocular or stereo input. In particular, we add feature-metric factor with different weights. Firstly, a fixed weight is used and is named "B + F_Metric_fixed $\boldsymbol{w}^f = 1$", but the results become worse. Then we manually tune the weight from small to large in the iteration process and named it "B + F_Metric_tuned $\boldsymbol{w}^f$". In comparison, our network learns better weight to dynamically fuse the dual visual factors.

TABLE IV: Ablation study of the influence of each factor on TartanAir validation dataset.

Configuration	Mono ATE-scale(m)↓		Stereo ATE(m)↓	
	VO	SLAM	VO	SLAM
Baseline(B)	0.264	0.129	0.100	0.060
B + F-Metric_fixed $\boldsymbol{w}^f = 1$	23.78	23.98	14.58	14.59
B + F-Metric_tuned $\boldsymbol{w}^f$	0.190	0.115	0.078	0.048
B + F-Metric_learned $\boldsymbol{w}^f$	**0.137**$^{48.1\%\downarrow}$	**0.080**$^{38.0\%\downarrow}$	**0.069**$^{31.0\%\downarrow}$	**0.036**$^{40.0\%\downarrow}$

Figure 3 visualizes the average dual visual confidence variation with respect to the optimization iteration step when inference. We notice that the re-projection factor dominates at the early stage. When the pose and depth state approximate correct values, feature-metric confidence increases rapidly. After several iterations, the average confidence of feature-metric exceeds the re-projection factor a little bit and they jointly contribute to the optimization.

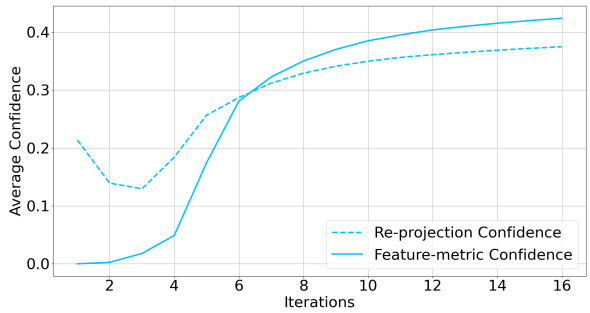

Fig. 3: The graph shows the change of average re-projection confidence and average feature-metric confidence with the number of iterations increasing.

Table V shows the impact of each factor on performance on the real-world EuRoC dataset. From the table, we observe that IMU factor has a larger impact on decreasing the trajectory error in comparison with the feature-metric factor. Specifically, we make experiments with different IMU weight. For example, for the method named "B + IMU_fixed $\boldsymbol{w}^u = 1$" , we only use the pre-integration weight $\boldsymbol{w}^{pu}$. For the method named "B + IMU_tuned $\boldsymbol{w}^u$" , we multiply a manually tuned vector on the pre-integration weight $\boldsymbol{w}^{pu}$. The results of this method approaches our learned performance. When dynamically combining the three factors together, the ATE of monocular and stereo VIO has reduced by 45.3% and 36.2% respectively.

TABLE V: Ablation study of the influence of each factor on the EuRoC dataset.

Configuration	Mono(VO/VIO) ATE-scale(m)↓	Stereo(VO/VIO) ATE(m)↓
Baseline(B)	0.179	0.080
B + F-Metric_fixed $\boldsymbol{w}^f = 1$	3.203	3.409
B + F-Metric_tuned $\boldsymbol{w}^f$	0.163	0.073
B + F-Metric_learned $\boldsymbol{w}^f$	0.143$^{20.1\%\downarrow}$	0.066$^{17.5\%\downarrow}$
B + IMU_fixed $\boldsymbol{w}^u = 1$	0.251	0.127
B + IMU_tuned $\boldsymbol{w}^u$	0.129	0.064
B + IMU_learned $\boldsymbol{w}^u$	0.114$^{36.3\%\downarrow}$	0.062$^{22.5\%\downarrow}$
B + F-Metric_learned $\boldsymbol{w}^f$ + IMU_learned $\boldsymbol{w}^u$	**0.098**$^{45.3\%\downarrow}$	**0.051**$^{36.2\%\downarrow}$

D. Memory and Runtime

Table VI shows the memory requirements of our VIO or VI-SLAM and runtime on different datasets with different resolution inputs. All tests are implemented with one NVIDIA RTX 3090 GPU. The runtime is the total runtime of VO or VIO divided by the total frame number of the sequence. In the table, we show the frames per second (fps), which is the inverse of the runtime.

TABLE VI: The memory requirement and runtime.

Dataset	Resolution	Memory		Runtime
		VO/VIO	V/VI-SLAM	
TartanAir [31]	384 × 512	12(G)	24(G)	8(fps)
EuRoC [32]	240 × 384	8(G)	24(G)	20(fps)
ETH3D-SLAM [33]	312 × 504	12(G)	24(G)	16(fps)

VI. Conclusions and Future Work

We present DVI-SLAM, a dual visual and inertial SLAM method that leverages dynamic multiple factors fusion to improve the SLAM accuracy. We verify that our DVI-SLAM method outperforms existing SLAM methods significantly in localization accuracy and robustness through extensive experiments. In the future, We would like to speed up the algorithm and reduce model memory requirements, for example by adjusting from dense to sparse tracking, which is an important step towards achieving real-time pose estimation on consumer-grade mobile processor platforms. In addition, in order to provide the accurate and complete 3D map, we also would like to better integrate RGB and depth sensors with Neural Radiance Fields (NeRFs) or 3D Gaussian Splatting methods.

Authorized licensed use limited to: ShanghaiTech University. Downloaded on September 11,2026 at 02:55:40 UTC from IEEE Xplore. Restrictions apply.

REFERENCES

[1] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox, "Demon: Depth and motion network for learning monocular stereo," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5038–5047.

[2] H. Zhou, B. Ummenhofer, and T. Brox, "Deeptam: Deep tracking and mapping," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 822–838.

[3] S. Shen, Y. Cai, W. Wang, and S. Scherer, "Dytanvo: Joint refinement of visual odometry and motion segmentation in dynamic environments," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 4048–4055.

[4] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, "Codeslam-learning a compact, optimisable representation for dense visual slam," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2560–2568.

[5] C. Tang and P. Tan, "Ba-net: Dense bundle adjustment network," *arXiv preprint arXiv:1806.04807*, 2018.

[6] J. Czarnowski, T. Laidlow, R. Clark, and A. J. Davison, "Deepfactors: Real-time probabilistic dense monocular slam," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 721–728, 2020.

[7] Z. Teed and J. Deng, "Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras," *Advances in neural information processing systems*, vol. 34, pp. 16 558–16 569, 2021.

[8] R. Clark, S. Wang, H. Wen, A. Markham, and N. Trigoni, "Vinet: Visual-inertial odometry as a sequence-to-sequence learning problem," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[9] L. Han, Y. Lin, G. Du, and S. Lian, "Deepvio: Self-supervised deep learning of monocular visual inertial odometry using 3d geometric constraints," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 6906–6913.

[10] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[11] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE, 2007, pp. 225–234.

[12] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE international conference on robotics and automation*. IEEE, 2007, pp. 3565–3572.

[13] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[14] K. Eckenhoff, P. Geneva, and G. Huang, "Closed-form preintegration methods for graph-based visual–inertial navigation," *The International Journal of Robotics Research*, vol. 38, no. 5, pp. 563–586, 2019.

[15] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[16] H. Liu, M. Chen, G. Zhang, H. Bao, and Y. Bao, "Ice-ba: Incremental, consistent and efficient bundle adjustment for visual-inertial slam," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1974–1982.

[17] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[18] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[19] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE transactions on robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[20] ——, "Visual-inertial monocular slam with map reuse," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 796–803, 2017.

[21] Z. Teed, L. Lipson, and J. Deng, "Deep patch visual odometry," *arXiv preprint arXiv:2208.04726*, 2022.

[22] W. Ye, X. Lan, S. Chen, Y. Ming, X. Yu, H. Bao, Z. Cui, and G. Zhang, "Pvo: Panoptic visual odometry," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 9579–9589.

[23] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 15–22.

[24] K. Eckenhoff, P. Geneva, and G. Huang, "Direct visual-inertial navigation with analytical preintegration," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1429–1435.

[25] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II 13*. Springer, 2014, pp. 834–849.

[26] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2017.

[27] J. Engel, J. Stückler, and D. Cremers, "Large-scale direct slam with stereo cameras," in *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2015, pp. 1935–1942.

[28] R. Wang, M. Schworer, and D. Cremers, "Stereo dso: Large-scale direct sparse visual odometry with stereo cameras," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3903–3911.

[29] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 2015, pp. 234–241.

[30] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2016.

[31] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "Tartanair: A dataset to push the limits of visual slam," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4909–4916.

[32] 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*, vol. 35, no. 10, pp. 1157–1163, 2016.

[33] T. Schops, T. Sattler, and M. Pollefeys, "Bad slam: Bundle adjusted direct rgb-d slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 134–144.

[34] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 573–580.

[35] Z. Teed and J. Deng, "Deepv2d: Video to depth with differentiable structure from motion," *arXiv preprint arXiv:1812.04605*, 2018.

[36] W. Wang, Y. Hu, and S. Scherer, "Tartanvo: A generalizable learning-based vo," in *Conference on Robot Learning*. PMLR, 2021, pp. 1761–1772.

[37] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2016.

[38] T. Qin, J. Pan, S. Cao, and S. Shen, "A general optimization-based framework for local odometry estimation with multiple sensors," *arXiv preprint arXiv:1901.03638*, 2019.

[39] S. Leutenegger, "Okvis2: Realtime scalable visual-inertial slam with loop closure," *arXiv preprint arXiv:2202.09199*, 2022.

[40] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt, "Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration," *ACM Transactions on Graphics (ToG)*, vol. 36, no. 4, p. 1, 2017.

[41] T. Whelan, S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison, "Elasticfusion: Dense slam without a pose graph." Robotics: Science and Systems, 2015.

[42] C. Kerl, J. Sturm, and D. Cremers, "Dense visual slam for rgb-d cameras," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 2100–2106.

[43] Y. Zhang, F. Tosi, S. Mattoccia, and M. Poggi, "Go-slam: Global optimization for consistent 3d instant reconstruction," *arXiv preprint arXiv:2309.02436*, 2023.

Authorized licensed use limited to: ShanghaiTech University. Downloaded on September 11,2026 at 02:55:40 UTC from IEEE Xplore. Restrictions apply.