2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

Multi-Parameter Optimization for a Robust RGB-D SLAM System*

Yizhao Wang[1], Xiaoxiao Zhu[1], Guohan He[1] and Qixin Cao[1,2]

Abstract— **SLAM systems can retrieve their metric scales and depth information using RGB-D cameras. However, limited by the sensing range and objects structure, RGB-D cameras can not always work well, resulting in failures sometimes. In this work, we present initialization and localization methods based on maximum-a-posteriori estimation. Our system endows monocular keypoints with valid depth values and introduce them into bundle adjustment. Depth bias coefficient and scale factor are also optimized in the local window, obtaining robustness in large scale environments and long-running operations. The experimental results indicate that our system provides the best robustness compared with other excellent methods in the literature, being able to process the most challenging sequences in the TUM RGB-D dataset.**

I. Introduction

Simultaneous Localization and Mapping (SLAM) techniques have been developed for robots and AR/VR devices to build a map of surrounding environment and estimate their localization in the map simultaneously. Compared with monocular case, visual SLAM in RGB-D configuration retrieves the true scale of the environment, contributing to AR/VR services provided by portable smart products embedding low-cost micro RGB-D cameras, such as smartphones and AR glasses.

However, most embedded micro RGB-D cameras can hardly obtain accurate depth measurements due to the restriction of their limited costs and sizes, especially Time-of-Flight (ToF) sensors proved fragile in depth discontinuities and dark surfaces [1]. The unstable and incomplete depth data may cause difficulty of initialization while locating and mapping with significant drift, being worse in large open environments.

Although current famed open-source SLAM systems for RGB-D sensors can achieve impressive accuracy tested by many popular datasets, they become fragile in regions where depth maps are incomplete and unstable. Specifically, we evaluated the open-source state-of-the-art methods ElasticFusion [2], ORB-SLAM2 [3] and its extension ORB-SLAM3 [4] on the *Robot SLAM* sequences in the TUM RGB-D dataset [5], as they contain poor texture, motion blur, strong rotations and severely incomplete depth data. All above SLAM systems got lost during the procedure, indicating the lack of robustness. Hence, extracting and optimizing valid depth information from the raw incomplete and unstable ones is the key of the superior SLAM accuracy and robustness when using low-cost RGB-D cameras or running in open areas where depth sensors do not work well.

In this work, we propose a novel robust RGB-D based initialization and localization method using nearest neighbor resampling (NNR) to retrieve invalid depth data and formulating the optimal pose estimation problem as *resampling BA* derived from maximum-a-posteriori (MAP) estimation. We build our system on ORB-SLAM [3] [4], the state-of-the-art work achieving excellent robustness and accuracy in most public datasets. The main contributions of our work are:

- We use NNR to render the keypoints without depth measurements the valid depth values with minor errors, obtaining as much information as possible to enhance tracking robustness.
- The depth bias and scale factor are introduced explicitly into the RGB-D optimization to reduce the accumulated drift in severe and long-term environments.
- Robust initialization and localization methods which formulate optimal pose estimation problem as *Depth-Scale BA* and *resampling BA* taking properly into account the depth bias, scale factor and features completed using depth NNR.
- A dataset containing a collection of handheld RGB-D sequences captured by a smartphone with a low-cost micro ToF sensor. Whereas this dataset is not accurate enough, it can be used to evaluate qualitative performance of RGB-D SLAM systems visually in scenes where depth maps are incomplete and unstable.

We discuss related work in the next section and then we describe our proposal and its details. After that we evaluate and compare our method with the state-of-the-art methods in the TUM RGB-D dataset and our author-collected dataset, demonstrating the excellent performance of our system.

II. Related Work

In the literature, DNA-SLAM of Wasenmüller et al. [6] proposed a reliability estimation for pixels integrating local depth derivatives into a weighting evaluation. This method integrated into DVO [7] performs a lower accuracy than keyframe-based algorithms though a longer running time. This work was extended by Ameli et al. [8] using GPU acceleration to achieve real-time performance. A direct dense RGB-D odometry was proposed by Gutierrez-Gomez et al. [9] which parameterizes geometric error by inverse depth to fit depth error model requiring GPU for real-time application. Proença et al. developed a probabilistic RGB-D odometry [10] using combination of points, planes and line segments, which models depth uncertainty and denoises by building a

*Research supported by the National Natural Science Foundation of China (61703273).

[1]State Key Laboratory of Mechanical System and Vibration, Shanghai Jiao Tong University, Shanghai, China.

[2]Institute of Medical Robotics, Shanghai Jiao Tong University, shanghai, China. wangyizhao@sjtu.edu.cn, ttl@sjtu.edu.cn, freewind@sjtu.edu.cn, qxcao@sjtu.edu.cn

depth fusion framework based on Mixtures of Gaussians. However, the lack of map optimization and place recognition makes its performance inferior to other keyframe-based approaches. BAD SLAM [11] proposed a direct bundle adjustment (BA) for RGB-D SLAM achieving excellent accuracy with GPU. However, this strategy is sensitive to rolling shutter and sensors synchronization. Thyagharajan et al. [12] presented a method using Map-Point Consensus based Outlier Rejection (MC-OR) to remove inconsistent features and Adaptive Virtual Camera (AVC) to correct depth measurement error, achieving excellent accuracy. Whereas the radical strategy to select features tends to exacerbate localization in areas with poor texture and severely incomplete depth information. To summarize the systems discussed above, most of them focus on selecting and optimizing the valid pixels or features from raw measurements to improve accuracy using formulated error models, neglecting scenes that contain severely incomplete depth data. Some algorithms without keyframes proved insufficient compared with keyframe-based ones, becoming the gold standard in visual SLAM [13]. Additionally, using GPU acceleration for real-time performance in some systems limited the application on portable smart products.

III. Method

Depth measurements not only benefit the system bootstrapping and creation of landmarks, but endow the map with metric scale. However, the performance of RGB-D cameras is impacted by environments [1]. When ORB-SLAM [3] [4] in RGB-D configuration processes in severe areas (e.g. *Robot SLAM* sequences in the TUM RGB-D dataset [5]), the number of keypoints with valid depth values extracted in per frame would be too insufficient to create an initial map or track the camera pose, resulting in tracking failure.

In this case, extracting adequate features with valid depth values from incomplete depth images is the key to the robust operation, which is based on the following ideas:

- According to the performance of low-cost consumer RGB-D cameras [1], except for regions beyond sensing range or on dark/glossy surfaces, most invalid depth measurements are introduced around depth or color discontinuities, where many ORB [14] features tend to be extracted especially in low-texture areas (Fig. 1).
- Most keypoints without valid depth values can be endowed with approximations using NNR, whose errors are not much more significant compared with the precision of low-cost RGB-D cameras disturbed by noises.
- The bias of depth measurements changes slowly along with the increase of running time and changes of environments [1]. During a short period we can regard it as a constant.
- Monocular SLAM builds accurate up-to-scale initial maps [15], which can be launched in the case of severe incompletion of depth maps. Then the few valid depth values are formulated as *Depth-Scale BA* where scale factor is parameterized explicitly to accelerate its convergence [16] inspired by the inertial case in [17].

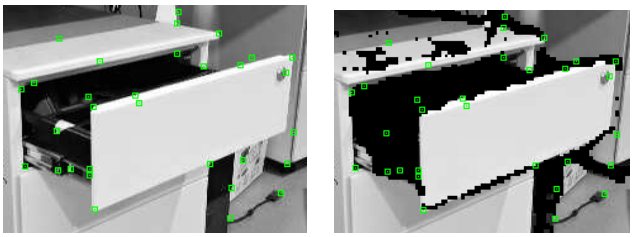


Fig. 1. ORB extraction (left) and the fusion with depth map (right) recorded by HUAWEI P30 Pro where dark areas indicate invalid depth values.

Based on these ideas the pipeline of our method can be split as follows:

1) **Depth nearest neighbor resampling**: Endow features without valid depth information with values obtained from their nearest neighbors which we call the *resampling keypoints* to add robustness of tracking.
2) **Depth-Scale BA**: Our system tries to initialize in RGB-D case first. If the initialization fails for a while, we initialize and run pure monocular SLAM [15] for a while to obtain an up-to-scale initial map whose metric scale can be retrieved later using corresponding depth data. After this period, we perform full BA optimization.
3) **Resampling BA**: After initialization, we perform *resampling motion-only BA* in the tracking thread and *resampling local BA* in the local mapping thread, similar to ORB-SLAM2 [3], where the main difference is that we perform resampling motion-only BA using all information in resampling keypoints but resampling local BA not. Both of them take into account the depth bias coefficient as a parameter to optimize.

All these optimization problems are carried out and solved by g2o [18], a C++ framework for nonlinear graph optimization. The details of our proposal are discussed next.

A. Depth Nearest Neighbor Resampling

Our system tries to run in RGB-D configuration and extract ORB per frame similar to [3]. For each feature with a valid depth value, we generate the stereo keypoint $\mathbf{x}_\mathrm{s} \in \mathbb{R}^3$ as proposed in [19]. As for features with invalid depth values, we formulate a nearest neighbor around each keypoint and perform RANSAC iterations for valid depth values. If a depth value with enough inliers is found, we assign it to the original keypoint and generate the *resampling keypoint* $\mathbf{x}_\mathrm{r} \in \mathbb{R}^3$ using the same procedure as the stereo keypoint. Otherwise, in the case of the shortage of inliers, we generate the monocular keypoint $\mathbf{x}_\mathrm{m} \in \mathbb{R}^2$ following [3] as presented in Fig. 4.

It is worth noting that although resampling keypoints are disturbed by errors due to depth NNR, as shown in the experiments of Section IV-B, most of them provide depth values quite close to corresponding expectations which could be recorded by depth sensors. Whereas depth NNR focuses on expanding features with valid depth values, we just use resampling keypoints in the tracking thread to improve robustness when depth maps are seriously incomplete.

B. Depth-Scale BA

SLAM in RGB-D configuration can initialize from just one frame with depth measurements [3], but it tends to fail in the case of the shortage of stereo and resampling keypoints due to severely incomplete depth data. To obtain robustness against this situation, after several failures of RGB-D initialization, our system will initialize and run monocular SLAM [15] for a few seconds to construct and optimize the up-to-scale initial map. This map contains n keyframes whose poses are denoted as $\bar{\mathbf{T}}_{1:n} = [\mathbf{R}, \bar{\mathbf{t}}]_{1:n}$, where $\mathbf{R}_k \in \mathrm{SO}(3)$ and $\bar{\mathbf{t}}_k \in \mathbb{R}^3$ are the body orientation and position of the keyframe k, and the bar denotes up-to-scale variables. To capture more valid information, keyframes are inserted more frequently than ORB-SLAM in this step [17]. The span of monocular SLAM depends on the amount of stereo and resampling keypoints, which should accumulate enough for later optimization.

Once the initial map contains enough stereo/resampling keypoints, the *Depth-Scale BA* is performed to retrieve the state parameters $\mathcal{Y} = \{s, \mathbf{b}\}$ where $s \in \mathbb{R}^+$ is the scale factor of the up-to-scale map, and $\mathbf{b} = (b_0, b_1) \in \mathbb{R}^2$ is the bias coefficient of the depth sensor to compensate the drift of depth pixels. Although $\mathbf{b}$ depends on many factors [1], we assume that it is constant during a short period.

This optimization problem involves the set of depth measurements $\mathcal{D}_{\mathcal{X}_{1:n}} \doteq \{\mathcal{D}_{\mathcal{X}_1}, ..., \mathcal{D}_{\mathcal{X}_n}\}$, where $\mathcal{D}_{\mathcal{X}_k}$ is the set of depth measurements of the stereo/resampling keypoints in the keyframe k with matched 3D points, and $\mathcal{X}_k$ is the set of all matches between map points and stereo/resampling keypoints in the keyframe k. Hence, the posterior distribution of the MAP estimation problem can be formulated as:

$$p(\mathcal{Y} \mid \mathcal{D}_{\mathcal{X}_{1:n}}) \propto p(\mathcal{D}_{\mathcal{X}_{1:n}} \mid \mathcal{Y}) p(\mathcal{Y}) \tag{1}$$

where $p(\mathcal{D}_{\mathcal{X}_{1:n}} \mid \mathcal{Y})$ denotes the likelihood distribution of the depth measurements based on the depth states and $p(\mathcal{Y})$ denotes the prior of the depth states. Considering measurements are independent, the likelihood can also be written as:

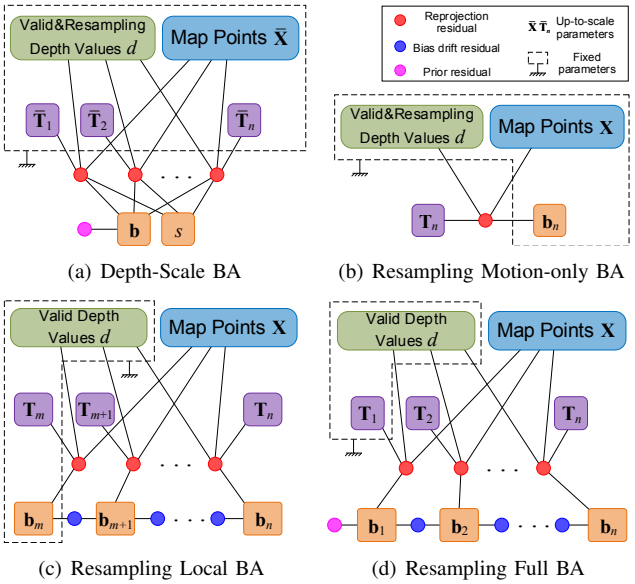
$$p\left(\mathcal{D}_{\mathcal{X}_{1:n}} \mid \mathcal{Y}\right) = \prod_{k=1}^{n} \prod_{j \in \mathcal{X}_k} p\left(\mathcal{D}_{kj} \mid \mathcal{Y}\right) \tag{2}$$

where $\mathcal{D}_{kj} \in \mathcal{D}_{\mathcal{X}_k}$ stands for the depth measurement of the stereo/resampling keypoint $\mathbf{x}_{(\cdot)}^{kj} \in \mathbb{R}^3$ in the keyframe k. Then the MAP estimation is transformed by taking negative logarithm and its optimal solution can be obtained as follows:

$$\begin{aligned} \mathcal{Y}^* &= \underset{\mathcal{Y}}{\operatorname{argmax}}\, p(\mathcal{Y} \mid \mathcal{D}_{\mathcal{X}_{1:n}}) = \underset{\mathcal{Y}}{\operatorname{argmax}} \left(p(\mathcal{Y}) \prod_{k=1}^{n} \prod_{j \in \mathcal{X}_k} p(\mathcal{D}_{kj} \mid \mathcal{Y}) \right) \\ &= \underset{\mathcal{Y}}{\operatorname{argmin}} \left(-\log\left(p\left(\mathcal{Y}\right)\right) - \sum_{k=1}^{n} \sum_{j \in \mathcal{X}_k} \log\left(p\left(\mathcal{D}_{kj} \mid \mathcal{Y}\right)\right) \right) \end{aligned} \tag{3}$$

Assuming Gaussian error for depth measurements likelihood and prior distribution, the optimization problem can be stated as:

$$\mathcal{Y}^* = \underset{\mathcal{Y}}{\operatorname{argmin}} \left(\|\mathbf{r}_{\mathrm{p}}\|_{\Sigma_p}^2 + \sum_{k=1}^{n} \sum_{j \in \mathcal{X}_k} \rho_{\mathrm{Hub}}\left(\|\mathbf{r}_{kj}\|_{\Sigma_{kj}} \right) \right) \tag{4}$$



(a) Depth-Scale BA (b) Resampling Motion-only BA

(c) Resampling Local BA (d) Resampling Full BA

Fig. 2. Factor graph representations for different optimizations.

where $\mathbf{r}_{\mathrm{p}}$ and $\mathbf{r}_{kj}$ are the residuals of the prior and depth measurement of stereo/resampling keypoints, being Σ_p and Σ_{kj} their covariances. ρ_{Hub} is the robust Huber kernel to reduce the impact of data association errors.

As minimizing the reprojection errors is the gold standard in feature-based SLAM, the residual $\mathbf{r}_{kj}$ is defined as:

$$\mathbf{r}_{kj} = \mathbf{x}_{(\cdot)}^{kj} - \pi_{(\cdot)}\left(\mathbf{R}_k\left(s\bar{\mathbf{X}}^{kj}\right) + s\bar{\mathbf{t}}_k\right) \tag{5}$$

where $\bar{\mathbf{X}}^{kj} \in \mathbb{R}^3$ stands for the up-to-scale map point matched with the stereo/resampling keypoint $\mathbf{x}_{(\cdot)}^{kj} \in \mathbb{R}^3$, and $\pi_{(\cdot)} \colon \mathbb{R}^3 \to \mathbb{R}^3$ is the projection function. Considering bias $\mathbf{b}$ of depth measurements, $\mathbf{x}_{(\cdot)}^{kj}$ is formulated as:

$$\mathbf{x}_{(\cdot)}^{kj} = \left(\mathbf{x}'^{kj}_{(\cdot)}, u_R^{kj}\right) = \left(u_L^{kj}, v_L^{kj}, u_L^{kj} - \frac{f_x b}{b_1 d^{kj} + b_0}\right) \tag{6}$$

where $\mathbf{x}'^{kj}_{(\cdot)} = \left(u_L^{kj}, v_L^{kj}\right) \in \mathbb{R}^2$ is the keypoint extracted in monocular SLAM with a valid or resampling depth value d^{kj} which is neglected until Depth-Scale BA is performed.

The updates on the state parameters are in different means. The scale factor s is updated as $s^{\mathrm{new}} = s^{\mathrm{old}} \exp(\delta s)$ to maintain it positive during optimization [17], whereas the bias coefficient $\mathbf{b}$ is updated additively.

This optimization can be represented as the factor graph shown in Fig. 2(a) and carried out by g2o [18]. In this case, a good initial seed is essential for the quick convergence to avoid local minima. Hence, the state parameters are initialized as:

$$s^{\mathrm{init}} = \left(\sum_{k=1}^{n} \sum_{j \in \mathcal{X}_k} \frac{d^{kj}}{\left\| \bar{\mathbf{X}}^{kj} - \bar{\mathbf{t}}_k \right\|} \right) \Bigg/ \left(\sum_{k=1}^{n} |\mathcal{X}_k| \right) \tag{7}$$

being s^{init} the average of all ratios between valid/resampling depth values and up-to-scale distances from corresponding camera centers to matched 3D points, and $\mathbf{b}^{\mathrm{init}} = (0, 1)$, based on the prior assumption of an accurate initial map and

consistent depth measurements during a short period. Thus, the prior residual can be defined as $\mathbf{r}_\mathrm{p} = \mathbf{b} - \mathbf{b}^\mathrm{init}$, which is considered to be as close as possible to the initial value.

Once our system finishes the optimization, the up-to-scale map and body pose are scaled by the estimated scale. Then *resampling full BA* is performed which is explained next.

C. Resampling BA

After a successful initialization, our system performs BA in both tracking and local mapping threads based on MAP estimation. The main difference from [3] is that the resampling keypoints and depth bias are introduced to perform BA, which we call the *resampling BA*. Note that the characterization of resampling keypoints switches between tracking and local mapping threads.

a) Resampling Motion-only BA: In the tracking thread, our system performs motion-only BA per frame minimizing the reprojection error (Fig. 2(b)). The current body pose is the only variable to optimize, keeping fixed the other parameters and $\mathbf{b}_n = \mathbf{b}_{n-1}$ obtained from previous BA. Considering different dimensions of keypoints, the projection functions $\pi_{(\cdot)}$ are defined respectively that monocular $\pi_\mathrm{m} \colon \mathbb{R}^3 \to \mathbb{R}^2$ while stereo $\pi_\mathrm{s} \colon \mathbb{R}^3 \to \mathbb{R}^3$.

Although resampling keypoints $\mathbf{x}_\mathrm{r}$ may contain minor errors due to depth NNR, they can provide more information for data association. Hence, $\mathbf{x}_\mathrm{r}$ and their matched map points are introduced into motion-only BA enhancing robustness in case tracking gets lost. In fact, motion-only BA requires robustness more than accuracy since its goal is to track from last frame and compute an initial body pose for further refinement in the local mapping thread. Note that we formulate the resampling projection function as $\pi_\mathrm{r} \colon \mathbb{R}^3 \to \mathbb{R}^3$ using the same way as stereo π_s.

b) Resampling Local BA: After a new keyframe is inserted to the local mapping thread, our system performs local BA to minimize the reprojection error in a local window (Fig. 2(c)), based on the initial solution in the tracking thread.

For the same reason as resampling motion-only BA, we introduce $\mathbf{x}_\mathrm{r}$ and matched points into local BA. Since this step aims to obtain the optimal estimation of the body pose, we neglect the virtual right coordinate of the resampling keypoint with minor depth error and redefine it as $\mathbf{x}'_\mathrm{r} \in \mathbb{R}^2$ which can be regarded as a monocular $\mathbf{x}_\mathrm{m}$ in this step. To match the dimension of $\mathbf{x}'_\mathrm{r}$, thus, we reformulate the resampling projection function as $\pi'_\mathrm{r} \colon \mathbb{R}^3 \to \mathbb{R}^2$ in the form of π_m. In this way, the impact of minor error caused by depth NNR can be eliminated in resampling local BA. Thus, the state parameters with regard to the current keyframe n are defined as:

$$\mathcal{S}_n = \{\mathbf{X}^i, \mathbf{R}_l, \mathbf{t}_l, \mathbf{b}_l | i \in \mathcal{P}_L, l \in \mathcal{K}_L\} \tag{8}$$

where $\mathcal{K}_L$ are covisible keyframes and $\mathcal{P}_L$ are map points seen in $\mathcal{K}_L$ remaining the rest keyframes $\mathcal{K}_F$ observing $\mathcal{P}_L$ fixed. Then we state the MAP estimation problem where the posterior distribution is:

$$p(\mathcal{S}_n \mid \mathcal{X}_{\mathcal{K}_L}) \propto p(\mathcal{X}_{\mathcal{K}_L} \mid \mathcal{S}_n) p(\mathcal{S}_n) \tag{9}$$

where $\mathcal{X}_{\mathcal{K}_L}$ are all matches including corresponding depth measurements observed by covisible keyframes $\mathcal{K}_L$. Similar to the solution of Depth-Scale BA, this optimization problem can be derived as:

$$\mathcal{S}_n^* = \arg\min_{\mathcal{S}_n} \Bigg(\sum_{k \in \mathcal{K}_L \cup \mathcal{K}_F} \left\| \mathbf{r}_\mathrm{b}^k \right\|^2_{\Sigma_b} + \sum_{k \in \mathcal{K}_L \cup \mathcal{K}_F} \sum_{j \in \mathcal{X}_k} \rho_\mathrm{Hub} \left(\left\| \mathbf{r}_{kj} \right\|_{\Sigma_{kj}} \right) \Bigg) \tag{10}$$

where $\mathbf{r}_\mathrm{b}^k = \mathbf{b}^k - \mathbf{b}^{k-1}$ is the bias drift residual indicating that the bias coefficient of the depth sensor should be close to that of the last keyframe, since the span between them is short. The reprojection error $\mathbf{r}_{kj}$ is defined similarly to (5), where the products of s and up-to-scale variables are replaced by those with metric scale, and $\mathbf{b}$ in the virtual right coordinate of stereo $\mathbf{x}_\mathrm{s}^{kj}$ is replaced by $\mathbf{b}^k$.

c) Resampling Full BA: This step is launched to further optimize the map after a loop correction. It extends resampling local BA to all keyframes and map points in the map taking into account the prior residual $\mathbf{r}_\mathrm{p}$ (Fig. 2(d)).

Note that resampling full BA is also performed after Depth-Scale BA. In this case, all reprojection errors are formulated using the common bias coefficient $\mathbf{b}$ obtained from previous Depth-Scale BA, since $\mathbf{b}$ is assumed to be constant during a short period. Thus, the prior residuals $\mathbf{r}_\mathrm{b}^k$ are replaced by $\mathbf{r}_\mathrm{p}$ to constrain the update of $\mathbf{b}$.

IV. Experiments

To evaluate the performance of our system in different environments, we run our system on both the TUM RGB-D dataset [5] and the author-collected dataset. The TUM RGB-D dataset consists of indoors sequences captured by three depth sensors respectively. We also propose the author-collected dataset recorded by a smartphone with a low-cost embedded depth camera to evaluate our system. The results show that our proposal outperforms the state-of the-art methods. All experiments were run in an Intel Core i7-8750H CPU, at 2.2GHz, with 8GB memory.

A. Author-collected Dataset

The author-collected dataset was recorded by HUAWEI P30 Pro, a smartphone with a low-cost embedded micro ToF sensor. We fixed the smartphone on ZEB-REVO RT, a high precision portable laser scanner. We proceeded indoors with the handheld device, as shown in Fig. 3, and regarded the estimated results from the laser scanner as the groundtruth, since its accuracy is much higher than the visual SLAM. Limited by the crude experimental equipment, the evaluation results can not mean a lot. However, the qualitative results can still present the performance of SLAM methods visually by plotting trajectories. We have recorded ten sequences in different scenes including studios, malls and laboratories, most of whose trajectories are longer than 200m. Due to space limitations, we will publish this dataset and its detailed descriptions later.

Fig. 3. HUAWEI P30 Pro with ZEB-REVO RT portable laser scanner.

TABLE I

PROPORTION (%) THAT MAD AND MRD ARE LOWER THAN CERTAIN VALUES, USING DEPTH NEAREST NEIGHBOR RESAMPLING FOR STEREO KEYPOINTS EXTRACTED IN DIFFERENT SEQUENCES.

Seq.*	MAD (mm)			MRD (%)		
	10	20	40	1	2	4
fr1	45.79	75.33	91.72	54.81	82.16	93.21
fr2	25.89	31.98	64.15	30.44	58.36	89.47
fr3	45.61	73.43	92.54	62.57	85.16	96.57

* *fr1* includes *fr1_desk*, *fr1_desk2* and *fr1_room*. *fr2* includes *fr2_pioneer_360*, *fr2_pioneer_slam*, *fr2_pioneer_slam2* and *fr2_pioneer_slam3*. *fr3* includes *fr3_office* and *fr3_nst*.

B. Depth Nearest Neighbor Resampling Performance

To analyze the performance of depth NNR, we perform depth NNR with radius of 5 pixels on each stereo keypoint. Table I presents the mean absolute deviation (MAD) and mean relative deviation (MRD) of resampling values compared with the practical values. In TUM/*freiburg1* and *freiburg3* sequences, almost half of the resampling values obtain high accuracy (MAD <10mm, MRD <1%) and over 90% values achieve acceptable accuracy (MAD <40mm, MRD <4%). Whereas the MAD in TUM/*freiburg2* sequences is larger than others due to the large depth in open scenes, almost 90% values provide acceptable accuracy that MRD <4%. Compared with the evaluation of Microsoft Kinect [1], the error caused by depth NNR is acceptable for tracking which requires more robustness than accuracy.

Table II provides a quantitative comparison with regard to different types of keypoints. We extract 1000 ORB features per image, and the keypoints with matched 3D points are presented in Fig. 4. The first block compares the amounts of map points matched with different types of keypoints extracted per frame in tracking. In TUM/*freiburg1* and *freiburg3* sequences, the amounts matched with stereo keypoints are much more than those from resampling and monocular ones. Whereas in TUM/*freiburg2* sequences and the author-collected dataset, the amounts matched with stereo keypoints drop significantly due to the poor performance of depth sensors. The last block compares the matching rates between keypoints and 3D points. The stereo and resampling keypoints provide similar matching rates almost two times that of monocular ones, showing that resampling keypoints are equivalent to stereo keypoints in tracking. As shown in Fig. 4, resampling keypoints tend to be created at edges of objects which are essential for SLAM. Thus, we can take advantage of the resampling keypoints in tracking to get more robustness.

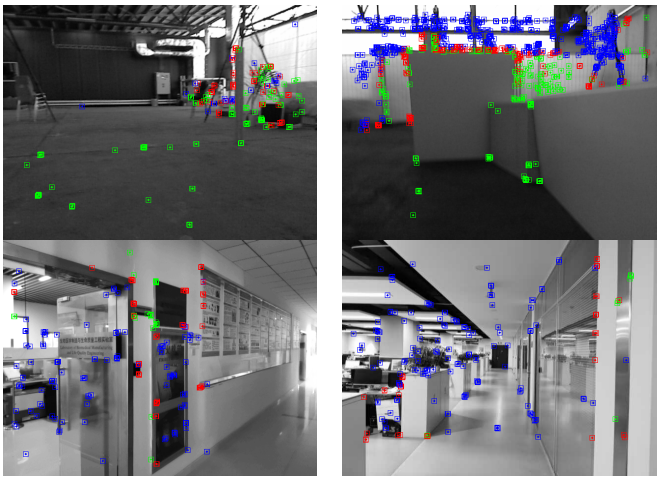


Fig. 4. Stereo keypoints (green), monocular keypoints (blue) and resampling keypoints (red) extracted per frame with matched map points. Top: TUM RGB-D dataset *fr2_pioneer_360* and *fr2_pioneer_slam*, bottom: author-collected dataset *lab* and *studio*.

TABLE II

AMOUNTS OF MAP POINTS MATCHED WITH KEYPOINTS EXTRACTED PER FRAME AND CORRESPONDING MATCHING RATES (%)

Seq.*	Amounts			Matching Rates (%)		
	Ster.	Resam.	Mono.	Ster.	Resam.	Mono.
fr1	222.15	15.83	5.97	25.85	18.71	9.73
fr2	91.89	16.47	13.21	14.45	17.16	5.94
fr3	345.07	20.41	1.89	36.06	33.62	23.19
Ours	35.18	17.69	150.77	27.76	24.32	18.18

* *fr1*, *fr2* and *fr3* are the same as in Table I. *Ours* stands for author-collected dataset *studio*, *lab* and *mall*.

C. Evaluation in Datasets

a) TUM RGB-D Dataset: We evaluate our method in not only the sequences where most literature processed their RGB-D systems, but the *Robot SLAM* sequences from *freiburg2* which contain strong rotation, motion blur, missing depth data and significant gaps where systems tend to get lost. We choose the median of five executions as the results to compare with other state-of-the-art methods in Table III. Our method present a competitive performance in most sequences, especially in large scale areas.

In *freiburg1* and *freiburg3* sequences, our system provides a similar accuracy to ORB-SLAM2. Noise-resilient SLAM [12] achieves a slight superiority to all others. In small scale environments, most regions are within the valid sensing range of RGB-D cameras where depth information can be recorded completely, and their biases caused by temperature rise can be neglected during a short period. Thus, the advantages of our method are not significant in these sequences.

In *freiburg2/Robot SLAM* sequences, our system is vastly superior to others as shown in Fig. 5. ORB-SLAM2 gets lost

TABLE III
PERFORMANCE COMPARISON IN TUM RGB-D DATASET BY RMSE ATE (m).

Seq.	ORB-SLAM2 [3]	Elastic-Fusion [2][1]	Noise-resilient SLAM [12][1]	Ours
fr1/desk	0.016	0.020	**0.015**	**0.015**
fr1/desk2	**0.022**	0.048	**0.022**	0.024
fr1/room	0.056	0.068	0.043	**0.041**
fr2/p_360[2]	X[3]	X	-[4]	**0.047**
fr2/p_slam3	X	X	-	**0.056**
fr3/office	0.011	0.017	**0.008**	0.009
fr3/nst	0.022	**0.016**	0.018	0.024

[1] We provide the values reported at their literature.
[2] p indicates pioneer.
[3] X indicates failure cases.
[4] - indicates results were not published.

TABLE IV
COMPUTING TIME OF OUR METHOD COMPARED WITH ORB-SLAM2.

Seq.	Tracking (ms)		Local Mapping (ms)	
	ORB-SLAM2	Ours	ORB-SLAM2	Ours
fr1/desk	24.80	29.14	196.61	263.72
fr1/desk2	25.36	29.48	164.63	228.38
fr1/room	23.15	27.92	154.89	195.96
fr2/p_360	X	24.38	X	168.59
fr2/p_slam3	X	25.94	X	176.99
fr3/office	27.21	31.05	279.80	322.15
fr3/nst	20.64	24.76	107.57	163.24

after a strong rotation or a rough motion blur, since it can not extract and match enough stereo keypoints. Additionally, We evaluate ORB-SLAM3 [4] in these challenging sequences. However, the novel multi-map capability of ORB-SLAM3 makes itself stuck in repetition of new map creation instead of expected map merging, probably due to the lack of loops along the trajectories. Our method can obtain competitive accuracy in most scenes, whereas a significant error appears in the last of *fr2_pioneer_slam3* sequence due to the strong rotation. We also evaluate our method in the synthetic dataset [11] created from the TUM RGB-D Dataset. Unlike BAD SLAM [11] obtaining higher accuracy, our system improves little in the synthetic dataset, perhaps since synthetic datasets are more suitable for direct methods as shown in [11].

Moreover, we compare the computing time of our method with ORB-SLAM2 as presented in Table IV. Although our system takes more time in tracking (∼5ms) and local mapping (∼50ms) due to the additional optimization, it can runs robustly in real-time on standard CPUs.

b) Author-collected Dataset: We also evaluate our system in the author-collected dataset and compare with ORB-SLAM2 in Table V. Our method is about 10 times more accurate than ORB-SLAM2 in RGB-D case. Actually, the severely incomplete depth maps can not provide sufficient stereo keypoints to retrieve metric scale. Moreover, the temperature of the embedded micro ToF sensor rises rapidly, resulting in the increasing bias of depth measurements. Thus, severe scale drift is accumulated during the procedure. In contrast, our system can track and retrieve metric scale with robustness using depth NNR and resampling BA to achieve higher accuracy as shown in Fig. 5.

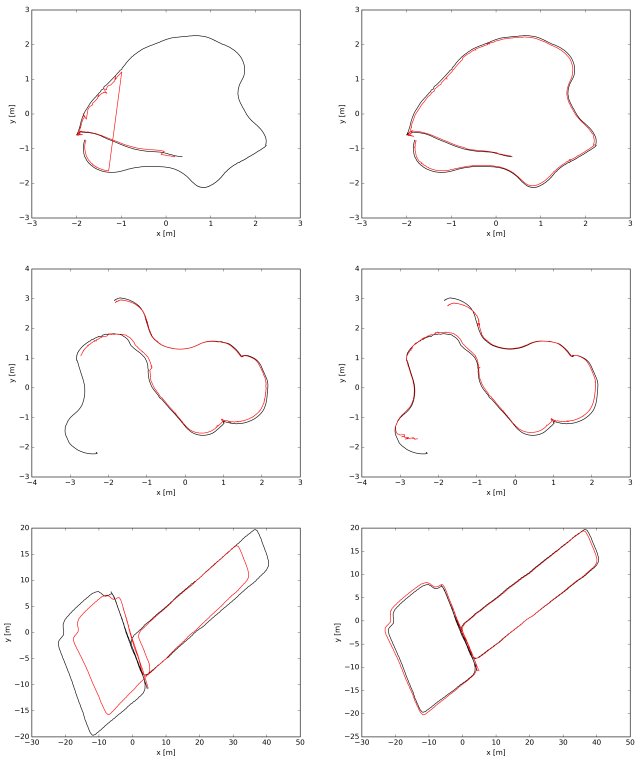


Fig. 5. Estimated trajectory (red) and groundtruth (black) in TUM RGB-D dataset *fr2_pioneer_360*, *fr2_pioneer_slam3* and author-collected dataset *studio*. Left: ORB-SLAM2 (RGB-D), right: our method.

TABLE V
PERFORMANCE COMPARISON IN AUTHOR-COLLECTED DATASET BY RMSE ATE (m).

Seq.	studio	mall	lab	garage	market
ORB-SLAM2	3.724	3.929	0.549	5.013	5.187
Ours	**0.448**	**0.273**	**0.058**	**0.879**	**0.951**

V. CONCLUSIONS

We propose RGB-D initialization and localization methods with impressive performance. Resampling motion-only BA considers the depth of resampling keypoints generated by depth NNR to enhance robustness in the tracking thread. Resampling local BA introduces depth bias coefficient into the optimization problem, contributing to improved pose estimation. Depth-Scale BA optimize both depth bias coefficient and scale factor based on the up-to-scale initial map in the case of RGB-D initialization failure in severe areas.

The experimental results in TUM RGB-D dataset reveal that our system provides excellent robustness in severe environments. We also realize that the errors introduced from depth NNR still slightly impact the performance of our system. Thus, how to obtain accurate depth completion with low computational cost is the direction of higher accuracy.

REFERENCES

[1] O. Wasenmüller and D. Stricker, "Comparison of kinect v1 and v2 depth images in terms of accuracy and precision," in *Computer Vision – ACCV 2016 Workshops* (C.-S. Chen, J. Lu, and K.-K. Ma, eds.), (Cham), pp. 34–45, Springer International Publishing, 2017.

[2] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "Elasticfusion: Real-time dense slam and light source estimation," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1697–1716, 2016.

[3] 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.

[4] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual-inertial and multi-map slam," 2020.

[5] 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*, pp. 573–580, 2012.

[6] O. Wasenmüller, M. D. Ansari, and D. Stricker, "Dna-slam: Dense noise aware slam for tof rgb-d cameras," in *Computer Vision – ACCV 2016 Workshops* (C.-S. Chen, J. Lu, and K.-K. Ma, eds.), (Cham), pp. 613–629, Springer International Publishing, 2017.

[7] 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*, pp. 2100–2106, 2013.

[8] M. Ameli, O. Wasenmüller, M. R. Soheili, J. Shanbehzadeh, and D. Stricker, "Accelerated dna-slam for rgb-d images," in *Proceedings of the 2018 International Conference on Image and Graphics Processing*, ICIGP 2018, (New York, NY, USA), p. 185–190, Association for Computing Machinery, 2018.

[9] D. Gutierrez-Gomez, W. Mayol-Cuevas, and J. Guerrero, "Dense rgb-d visual odometry using inverse depth," *Robotics and Autonomous Systems*, vol. 75, pp. 571 – 583, 2016.

[10] P. F. Proença and Y. Gao, "Probabilistic rgb-d odometry based on points, lines and planes under depth uncertainty," *Robotics and Autonomous Systems*, vol. 104, pp. 25 – 39, 2018.

[11] T. Schöps, T. Sattler, and M. Pollefeys, "Bad slam: Bundle adjusted direct rgb-d slam," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 134–144, 2019.

[12] A. Thyagharajan, O. J. Omer, D. Mandal, and S. Subramoney, "Towards noise resilient slam," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 72–79, 2020.

[13] H. Strasdat, J. Montiel, and A. J. Davison, "Visual slam: Why filter?," *Image and Vision Computing*, vol. 30, no. 2, pp. 65 – 77, 2012.

[14] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International Conference on Computer Vision*, pp. 2564–2571, 2011.

[15] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "Orb-slam: A versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[16] H. Strasdat, J. Montiel, and A. J. Davison, "Scale drift-aware large scale monocular slam," *Robotics: Science and Systems VI*, vol. 2, no. 3, p. 7.

[17] C. Campos, J. M. M. Montiel, and J. D. Tardós, "Inertial-only optimization for visual-inertial initialization," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 51–57, 2020.

[18] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *2011 IEEE International Conference on Robotics and Automation*, pp. 3607–3613, 2011.

[19] H. Strasdat, A. J. Davison, J. M. M. Montiel, and K. Konolige, "Double window optimisation for constant time visual slam," in *2011 International Conference on Computer Vision*, pp. 2352–2359, 2011.