Multimed Tools Appl
DOI 10.1007/s11042-016-3524-x

Semi-direct tracking and mapping with RGB-D camera for MAV

Shuhui Bu[1] · Yong Zhao[1] · Gang Wan[2] · Ke Li[2] · Gong Cheng[1] · Zhenbao Liu[1]

Received: 31 October 2015 / Revised: 20 March 2016 / Accepted: 7 April 2016

Abstract In this paper we present a novel semi-direct tracking and mapping (SDTAM) approach for RGB-D cameras which inherits the advantages of both direct and feature based methods, and consequently it achieves high efficiency, accuracy, and robustness. The input RGB-D frames are tracked with a direct method and keyframes are refined by minimizing a proposed measurement residual function which takes both geometric and depth information into account. A local optimization is performed to refine the local map while global optimization detects and corrects loops with the appearance based bag of words and a co-visibility weighted pose graph. Our method has higher accuracy on both trajectory tracking and surface reconstruction compared to state-of-the-art frame-to-frame or frame-model approaches. We test our system in challenging sequences with motion blur, fast pure rotation, and large moving objects, the results demonstrate it can still successfully obtain results with high accuracy. Furthermore, the proposed approach achieves real-time speed which only uses part of the CPU computation power, and it can be applied to embedded devices such as phones, tablets, or micro aerial vehicles (MAVs).

Keywords RGB-D SLAM · Localization · Tracking · Mapping · Reconstruction · Real-time

✉ Zhenbao Liu
liuzhenbao@nwpu.edu.cn

Shuhui Bu
bushuhui@nwpu.edu.cn

Yong Zhao
zd5945@126.com

[1] Northwestern Polytechnical University, 710072 Xi'an, China

[2] Information Engineering University, 450000 Zhengzhou, China

Published online: 22 April 2016

1 Introduction

With the rapid development of computer techniques, multimedia applications have been extensively used into our daily life. Most applications are based on images [19–23, 43, 44, 50], however we live in a 3D space, as a consequence using 3D data [2–5, 29] to represent the world is intuitively better than just using 2D images which are just projections of the 3D world. In recent years, simultaneous localization and mapping (SLAM) is one of the hot researches in robotics and computer vision community. The information of workspace map and robot location is essential for unmanned micro aerial vehicle (MAV) or any other robot to perform tasks in unknown environments. Flighting MAVs in unknown indoor environments, for example, demand real-time poses and orientation information for obstacle avoidance, motion planning, and navigation [9].

A series of sensors have been explored for this problem in the last few years, including monocular cameras, 2D laser scanners, and RGB-D sensors like time of flight (TOF) cameras or Microsoft Kinect [7, 9, 10, 16–18, 30, 51]. The mature technique is Light Detection and Ranging (LiDAR), but it only exploits 2D barriers information [16]. It best fits in environments characterized by vertical structures, but may fail in complex scenes, since most LiDARs only detect barriers that intersect the sensing plane. RGB-D sensors which provide both color and depth images directly have become a popular choice for dense reconstruction of unknown indoor environments. TOF cameras are quite expensive which baffles the wide application, nevertheless Microsoft Kinect offers a valuable alternative way, since it provides dense and high-frequency depth information with a low price, size, and weight.

In this paper, we propose a novel RGB-D based tracking and mapping system, which combines direct and feature based methods together seamlessly to improve the performances. Most feature based methods minimize re-projection pixel errors, however, they suffer from the absence of depth information. In order to improve the robustness, a novel measurement error function is proposed to appropriately mix depth and geometric information. Compared with state-of-the-art works, our method achieves higher accuracy and robustness just using CPU, which has great potential to be applied to embedded devices. In brief, there are three main contributions as follows:

1. To efficiently exploit captured information, the direct method is adopted to track current motion with high-speed, followed with a motion refinement based on feature correspondences. Therefore, an elegant balance between accuracy and efficiency can be realized.
2. A novel error function based on mixed depth and geometric measurements is designed to achieve high accuracy and robustness for pose estimation.
3. We create a large scale dataset which consists of several long trajectories. In addition, the dataset contains fast motion, repetitive scenes, and even lost depth information. The dataset is public available on a website, and therefore, researches can use it to evaluate their methods on this dataset.

Extensive experiments are conducted to evaluate the performances of the proposed approach. Several quantitative evaluations are presented in Section 4, which contain not only the trajectory and reconstruction quality but also computational performances. From the experiments, we can conclude that the proposed method can achieve superior performance.

2 Related work

A substantial number of works for camera pose optimization using RGB-D data have been published over the past few years. We classify them into following categories.

Feature based methods The first RGB-D SLAM is explored to estimate the pose of Kinect and reconstruct the indoor environment by Henry et al. [25]. They extract scale invariant feature transformation (SIFT) features [31] by SIFTGPU [49] from color images and match them to previous keyframes. Random sample consensus (RANSAC) is applied to refining these matches and then computing an initial transformation which is refined again using RGBD-ICP based on iterative closest point (ICP) algorithm [36]. Loop closure is detected with both color and depth information, and a sparse bundle adjustment method [45] is adopted to achieve global consistency. A similar method is proposed by Endres et al. [8]. Rather than only SIFT, other features including Speeded up robust features (SURF) [1] and Oriented FAST and Rotated BRIEF (ORB) [35] are also used, g2o solver [15] instead of bundle adjustment is applied to accomplishing global optimization.

Direct methods Newcombe et al. propose a frame-model solution, KinectFusion [34], which matches the current RGB-D images to the model surface on GPU rather than a keyframe to increase localization accuracy. The method uses an ICP variant for registration, and updates the scene model after new images tracked without RGB information. Rather than the ICP approache, Bylow et al. present a method which estimates the camera pose by directly minimizing the error with a signed distance function (SDF) [6].

Since both of the above methods rely on depth data for localization, they may fail at the circumstances of lacking geometric information. To handle such failure, Lee et al. propose RGBD-Fusion [28] which exploits both depth and visual information. However, it is still limited to small workspaces because of the inevitable drift in large environments, furthermore too much GPU memory and extensive computational power are required to process the dense scene model. Steinbrucker et al. propose a direct method which achieves real-time performance with only CPU needed [38]. The proposed method estimates camera motions by minimizing an energy function based on gray residuals and the depth data is used for mapping the pixels between RGB-D images. This gives better result than ICP based SLAM under circumstances with small displacements. Kerl et al. [26] propose a robust error function based on the t-distribution to reduce the influence of large residuals, which makes it insensitive to noises and outliers. They further propose a method which combines both gray and depth data residuals [27]. In addition, an entropy-based criterion is introduced for keyframes selection and loop closure detection followed by a pose graph optimization. Similar approach is also employed in semi-dense monocular SLAM algorithm like large-scale direct monocular SLAM (LSD-SLAM) [11].

Glocker et al. [47] proposed a VolumeFusion method to solve the limitations by representing the volumetric map with a rolling cyclical buffer and optimize the trajectory by a pose graph while loops are detected with the bag-of-words (BoW) based detector [12] using SURF descriptors. Rather than optimizing the global trajectory, Whelan et al. propose ElasticFusion [48], which performs the global dense surfel-based maps optimization through non-rigid surface deformations after the loops are detected using randomized ferns [13].

Comparison Direct methods achieve higher robustness in feature-less environments by using more information rather than only keypoint correspondences. However, they have some limitations compared with feature-based solutions. First, direct methods seem to lack robustness against moving objects, since pixels from small baselines may lead to many error correspondences and result in poor motion estimations or even lost. Second, some researches show that bundle based optimization is more accurate than a pose-graph optimization where sensor measurements are discarded, since the bundle adjustment optimizes both cameras and map over sensor measurements [33]. Moreover, direct methods also accumulate drift over frames while frame-model approach like KinectFusion has much smaller drift.

Our method inherits the advantages of both feature-based and direct approaches. First, our method handles small baselines with direct approach and wide baselines with keypoint correspondences. The procedure keeps a small drift over hundreds of frames without global optimization, and keyframes contain sensor measurements which enables a global optimization for better results. Second, it is very efficient since most frames are tracked directly from small baselines and features are only detected in keyframes. As features are extracted only for loop closure in approaches like LSD-SLAM [11] and VolumeFusion [47], they are reutilized with high efficiency while they are not only used for refinement and mapping, but also applied in local and global optimization. In addition, the system exploits fresh areas very quickly and works well against fast motion blur and pure rotation, furthermore, it is more robust with moving objects in scene or wide baselines.

3 Semi-direct tracking and mapping

The overview of proposed Semi-direct Tracking and Mapping (SDTAM) is depicted in Fig. 1 which contains four main parts: direct tracking, feature based mapping, global optimization, and map fusion.

1. The direct tracking component estimates relative poses of new RGB-D frames with respect to the last keyframe by minimizing the photometric and depth residuals. This is very efficient since no features are needed and the pose can be directly optimized from a coarse-to-fine scheme.
2. The keyframe poses are refined by the feature based mapping component and the chosen keyframes are inserted to the global map. A local optimization which is a combination of bundle adjust and pose graph optimization is then performed to optimize the local mappoints and keyframe poses.
3. The global optimization is taken to detect and close loops so that the consistent map can be obtained.
4. The map fusion unit fuses the keyframes with an octree representation and generates a textured triangle mesh for real-time visualization.

3.1 Basic notations

In order to make reader easily understand the representation of coordinate and projection, the symbol definitions are listed in Table 1, and the notations are briefly described as follows.

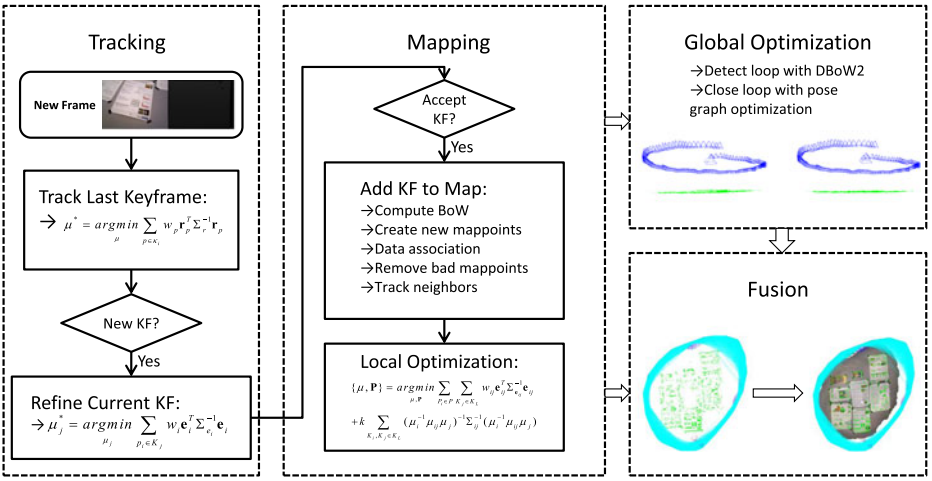


Fig. 1 The framework of the proposed method. RGB-D frames are directly tracked by minimizing both photometric and depth residuals and a minimization of the proposed measurement errors is taken to refine the pose of keyframe (KF) with detected keypoints. When frame is accepted, it is inserted to the global map and a local optimization is performed to optimize the local map. Meanwhile, the loop is detected with a BoW method based on ORB descriptors and closed by a pose graph optimization. Finally, keyframes are fused to generate a textured triangle mesh for demonstration and further use

3.1.1 Coordinate definition and transformation

The camera pose is generally represented with a transformation matrix H:

$$\mathbf{H}_{4\times4} = \begin{bmatrix} \mathbf{R}_{3\times3} & \mathbf{t}_{3\times1} \\ \mathbf{0} & 1 \end{bmatrix}, \tag{1}$$

where the rotation $\mathbf{R}$ is a 3×3 orthogonal matrix $\mathbf{R} \in SO(3)$, in which $SO(3)$ represents the 3D rotation group of Lie group [37]. The translation $\mathbf{t}$ is a 3×1 vector $\mathbf{t} \in \mathbb{R}^3$. The transformation matrix $\mathbf{H}$ is over-parametrized and has twelve parameters with only six degrees of freedom. Therefore, a twist coordinate representation μ is given as a member of the Lie group $SE(3)$ [37]:

$$\mu = (v_1, v_2, v_3, q_1, q_2, q_3)^T \in \mathbb{R}^6, \tag{2}$$

where v_1, v_2, v_3 represent translation and q_1, q_2, q_3 are the angular positions corresponding to rotation matrix $\mathbf{R}$, $SE(3)$ denotes the 3D rigid transformations. The transformation $\mathbf{H}$ can be calculated from μ using the exponential function according to Lie algebra [37]:

$$\mathbf{H} = \exp(\mu). \tag{3}$$

To transform a point $\mathbf{P} = (X, Y, Z, 1)^T$ in the world coordinate to the camera coordinate, we can use the left-multiplication of matrix $\mathbf{H}$:

$$\mathbf{P}' = \mathbf{R}\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \mathbf{t} = \mathbf{H}\mathbf{P} = \exp(\mu)\mathbf{P}. \tag{4}$$

Table 1 The mathematical symbols and their definitions in the paper

Symbols	Definitions
$\mathbf{H}_{4\times4}$	Transformation matrix
$SO(3)$	3D rotation group [37]
$SE(3)$	3D rigid transformation group [37]
$\mathbf{R}$	3D rotation matrix
$\mathbf{t}_{3\times1}$	3D translation vector
μ	Twist coordinate
$\mathbf{P}$, $\mathbf{P}'$	Points in 3D space
$\mathbf{P}_C$	Point in camera coordinate
$\mathbf{p}$, $\mathbf{p}'$	Pixel positions and depths in the image
x, y	Pixel coordinates at x and y axes in the image
d	Depth
$Proj(\cdot)$	Projection operator
f_x, f_y	Focal lengths
$[\cdot]_d$	The depth part of the measurement
$[\cdot]_Z$	The Z component of a 3D point
$I_i(p)$	The value of pixel p in i-th image
r_p	Photo-metric and depth errors
w_p	Weight based on t-distribution
$\sum_r$	Covariance of r_p
$\mathbf{e}$	Measurement error

3.1.2 Camera projection

The measurement of a world point $\mathbf{P}_C = (X_C, Y_C, Z_C, 1)^T$ in the camera-oriented coordinate can be represented as $\mathbf{p} = (x, y, 1/d)^T$, including both pixel coordinates $(x, y)^T$ and inverse depth measurement $1/d$. The reason of using inverse depth measurements instead of depth information will be explained in Section 3.3.2.

A standard pinhole camera model is used to represent the projection:

$$\mathbf{p} = Proj(\mathbf{P}_C) = \left(\frac{X_C f_x}{Z_C} + x_0, \frac{Y_C f_y}{Z_C} + y_0, 1/Z_C \right)^T. \tag{5}$$

Here f_x, f_y are the focal lengths and x_0, y_0 are the coordinates of the camera center in the standard pinhole camera model.

On the contrary, if measurement $\mathbf{p}$ is available, $\mathbf{P}_C$ can be computed with the inverse project function $Proj^{-1}$:

$$\mathbf{P}_C = Proj^{-1}(\mathbf{p}) = \left(\frac{x - x_0}{f_x} d, \frac{y - y_0}{f_y} d, d, 1 \right)^T. \tag{6}$$

3.2 Direct based tracking

After the map generated from the first RGB-D frame, the tracking thread computes relative transformation $\mu_{\mathbf{ji}} \in SE(3)$ of current new frame I_j against the last keyframe K_i. For each

pixel p with valid depth information in K_i, try to find its corresponding measurement p' in I_j with the warp function:

$$p' = warp(\mu_{ji}, p) = Proj(P') = Proj(exp(\mu_{ji}) \cdot Proj^{-1}(p)). \tag{7}$$

Once pixel p' are in sight of the image, the photometric and depth errors can be defined as

$$r_p = \begin{bmatrix} I_i(p) - I_j(p') \\ [p']_d - [P']_Z \end{bmatrix}, \tag{8}$$

where $[\cdot]_d$ returns the depth part of the measurement, $[\cdot]_Z$ means the Z component of a 3D point, and $I_i(p)$ denotes value of pixel p in i-th image.

The optimized relative pose μ^*_{ji} can be computed by minimizing both the photometric and depth residuals:

$$\mu^*_{ji} = \underset{\mu}{argmin} \sum_{p \in K_i} w_p r_p^T \Sigma_r^{-1} r_p, \tag{9}$$

here Σ_r is the covariance of r_p and w_p is the weight based on t-distribution $p_t(0, \Sigma, v)$:

$$w_p = \frac{v+1}{v + r_p^T \Sigma^{-1} r_p}. \tag{10}$$

A coarse-to-fine scheme is employed and the relative pose μ_{ji} is computed iteratively.

Assuming that relative pose parameters are normally distributed, the information matrix of μ_{ji} can be represented by the Hessian matrix $\mathbf{A}$, which means $\mu \sim N(\mu^*, A^{-1})$. To ensure enough overlap between last keyframe and new frames, a new keyframe is created when the distance exceeds the fixed threshold. The relative distance is curved by a weighted combination of translation and rotation described in [11]:

$$dist(\mu_{ji}) := \mu_{ji}^T \mathbf{W} \mu_{ji}, \tag{11}$$

here $\mathbf{W}$ is a diagonal matrix with different weights for each parameter in μ_{ji}, and the translation weights are scaled according to the mean inverse depth.

3.3 Feature based mapping

3.3.1 Pose refinement

The direct tracking system is robust but with inevitable drift after several keyframes. Once a new keyframe is inserted by the tracking thread, we refine the keyframe pose with the feature based method. The procedure used to refine the pose can be summarized as follows:

1. Hundreds of (about 1000) keypoints are detected with adaptive fast algorithm and described with ORB feature descriptors.
2. Matches between last keyframe K_i and current keyframe K_j are found by projecting last mappoints to current image, and then current pose μ_j is updated by the correspondences.
3. A local sub-map is composed of keyframes which share observations with the current frame and the final pose μ_j is determined by tracking the sub-map.

The projection $\mathbf{p}'$ of map point $\mathbf{P}$ can be represented as:

$$\mathbf{p}' = Proj(\mathbf{P}_C) = Proj(\exp(\mu_j)\mathbf{P}). \tag{12}$$

Given a set of matches between 3D map points and RGB-D frame, we can estimate the camera pose μ_j. To take the best use of both pixel coordinates and depth information,

we propose a robust error function to estimate the motion. The error $\mathbf{e} = (e_x, e_y, e_d)^T$ is defined as the difference between measurements $\mathbf{p} = (x, y, 1/d)^T$ and predicted $\mathbf{p}'$ of a map point $\mathbf{P}$:

$$\mathbf{e} = \begin{bmatrix} e_x \\ e_y \\ e_d \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1/d \end{bmatrix} - Proj(\exp(\mu_j)\mathbf{P}). \tag{13}$$

In ideal circumstances, the residuals should be zero. However, due to measurement noises, the residuals are distributed according to the probabilistic sensor model $p(\mathbf{e}|\mu_j)$. We estimate the camera pose μ_j by minimizing the weighted squared residual function:

$$\mu_j^* = \underset{\mu_j}{argmin} \sum_{p_i \in K_j} w_i \mathbf{e}_i^T \Sigma_{e_i}^{-1} \mathbf{e}_i. \tag{14}$$

The weights w_i are given to decrease influence of large residuals by the robust Tukey weighting function, which is based on the robust statistical distribution of the given associated residuals.

$$w_i = \begin{cases} \left(1 - \frac{\mathbf{r}_i^T \Sigma_{e_i}^{-1} \mathbf{r}_i}{\sigma}\right)^2, & \mathbf{r}_i^T \Sigma_{e_i}^{-1} \mathbf{r}_i \le \sigma, \\ 0, & \mathbf{r}_i^T \Sigma_{e_i}^{-1} \mathbf{r}_i > \sigma. \end{cases} \tag{15}$$

3.3.2 Determination of Σ_{e_i}

We solve the problem described in the (14) with the g2o framework [15] and thus the optimized pose μ_j can be obtained. The information matrix Σ_e^{-1} adjusts the weights of geometric error e_x, e_y and depth error e_d. Theoretically, the geometric error and depth error cannot be optimized together directly, here we give a formulation to transform the depth error e_d to geometric error, so that the information matrix can be identified.

As we know, the Kinect 1.0, one type of RGB-D sensors, obtains depth information with an infrared (IR) projector and an IR camera, which based on similar mathematics foundation of the stereo triangulation. As we can see in Fig. 2, point O is the projector origin and P is the real point with ground-truth depth d corresponding to a projection p in the camera image plane. d' is the depth estimation which results from the measurement p' and error e. Assuming the image plane is parallel to OO', we add an auxiliary line PG which is parallel to OO'. The following equations can be obtained from the triangle similarity relationship:

$$\begin{cases} \frac{e}{D'} = \frac{f}{d}, \\ \frac{d'-d}{D'} = \frac{d'}{D}. \end{cases} \tag{16}$$

Here f is the focal length and D is the distance between projector center and camera center. With further derivation we can obtain:

$$e = \frac{fD'}{d} = \frac{fD(d'-d)}{d'd} = fD(1/d - 1/d'). \tag{17}$$

It explains the reason why the inverse depth rather than depth is used for optimization. Assuming the reprojected residuals are independent with an identity covariance, from the equation we can identify the information matrix $\Sigma_e^{-1} = diag(1, 1, (fD)^2)$. Since we detect keypoints on different levels of pyramid of images, the information matrix is also associated with the scale of image in practical circumstances. For different keypoints p_i in the (14), the information matrix $\Sigma_{e_i}^{-1} = fac^{-l_i} \Sigma_e^{-1}$. Here fac is the scale factor of pyramid, and l_i means the level where keypoint p_i is detected.

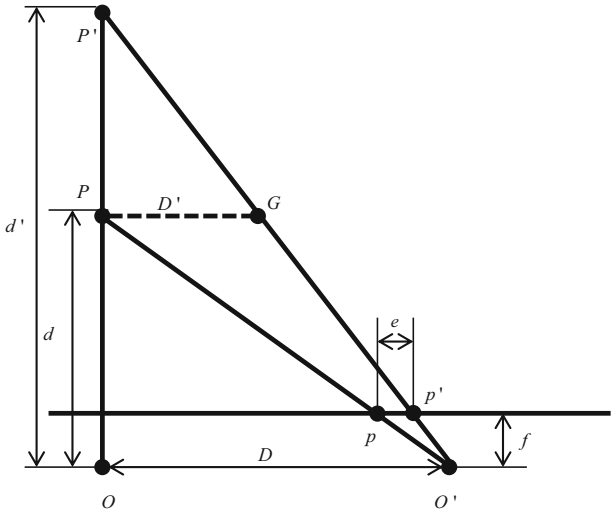


Fig. 2 The transformation from depth error $d' - d$ to pixel based error e

Although the above procedure is generally good enough to estimate the camera motion, further improvement can also be achieved just using the projection residual for points without depth information. Some corresponding points may not contain valid depth measurements (the depth value is assigned to be zero when it is not valid), these points may get a small weight and be ignored because of the huge residual. Since they still catch useful pixel coordinates information, we treat them with a different Σ_e where e_d has a large variance so that RGB data is still used.

3.3.3 Keyframe insertion

After the pose μ_j of keyframe K_j is refined by tracking the local mappoints, we insert it to the map if the estimation is good enough and mapping thread is idle. Several procedures are conducted for a keyframe insertion:

1. **BoW computation**: The BoW vector is computed and registered to a database for triangulation, loop closure detection and relocalization. We use the approach DBoW2 [32] proposed by Rául et al., which is comparable to the well-know Fast Appearance Based Mapping (FAB-MAP) [14] but based on ORB descriptors with better efficiency.
2. **New map points creation**: We try to estimate the world coordinate P of keypoint p with the inverse project function $P = Proj^{-1}(p)$. For keypoints without valid depth information, we find their correspondences by epipolar search in nearby keyframes and obtain their 3D positions from triangulation.
3. **Data association**: After new mappoints created, their correspondences in neighbor keyframes are found by projection and new observations are added. Once the corresponding keypoints refer to existed mappoints which should be the same mappoint, they are combined together for reducing redundancy.
4. **Bad mappoints culling**: Although the proposed matching method based on projection is pretty robust with very few outliers, bad mappoints are inevitable and should be

removed since they are harmful to pose optimization. A mappoint that meets one of the following conditions is regarded to be bad:

(a) It contains less than 2 observations.
(b) It is observed by less than 3 keyframes and with found ratio below 0.2 after local optimization.

Since outlier observations will be removed after local optimization, a point is checked for several times to make sure it is reliable.

5. **Nearby keyframes tracking**: The connections of keyframes are updated according to the number of shared observations. Next, nearest neighbor keyframes are selected and they are tracked directly using the method introduced in Section 3.2. The tracking results are regarded as constraints for successive procedure.

3.3.4 Local optimization

A local optimization is performed to refine the current keyframe K_j and its connected keyframes K_c with all map points $\mathbf{P}$ seen by them. Keyframes K_f that see these points are also included but remain fixed during optimization. The optimization is performed with a combination of local bundle adjustment and pose graph optimization:

$$\begin{aligned}\{\mu, \mathbf{P}\} = \underset{\mu,\mathbf{P}}{argmin} & \sum_{P_i \in P} \sum_{K_j \in K_L} w_{ij} \mathbf{e}_{ij}^T \Sigma_{\mathbf{e}_{ij}}^{-1} \mathbf{e}_{ij} \\ & + k \sum_{K_i, K_j \in K_L} (\mu_i^{-1} \mu_{ij} \mu_j)^{-1} \Sigma_{ij}^{-1} (\mu_i^{-1} \mu_{ij} \mu_j),\end{aligned} \tag{18}$$

where K_L means all the local keyframes $K_j \cup K_c \cup K_f$ and k is the pose graph factor used to adjust weights of direct odometer constraints.

3.4 Global optimization

Loop closure and global optimization are always demanded for incremental frame-to-frame tracking approaches due to the drift accumulation [8, 27]. A small drift in the motion estimation could lead to intolerable error over keyframes.

The DBoW2 [32] package is adopted for loop detection in this work. We collect all keyframes sharing words with current keyframe and compute the similarity between them, while all the neighbor keyframes that already share observations are discarded. Then the candidates are sorted according to the similarity. In order to gain robustness, they are chosen to be loop candidates only if its neighbors are also candidates to gain robustness. For remaining candidates, we find their correspondences with current keyframe and compute a coarse relative transformation using RANSAC. Once a candidate with enough inliers is found, we accept the loop and optimize the relative pose by projection matching.

After the loop is detected, a data association step is firstly performed to fuse duplicated mappoints and add new observations. The approach introduced in Section 3.3.4 could be used to optimize poses of both keyframes and points, but it is computationally expensive because of too many edges. Since bundle adjust approach takes too much time for global optimization, a pose graph optimization described in [40] is adopted to close the loop effectively. An undirected weighted graph is used to describe the co-visibility information between keyframes, while a keyframe is presented as a node and an edge represents the relative transformation between the two linked keyframes. An edge exists if two linked

keyframes share some mappoints, and its weight is defined as the number of the shared mappoints. This approach retains preciseness which benefits from local optimization and requires small computation. Since it only optimizes the keyframe poses, the related map points should be transformed after the graph optimization. Figure 3 illustrates the loop fusion results of sequence *fr3/near* from TUM datasets [41] which shows a consistent map is obtained after the loop is closed.

3.5 Relocalization

Although great efforts are applied to the tracking system to make it as robust as possible, it may still fail in some conditions. Once the tracking is lost, a relocalization procedure should be done to recover tracking system from failure. The BoW vector is computed and keyframe candidates are sorted as introduced in Section 3.4. A coarse motion is estimated with RANSAC algorithm based on correspondences between current frame and candidates. If enough inliers are found, tracking could continue based on the estimated pose.

3.6 Map fusion

In order to manage point clouds effectively, the volumetric 3D mapping algorithm [39] is used for display, path planning and obstacle avoidance. Different from most existed implementations that heavily rely on GPUs, this method generates triangle meshes with textures from a signed distance function and runs on a standard CPU in real-time, which makes it is suitable on platforms like embedded devices or MAVs. For achieving better computation performance the procedure is only performed after keyframe insertion.

4 Experiments and results

We evaluate the proposed approach on two widely used RGB-D benchmarks TUM [41] and ICL [24] since they both provide synchronized ground-true poses which can be adopted to evaluate the tracking accuracy. Because the ground-truth point cloud model is provided in ICL, the reconstruction accuracy is also compared. However, both of TUM and ICL do not

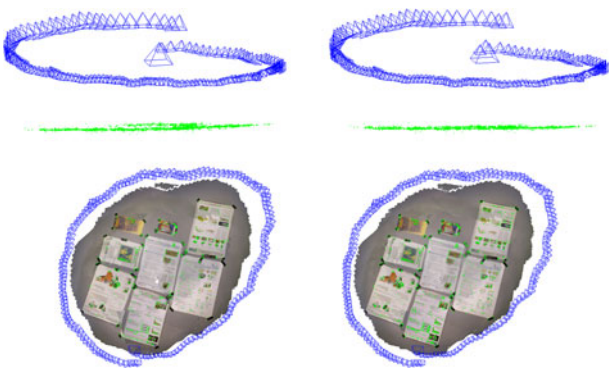
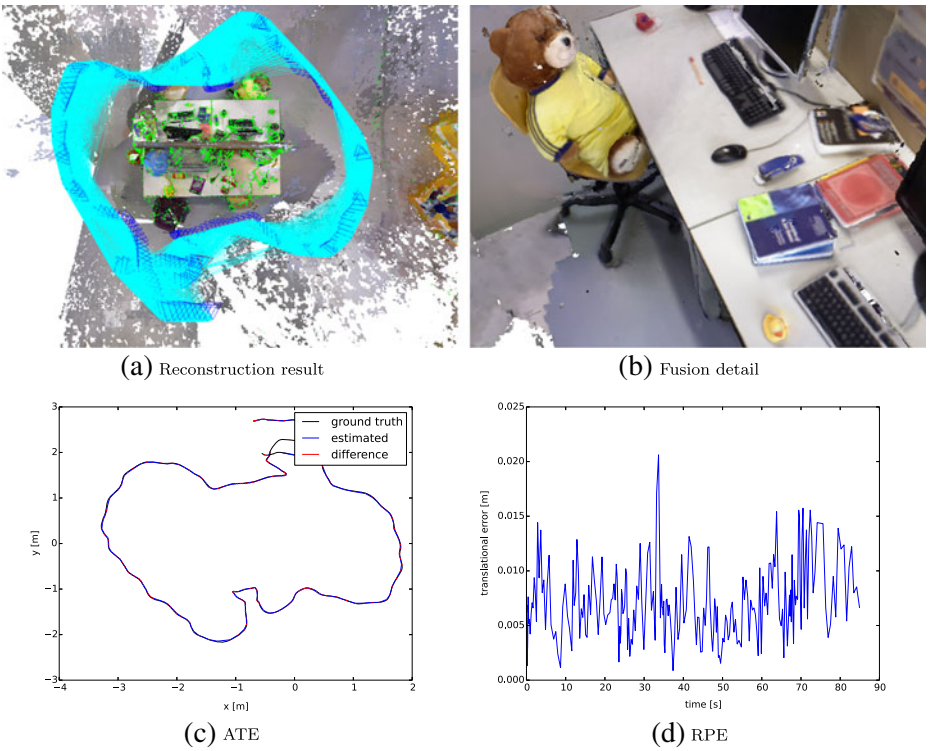


Fig. 3 Comparison results before and after loop fusion of sequence *fr3/near*. The mappoints (*green*) before (*left-top*) are not in one plane and their positions are corrected after loop closed (*right-top*). The texture is not aligned correctly before optimization (*left-bottom*) and seems well after (*right-bottom*)



(a) Reconstruction result (b) Fusion detail

(c) ATE (d) RPE

Fig. 4 The reconstruction results of *fr3/office* sequence of TUM dataset. **a** The reconstruction result seen from *top* of the desk where mappoints are *green* and keyframes are *blue*. **b** The detail of fusion mesh result demonstrates high accuracy of our system. **c** The two-dimensional plot of estimated trajectory compared against ground truth trajectory. **d** The relative pose error (RPE) plot indicates a small drift of our approach

contain large-scale sequences, so that experiments on large sequences captured by ourself are also conducted. The dataset is public available on the website and some results based on the dataset are shown in Section 4.3. To demonstrate the computation performance of our algorithm, the time usage statistics results are illustrated in Section 4.4. We also offer a demonstration video at *youtube*.[1]

4.1 Experiments on TUM

TUM dataset [41] has 39 sequences which are captured in two different indoor environments. It contains ground-truth trajectory from motion capture system and provides tools for trajectory accuracy evaluation. The results of the proposed method on *fr3/office* sequence is depicted in Fig. 4, where the mappoints are shown in green, keyframes in blue, current keyframe in red, and keyframe connections in sky-blue. Lots of details including the bear doll, the pen and books on the desk are shown clearly in Fig. 4b, which indicates high accuracy of the proposed method. The ground-truth and estimated trajectories are depicted in Fig. 4c and they are almost identical. We plot the relative pose error in meters with respect

[1] https://youtu.be/Gy_eA1a86cU

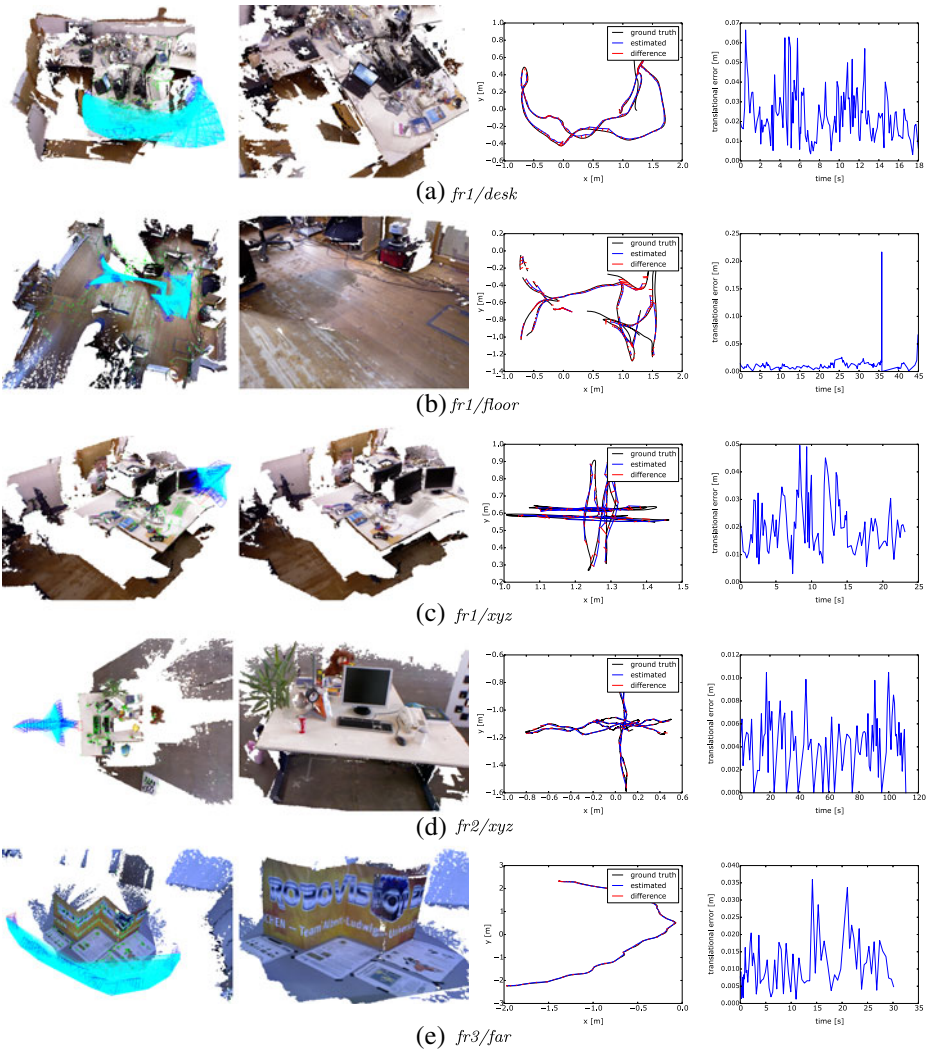


Fig. 5 More results of our algorithm on TUM dataset illustrate the reconstruction outcomes (1st column), fusion details (2nd column), trajectory comparisons (3rd column), and relative pose errors (4th column). The *blue lines* in the first column images represent the tracked trajectories, and cyan lines show the local connections between keyframes. The *white* areas in the images of first and second column images are caused by the RGB-D camera cannot provide depth values at those areas. Therefore, some regions are not reconstructed

to time in Fig. 4d and most translation errors are below 1.5 cm, which indicates the small drift of our algorithm. Some more results of other sequences are illustrated in Fig. 5.

The root-mean-square-error (RMSE) introduced in the TUM RGB-D benchmark [41] is used to evaluate the trajectory accuracy of the direct tracking odometer described in Section 3.2, and results of our system with or without loop fusion are provided in Table 2. In addition, we compare our method to four state-of-the-art approaches: RGB-D SLAM [8], KinectFusion [34], direct visual odometer (DVO) [27], and VolumeFusion [47]. The table lists RMSE of the absolute trajectory error (unit is m) for different methods, where

Table 2 Comparison of absolute trajectory error (ATE) of proposed method and four state-of-the-art approaches

	SDTAM (Ours)				Kinect	RGB-D	Volume
Sequence name	Direct	Direct+KF	Direct+KF+Loop	DVO [27]	Fusion [34]	SLAM [8]	Fusion [47]
fr1/xyz	0.054	**0.011**	**0.011**	**0.011**	0.026	0.014	0.017
fr1/rpy	0.086	0.031	0.031	**0.020**	0.133	0.026	0.028
fr1/desk	0.055	**0.018**	**0.018**	0.021	0.057	0.023	0.037
fr1/desk2	0.117	**0.043**	**0.043**	0.046	0.420	0.043	0.071
fr1/room	0.305	0.205	0.084	**0.053**	0.313	0.084	0.075
fr1/plant	0.039	0.072	0.034	**0.028**	0.598	0.091	0.047
fr2/xyz	0.017	0.015	0.015	0.018	–	**0.008**	0.029
fr2/person	0.180	**0.079**	**0.079**	–	–	–	-
fr3/long	0.104	0.018	**0.010**	0.035	0.064	0.032	0.030
fr3/nst	0.045	0.020	**0.013**	0.018	–	0.017	0.031
fr3/far	0.010	**0.009**	**0.009**	0.017	–	–	-
fr3/sit_xyz	0.028	**0.008**	**0.008**	–	–	–	-
fr3/sit_halfsph	0.116	**0.012**	**0.012**	–	–	–	-
fr3/walk_xyz	1.436	**0.011**	**0.011**	–	–	–	-
fr3/walk_halfsph	0.649	**0.060**	**0.060**	–	–	–	-

The measure metric is root-mean-square-error (RMSE) and the unit is meter (m). From the results, the proposed method outperforms other methods. In the table, 'Direct' represents the results of direct tracking, 'Direct+KF' means the method that refines keyframe poses with the feature-based method and local optimization while global optimization is not included, and 'Direct+KF+Loop' denotes the results based on 'Direct+KF' with global optimization and loop closure detection. The above mentioned three columns show results of different combinations of key techniques in the proposed SDTAM. The last four rows results indicate that our system remains high accuracy and robustness in situations with moving objects

our approach achieves the best accuracy. Some of rows get the same RMSE with or without loop fusion since no loop is detected in these sequences. It should be noted that our approach could handle pure rotation and even works well in sequences with strong motion blur and fast rotation movements such as *fr1/desk*. Especially in sequences of *fr3*, our system achieves remarkable preciseness with very small drift and obtains much smaller RMSE over hundreds of keyframes in *fr3/office* sequence than other approaches even without loop fusion. We consider that this is resulted from that images in sequences of *fr3* are undistorted while other sequences not and an ideal pinhole camera model is used in this paper. And thanks to the error function based on measurements introduced in Section 3.3, our method can obtain the optimal estimation that makes the best use of measurements. The results in the last four rows indicate that our system remains high accuracy and robustness in situations with moving objects while direct odometer drifts a lot in these sequences.

4.2 Experiments on ICL

As referred in paper [24, 47], an algorithm achieving state-of-the-art trajectory accuracy on a camera trajectory benchmark does not always imply a high-quality surface reconstruction due to the frame-to-model tracking component of the system. To evaluate the accuracy of 3D reconstructions, the ICL dataset [24] provides a ground-truth point cloud model for

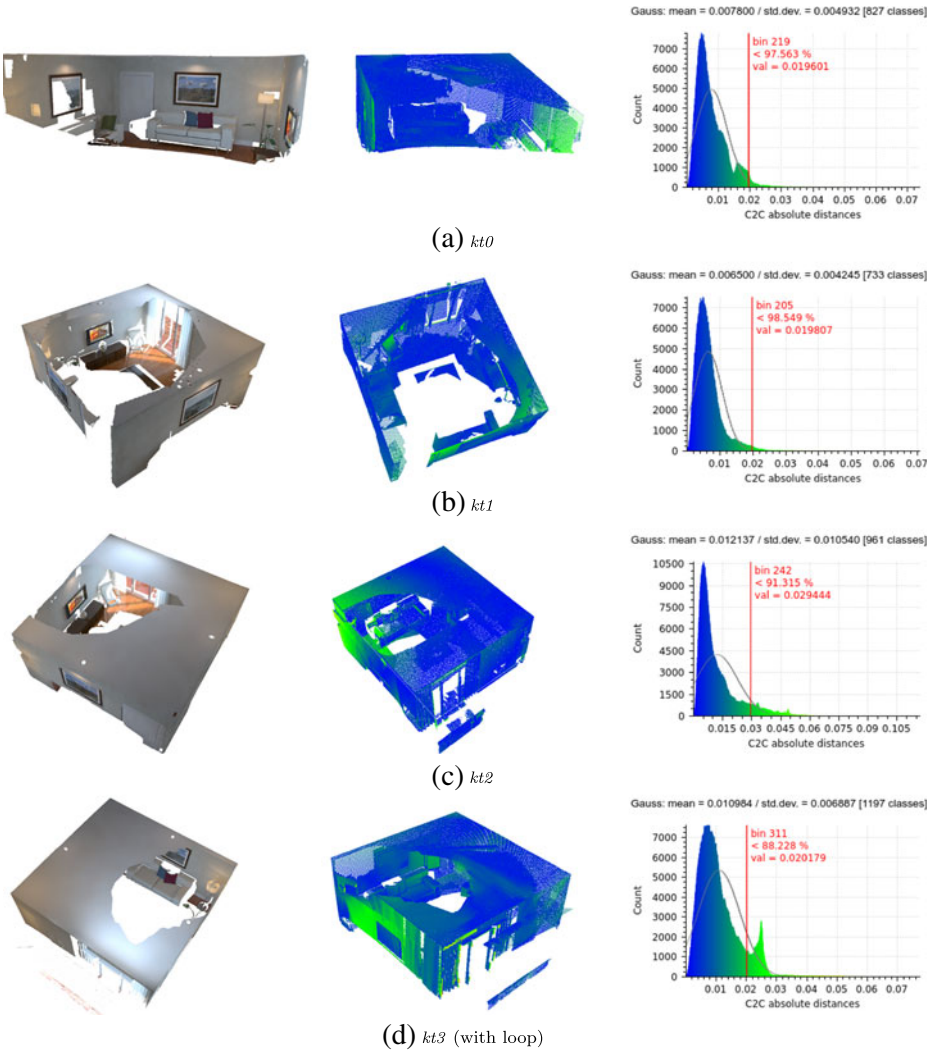


Fig. 6 The reconstructed surface evaluation results for all four trajectories on the living room dataset. The *left* images demonstrate the fusion results of the reconstructions. The point clouds are color coded with respect to the obtained ICP error and the histogram statistics are shown in the *right* column

the living room sequences. We save our mesh model from fusion to an *.obj* file and align the sampled cloud points to the ground true model using ICP with *CloudCompare*,[2] an open source software used to compare the ground truth model with the reconstruction result and compute reconstruction statistics. The living room dataset contains four sequences, and the evaluation results of all reconstructions are plotted in Fig. 6. The heatmaps show the differences between our reconstructed mesh and the ground-truth model, in which blue

[2] http://www.danielgm.net/cc/

Table 3 RMSE (cm) comparison of the absolute trajectory error (ATE) to five other systems

System	*kt0*	*kt1*	*kt2*	*kt3*	Average
DVO [27]	10.4	2.9	19.1	15.2	11.9
RGB-D SLAM [8]	2.6	0.8	1.8	43.3	12.1
MRSMap [42]	20.4	22.8	18.9	109	42.8
Kintinuous [46]	7.2	**0.5**	**1.0**	35.5	10.9
ElasticFusion [48]	0.9	0.9	1.4	10.6	3.4
SDTAM (Ours)	**0.8**	1.1	1.2	**1.8**	**1.2**

indicates small error and red indicates large error. We also compute the histogram statistics with color for all the cloud points and illustrate them in Fig. 6.

We compare our approach to five other state-of-the-art SLAM methods: DVO [27], RGB-D SLAM [8], MRSMap [42], Kintinuous [46], and ElasticFusion [48]. The RMSE statistics of the ATE are listed in Table 3 and comparison results of surface accuracy are summarized in Table 4, which proves that our method obtains higher accuracy on both trajectory and surface evaluations. Especially in sequence *kt3*, our system fuses a loop and keeps high accuracy while most approaches show poor performances.

4.3 Experiments on NPU

Above experiments use datasets of small workspaces, in order to evaluate the ability of handling large-scale data, we test the proposed method on several sequences which contain long trajectories. The dataset[3] is recorded by a Kinect for XBOX 360, which contains several sequences in the campus of Northwestern Polytechnical University.

The dataset contains three sequences:

1. *NPU/ShelvesSmall*: The sequence contains two loops with some repetitive scenes and its trajectory is about 40 meters.
2. *NPU/Shelves*: The scene of this sequence is a large room in the library with hundreds of shelves, which is highly repetitive and the trajectory is about 100 meters long with two loops.
3. *NPU/LibraryFloors*: This sequence consists of two floors in the library hall with an over 100 meters long trajectory.

Each sequence is composed of colorful and depth image pairs which have been aligned with OpenNI. This is a challenging dataset since it contains fast motion, rolling shutter, repetitive scenes, and even poor depth informations. In our experiments, the camera is calibrated and the parameters are offered in the dataset.

Our system runs in real time on the entire dataset and some of reconstruction results are demonstrated in Figs. 7, 8 and 9. Figure 7 shows the result of sequence *NPU/ShelvesSmall* which contains two loops with strong visual aliasing that may lead to wrong loop detection. From the resulting image, we can see that the proposed method can successfully detect the loop and output correct trajectory. The result of sequence *NPU/LibraryFloors* is depicted in Fig. 8. The sequence captures the scene just containing planar objects which might lead to

[3]The NPU dataset is public available at http://adv-ci.com/rgbd/npu/

Table 4 Comparison of surface reconstruction quality for all sequences of living room dataset

System	*kt0*	*kt1*	*kt2*	*kt3*	Average
DVO [27]	3.2	6.1	11.9	5.3	6.6
RGB-D SLAM [8]	4.4	3.2	3.1	16.7	6.8
MRSMap [42]	6.1	14.0	9.8	24.8	13.7
Kintinuous [46]	1.1	0.8	0.9	15.0	4.4
ElasticFusion [48]	**0.7**	0.7	**0.8**	2.8	1.3
SDTAM (Ours)	**0.7**	**0.6**	1.2	**1.1**	**0.9**

The mean distances (cm) of ICP errors for the point clouds are illustrated

the failure of strong depth-relied method, e.g. KinectFusion. But our approach successfully reconstructs the scene, because our method utilizes both the information from the image and depth, and consequently better robustness can be realized. Figure 9 shows the proposed method correctly detects two loops in the scene, which consists of many repetitive objects.

4.4 Computational performances and parameters

To evaluate the computational performances of our algorithm, the time usage statistics for some important functions are measured and analyzed. Experiments are performed on a notebook running 64-bit Linux, where the hardware includes Intel Core i7-4710 CPU with 2.50GHz and 16 GB RAM. The sequence *fr3/office* is used as the evaluating data due to the fact that both loop fusion and relocalization are included. The time usage statistics of this sequence with 1000 and 500 keypoints in each keyframe are listed in Tables 5 and 6, respectively. Although most of the data used in the experiments are captured from Microsoft Kinect RGB-D sensor, the proposed method is not limited to the Kinect. The data captured from other RGB-D sensors such as Intel Real-Sense or Prime-Sense can also be processed using the proposed method to implement real-time SLAM. The difference of using various RGB-D sensors is adopting their own camera calibration coefficients.

As we can see from the statistics, it only takes about 15 ms to track a new frame directly with one separated thread on CPU, which means that the system easily achieves real-time on a normal computer and has great potential to remain real-time on embedded systems like cellphones and robots. It takes 64.4 ms to refine a keyframe with 1000 keypoints in average, including the feature detection, keypoint matching, and pose optimization. The mapping

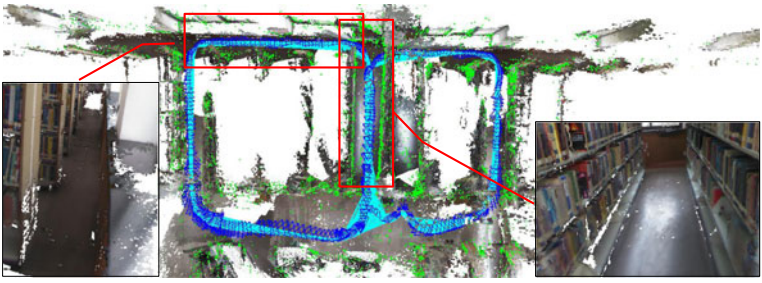


Fig. 7 The reconstruction result of sequence *NPU/ShelvesSmall*. The sequence contains two loops with some repetitive scenes and its trajectory is about 40 meters

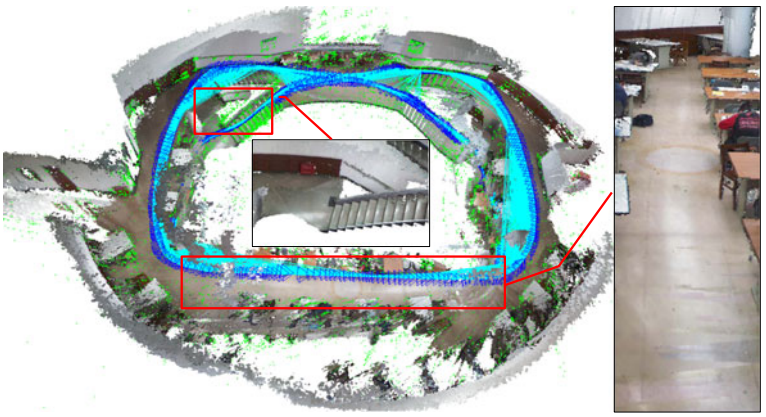


Fig. 8 The reconstruction result of sequence *NPU/LibraryFloors*. The sequence consists of two floors in the library hall with an over 100 meters long trajectory. The reconstructed mesh contains 19,169,329 points and 23,246,188 faces

thread handles keyframe insertion with an average time of 145.9 ms (1000 keypoints) and local optimizations are performed in the spare time with 208.2 ms in average taken. The global optimization thread detects loops with 6.5 ms for each keyframe in average and takes 594.9 ms to correct the loop detected.

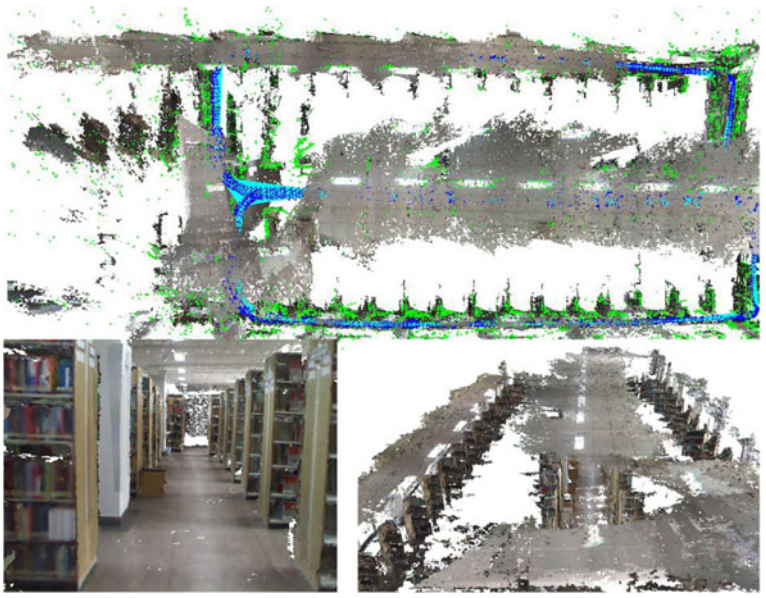


Fig. 9 The reconstruction result of sequence *NPU/Shelves*. From the results, we can conclude that the proposed method detects two loops successfully under the highly repetitive scene over 100 meters

Table 5 Time statistics of sequence *fr3/office* with 1000 keypoints detected

Function	Count	Min	Mean	Max	Total
DirectTracking	2484	5.4ms	14.5ms	35.8ms	35.9 s
PoseRefinement	319	36.3ms	64.4ms	192.3ms	20.6 s
KeyframeInsertion	308	0.0ps	145.9ms	328.6ms	44.9 s
LocalOptimization	232	3.4ms	208.2ms	2.4 s	48.3 s
LoopDetection	306	131.0us	6.5ms	24.1ms	2.0 s
LoopClosure	1	594.9ms	594.9ms	594.9ms	594.9ms
FuseOneFrame	306	34.2ms	75.0ms	894.6ms	23.0 s
Relocalization	1	39.0ms	39.0ms	39.0ms	39.0ms

In summary, the computational complexity of the direct tracking is $O(n_w \times n_h)$ where n_w and n_h represent the image width and height. The computational complexity of keyframe processing is $O(n_k)$ where n_k is the number of keypoints. Therefore the proposed method can achieve real-time speed due to the linear computation complexity.

The computational performance of tracking compared with other methods is listed in Table 7. From the comparison, we can conclude that most of the methods can achieve real-time tracking with the frame-per-second of 30 or above. However, it should be noted that the comparison is not precise because different methods use different measurement methods. In addition, the time statistics are measured using different computers.

While most of parameters can be identified automatically, users need to adjust the keypoints number for each frame and $\mathbf{W}$ in the (11). Theoretically, higher accuracy can be obtained with more keypoints but faster with less keypoints. Therefore, a good balance between accuracy and speed can be achieved through selecting an optimal keypoints number. In the previous experiments, 500 and 1000 keypoints are used to evaluate the performance, while other experiments just use keypoint number of 1000. The $\mathbf{W}$ is a 6×6 diagonal matrix which controls the weights for calculating moving or rotating difference between keyframes. In the implementation, we evaluate different values and find that they are not affecting the overall accuracy. Different values in $\mathbf{W}$ just cause different keyframe

Table 6 Time statistics of sequence *fr3/office* with 500 keypoints detected

Function	Count	Min	Mean	Max	Total
DirectTracking	2484	4.7ms	14.9ms	33.7ms	37.1 s
PoseRefinement	320	19.6ms	34.2ms	106.6ms	10.9 s
KeyframeInsertion	322	0.0ps	68.5ms	183.2ms	22.1 s
LocalOptimization	312	1.5ms	182.3ms	1.4 s	56.9 s
LoopDetection	320	65.0us	2.7ms	12.3ms	855.5ms
LoopClosure	320	306.7ms	306.7ms	306.7ms	306.7ms
FuseOneFrame	320	31.4ms	75.5ms	600.3ms	24.2 s
Relocalization	1	12.8ms	12.8ms	12.8ms	12.8ms

Table 7 Computational performance of tracking for some recent methods

Method	Tracking time (ms)
RGB-D SLAM [8]	350
DVO [27]	32
KinectFusion [34]	15 (with 320^3 volxel resolution)
VolumeFusion [47]	30
SDTAM (Ours)	15

The time unit is milli-second. It should be noted that the comparison is not precise because different methods use different measurement methods. In addition, the time statistics are measure using different computers

numbers, which may slightly affect the computational performance. In our experiments, for all sequences we set the **W** to be an identity matrix that performs well.

5 Conclusions

In this paper we present a novel SLAM approach for RGB-D cameras with high efficiency and accuracy as a result of the seamless combination of direct and feature based methods. The system remains robust in challenging conditions with motion blur, fast pure rotation, and large moving objects. The evaluation results show small drift and high accuracy on both trajectory and reconstruction thanks to the novel error function, which is based on measurements that both geometric and depth informations are utilized in the framework. Furthermore, the proposed method can achieve real-time speed without the use of GPU, and it can be easily applied to embedded systems.

Although the proposed method achieves good performances on accuracy and robustness, there are some improvements can be made in the future. First, the fusion just adopts the keyframe, therefore, the resulting 3D model is not smooth when the system is doing exploring tasks and limited keyframes is available. Second, the fusion procedure adds meshes incrementally, thereby it may fail in dynamic scenes. To make the RGB-D SLAM system more robust, in the following researches semantic information will be extracted from captured data and novel SLAM mechanics will be explored to further improve the accuracy and robustness.

Acknowledgments This work is partly supported by grants from National Natural Science Foundation of China (61202185, 61473231, 61573284), the Fundamental Research Funds for the Central Universities (310201401-(JCQ01009,JCQ01012)), Open Projects Program of National Laboratory of Pattern Recognition (NLPR).

References

1. Bay H, Tuytelaars T, Van Gool L (2006) Surf: Speeded up robust features. In: Computer vision–ECCV 2006. Springer, pp 404–417

2. Bu S, Cheng S, Liu Z, Han J (2014) Multimodal feature fusion for 3d shape recognition and retrieval. IEEE Multimedia 21(4):38–46
3. Bu S, Han P, Liu Z, Li K, Han J (2014) Shift-invariant ring feature for 3d shape. Vis Comput 30(6–8):867–876
4. Bu S, Liu Z, Han J, Wu J, Ji R (2014) Learning high-level feature by deep belief networks for 3-d model retrieval and recognition. IEEE Trans Multimedia 16(8):2154–2167
5. Bu S, Han P, Liu Z, Han J, Lin H (2015) Local deep feature learning framework for 3d shape. Comput Graph 46:117–129
6. Bylow E, Sturm J, Kerl C, Kahl F, Cremers D (2013) Direct camera pose tracking and mapping with signed distance functions. In: RGB-D workshop on advanced reasoning with depth cameras (RGB-D 2013)
7. Chen C, Liu K, Kehtarnavaz N (2013) Real-time human action recognition based on depth motion maps. J Real-Time Image Proc 1–9
8. Endres F, Hess J, Engelhard N, Sturm J, Cremers D, Burgard W (2012) An evaluation of the rgb-d slam system. In: IEEE international conference on robotics and automation (ICRA), 2012. IEEE, pp 1691–1696
9. Engel J, Sturm J, Cremers D (2012) Camera-based navigation of a low-cost quadrocopter. In: IEEE/RSJ international conference on intelligent robots and systems (IROS), 2012. IEEE, pp 2815–2821
10. Engel J, Sturm J, Cremers D (2012) Accurate figure flying with a quadrocopter using onboard visual and inertial sensing. IMU 320:240
11. Engel J, Schöps T, Cremers D (2014) Lsd-slam: large-scale direct monocular slam. In: Computer Vision–ECCV 2014. Springer, pp 834–849
12. Gálvez-López D, Tardos JD (2011) Real-time loop detection with bags of binary words. In: IEEE/RSJ international conference on intelligent robots and systems (IROS), 2011. IEEE, pp 51–58
13. Glocker B, Shotton J, Criminisi A, Izadi S (2015) Real-time rgb-d camera relocalization via randomized ferns for keyframe encoding. IEEE Trans Vis Comput Graph 21(5):571–583
14. Glover A, Maddern W, Warren M, Reid S, Milford M, Wyeth G (2012) Openfabmap: an open source toolbox for appearance-based loop closure detection. In: IEEE international conference on robotics and automation (ICRA), 2012, pp 4730–4735
15. Grisetti G, Strasdat H, Konolige K, Burgard W (2011) g2o: a general framework for graph optimization
16. Grzonka S, Grisetti G, Burgard W (2009) Towards a navigation system for autonomous indoor flying. In: IEEE international conference on robotics and automation, 2009. ICRA'09. IEEE, pp 2878–2883
17. Han J, Pauwels EJ, De Zeeuw PM, De With PH (2012) Employing a rgb-d sensor for real-time tracking of humans across multiple re-entries in a smart environment. IEEE Trans Consum Electron 58(2):255–263
18. Han J, Shao L, Xu D, Shotton J (2013) Enhanced computer vision with microsoft kinect sensor: a review. IEEE Trans Cybern 43(5):1318–1334
19. Han J, He S, Qian X, Wang D, Guo L, Liu T (2013) An object-oriented visual saliency detection framework based on sparse coding representations. IEEE Trans Circuits Syst Video Technol 23(12):2009–2021
20. Han J, Zhang D, Hu X, Guo L, Ren J, Wu F (2014) Background prior based salient object detection via deep reconstruction residual. IEEE Trans Circuits Syst Video Technol 25(8):1309–1321
21. Han J, Zhou P, Zhang D, Cheng G, Guo L, Liu Z, Bu S, Wu J (2014) Efficient, simultaneous detection of multi-class geospatial targets based on visual saliency modeling and discriminative learning of sparse coding. ISPRS J Photogramm Remote Sens 89:37–48
22. Han J, Zhang D, Cheng G, Guo L, Ren J (2015) Object detection in optical remote sensing images based on weakly supervised learning and high-level feature learning. IEEE Trans Geosci Remote Sens 53(6):3325–3337
23. Han J, Chen C, Shao L, Hu X, Han J (2015) Learning computational models of video memorability from fmri brain imaging. IEEE Trans Cybern 45(8):1692–1703
24. Handa A, Whelan T, McDonald J, Davison AJ (2014) A benchmark for rgb-d visual odometry, 3d reconstruction and slam. In: IEEE international conference on robotics and automation (ICRA), 2014. IEEE, pp 1524–1531
25. Henry P, Krainin M, Herbst E, Ren X, Fox D (2012) Rgb-d mapping: using kinect-style depth cameras for dense 3d modeling of indoor environments. Int J Robot Res 31(5):647–663

26. Kerl C, Sturm J, Cremers D (2013) Robust odometry estimation for rgb-d cameras. In: IEEE international conference on robotics and automation (ICRA), 2013. IEEE, pp 3748–3754
27. Kerl C, Sturm J, Cremers D (2013) Dense visual SLAM for RGB-D cameras. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, pp 2100–2106
28. Lee S-O, Lim H, Kim H-G, Ahn SC (2014) Rgb-d fusion: real-time robust tracking and dense mapping with rgb-d data fusion. In: IEEE/RSJ international conference on intelligent robots and systems (IROS 2014), 2014. IEEE, pp 2749–2754
29. Li W, Zhang Z, Liu Z (2010) Action recognition based on a bag of 3d points. In: IEEE computer society conference on computer vision and pattern recognition workshops (CVPRW), 2010. IEEE, pp 9–14
30. Liu L, Shao L (2013) Learning discriminative representations from rgb-d video data. In: Proceedings of the 23rd international joint conference on artificial intelligence. AAAI Press, pp 1493–1500
31. Lowe DG (2004) Distinctive image features from scale-invariant keypoints, Int J Comput Vis 60(2):91–110
32. Mur-Artal R, Tardós JD (2014) Fast relocalisation and loop closing in keyframe-based slam. In: IEEE international conference on robotics and automation (ICRA), 2014. IEEE, pp 846–853
33. Mur-Artal R, Montiel J, Tardos JD (2015) Orb-slam: a versatile and accurate monocular slam system. arXiv:1502.00956
34. Newcombe RA, Izadi S, Hilliges O, Molyneaux D, Kim D, Davison AJ, Kohi P, Shotton J, Hodges S, Fitzgibbon A (2011) Kinectfusion: Real-time dense surface mapping and tracking. In: 10th IEEE international symposium on mixed and augmented reality (ISMAR), 2011. IEEE, pp 127–136
35. Rublee E, Rabaud V, Konolige K, Bradski G (2011) Orb: an efficient alternative to sift or surf. In: IEEE international conference on computer vision (ICCV), 2011. IEEE, pp 2564–2571
36. Segal A, Haehnel D, Thrun S (2009) Generalized-icp. In: Robotics: Science and Systems, vol 2
37. Selig J (2004) Lie groups and lie algebras in robotics. In: Computational noncommutative algebra and applications. Springer, pp 101–125
38. Steinbrucker F, Sturm J, Cremers D (2011) Real-time visual odometry from dense rgb-d images. In: IEEE international conference on computer vision workshops (ICCV Workshops), 2011. IEEE, pp 719–722
39. Steinbrucker F, Sturm J, Cremers D (2014) Volumetric 3d mapping in real-time on a cpu. In: IEEE international conference on robotics and automation (ICRA), 2014. IEEE, pp 2021–2028
40. Strasdat H, Davison AJ, Montiel J, Konolig K (2011) Double window optimisation for constant time visual slam. In: IEEE international conference on computer vision (ICCV), 2011. IEEE, pp 2352–2359
41. Sturm J, Engelhard N, Endres F, Burgard W, Cremers D (2012) A benchmark for the evaluation of rgb-d slam systems. In: IEEE/RSJ international conference on intelligent robots and systems (IROS), 2012. IEEE, pp 573–580
42. Stückler J, Behnke S (2014) Multi-resolution surfel maps for efficient dense 3d modeling and tracking. J Vis Commun Image Represent 25(1):137–147
43. Tao D, Jin L, Wang Y, Yuan Y, Li X (2013) Person re-identification by regularized smoothing kiss metric learning. IEEE Trans Circuits Syst Video Technol 23(10):1675–1685
44. Tao D, Jin L, Liu W, Li X (2013) Hessian regularized support vector machines for mobile image annotation on the cloud. IEEE Trans Multimedia 15(4):833–844
45. Triggs B, McLauchlan PF, Hartley RI, Fitzgibbon AW (2000) Bundle adjustment–a modern synthesis. In: Vision algorithms: theory and practice. Springer, pp 298–372
46. Whelan T, Kaess M, Fallon M, Johannsson H, Leonard J, McDonald J (2012) Kintinuous: spatially extended kinectfusion
47. Whelan T, Kaess M, Johannsson H, Fallon M, Leonard JJ, McDonald J (2015) Real-time large-scale dense rgb-d slam with volumetric fusion. Int J Robot Res 34(4–5):598–626
48. Whelan T, Leutenegger S, Salas-Moreno RF, Glocker B, Davison AJ (2015) Elasticfusion: dense slam without a pose graph. In: Robotics: science and systems
49. Wu C (2011) Siftgpu: A gpu implementation of scale invariant feature transform (sift)(2007), http://cs.unc.edu/~ccwu/siftgpu
50. Yu J, Tao D, Li J, Cheng J (2014) Semantic preserving distance metric learning and applications. Inf Sci 281:674–686
51. Yu M, Liu L, Shao L (2015) Structure-preserving binary representations for rgb-d action recognition. IEEE Trans Pattern Anal Mach Intell

Shuhui Bu received the MSc and PhD degrees in College of Systems and Information Engineering from University of Tsukuba, Japan in 2006 and 2009. He was an assistant professor (2009–2011) at Kyoto University, Japan. Currently, he is an associate professor at Northwestern Polytechnical University, China. His research interests are concentrated on computer vision and robotics, including SLAM, 3D shape analysis, image processing, pattern recognition, 3D reconstruction, and related fields. He has published approximately 40 papers in major international journals and conferences, including the IEEE TMM, IEEE TBME, TVC, C&G, ACM MM, ICPR, CGI, SMI, etc.

Yong Zhao received the BS degree from Northwestern Polytechnical University of China in 2014. Currently, he is working toward the MS degree at Northwestern Polytechnical University, China. His research interests include simultaneous localization and mapping (SLAM), micro air vehicle (MAV), image processing, and computer vision.

Gang Wan received the PhD degree in Cartography and Geographical information engineering from Information Engineering University, China in 2002. He is a professor at Information Engineering University, China. His research includes GIS, virtual reality, image processing. He has published over 20 journal and conference papers in the related areas.

Ke Li received the PhD degree in Cartography and Geographical information engineering from Information Engineering University, China in 2008. He is an associate professor at Information Engineering University, China. His research includes GIS, image processing, and computer vision. He has published over 15 journal and conference papers in the related areas.

Gong Cheng received the B.S. degree from Xidian University, Xi'an, China, in 2007 and the M.S. and Ph.D. degrees from Northwestern Polytechnical University, Xi'an, China, in 2010 and 2013, respectively. He is currently a Postdoctoral Fellow with the School of Automation, Northwestern Polytechnical University, Xi'an, China. His main research interests include computer vision and remote sensing image analysis.

Zhenbao Liu received the PhD degree in computer science from the College of Systems and Information Engineering, University of Tsukuba, Japan, in 2009. He is an associate professor at Northwestern Polytechnical University, Xi'an. He was a visiting scholar in the GrUVi Lab of Simon Fraser University in 2012. His research interests include 3D shape analysis, matching, retrieval and segmentation.