

# Non-iterative RGB-D-inertial odometry

Chen Wang<sup>1</sup>, Minh-Chung Hoang<sup>1</sup>, Lihua Xie<sup>1</sup> and Junsong Yuan<sup>2</sup>

## Abstract

*This paper presents a non-iterative solution to RGB-D-inertial odometry system. Traditional odometry methods resort to iterative algorithms which are usually computationally expensive or require well-designed initialization. To overcome this problem, this paper proposes to combine a non-iterative front-end (odometry) with an iterative back-end (loop closure) for the RGB-D-inertial SLAM system. The main contribution lies in the novel non-iterative front-end, which leverages on inertial fusion and kernel cross-correlators (KCC) to match point clouds in frequency domain. Dominated by the fast Fourier transform (FFT), our method is only of complexity  $\mathcal{O}(n \log n)$ , where  $n$  is the number of points. Map fusion is conducted by element-wise operations, so that both time and space complexity are further reduced. Extensive experiments show that, due to the lightweight of the proposed front-end, the framework is able to run at a much faster speed yet still with comparable accuracy with the state-of-the-arts.*

## Keywords

Non-Iterative, Odometry, Dense Mapping, RGB-D, Inertial Fusion

## 1 Introduction

Simultaneous localization and mapping (SLAM) is one of the most basic capabilities of autonomous robots, and has received increasing attention in tandem with the development of hardware, such as smarter sensors and faster processors (Bailey and Durrant-Whyte 2006). However, visual dense mapping algorithms are still difficult to be applied directly to micro-devices, e.g. head-mounted augmented reality (AR) devices or micro unmanned aerial vehicles (MUAV). This is because micro-devices are only able to provide limited computational resources, such as ultra-low power embedded processors, due to payload and power limitations. Moreover, dense maps, which are crucial for higher level applications, such as collision-free motion planning, object detection and scene understanding, require more computational resources to produce (Henry et al. 2012). This paper aims to develop a lightweight dense mapping system that can be carried by micro-devices, and can achieve faster computation with sufficient accuracy. Leveraging on inertial fusion and kernel cross-correlators (KCC), this paper proposes a non-iterative front-end (odometry) for RGB-D-inertial SLAM systems.

Odometry system, which is in charge of processing sensor information to generate observations to be fed to an estimation machinery (Valencia and Andrade-Cetto 2018), is the most important and time consuming part of a SLAM system. From the following analysis, we find that iterative solutions dominate the existing visual or visual-inertial odometry systems. First, iterative closest point

(ICP) is one of the most widely used algorithms for point cloud matching (Besl and McKay 1992). The problem of ICP and its variants lies in the high complexity, so that many modern algorithms such as (Newcombe et al. 2011a; Whelan et al. 2012, 2015b, 2016; Newcombe et al. 2015) need powerful GPU to process the large amount of data. Second, feature-based (Davison et al. 2007; Klein and Murray 2007; Forster et al. 2014; Henry et al. 2012; Mur-Artal et al. 2015) and direct methods (Kerl et al. 2013a,b; Gutierrez-Gomez et al. 2016; Engel et al. 2014; Newcombe et al. 2011b; Forster et al. 2014) have been proposed for monocular, stereo cameras, and RGB-D cameras (Endres et al. 2014; Kerl et al. 2013a). However, they also need iterative methods, e.g. Gauss-Newton, to minimize the reprojection or photometric error, so that the camera motion can be estimated. Third, the tightly-coupled inertial fusion SLAM algorithms using monocular camera (Concha et al. 2016), stereo camera (Leutenegger et al. 2015), and RGB-D camera (Laidlow et al. 2017) also need iterative numerical solutions to solve an associated non-linear optimization problem. Fourth, to remove outliers, iterative methods like

<sup>1</sup>School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore

<sup>2</sup>Department of Computer Science and Engineering, State University of New York at Buffalo, NY, USA

## Corresponding author:

Chen Wang, School of Electrical and Electronic Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798.

Email: wang.chen@zoho.com

---

RANSAC (Raguram et al. 2008) are also widely used (Mur-Artal et al. 2015; Endres et al. 2014). Last, the recent trend towards deep learning based-methods (Costante et al. 2016; Mohanty et al. 2016; Tateno et al. 2017; Clark et al. 2017; DeTone et al. 2017) stimulates a new wave of research, but the real-time performance may still be low, especially the iterative training process (back-propagation) can only be performed off-line. This challenge opens space for on-line learning techniques.

The above discussion shows that iterative solutions, including but not limited to the abovementioned, may be the main computational burden of an odometry system. To reduce the computational complexity, we have the following observations:

(i) A closed-form solution to odometry system may be able to reduce the computational complexity dramatically. Moreover, a closed-form solution doesn't require well-designed initialization to prevent convergence to a local minimum. However, the objective functions of traditional methods, e.g. tightly-coupled inertial fusion, feature-based and direct methods, are highly non-linear and difficult to find feasible closed-form solutions.

(ii) In many cases, inertial fusion typically introduces additional complexity due to additional filtering or optimization processes for inertial measurements. Is it possible to reduce the complexity based on inertial fusion? One of our intuitions is to determine the camera translation by visual techniques, and determine the attitude by rotation estimation from an inertial sensor, e.g. the attitude and heading reference system (AHRS). However, the magnetic drift is inevitable, the rotation estimation sometimes needs corrections from visual measurements.

To reduce the complexity while keeping sufficient accuracy, we propose to combine a non-iterative front-end (odometry) with an iterative back-end (local and global loop closure) for the RGB-D-inertial SLAM system. To this end, we reformulate the problem of point cloud matching by first leveraging on the inertial fusion to decouple the point clouds. Then a kernel cross-correlator (KCC) is applied to obtain a pose estimation with a closed-form solution. For the back-end, we leverage on the feature-based method that jointly optimizes key-frame poses with landmark features over a pose graph. This results in a lightweight key-frame-only based loop closure system that aims to correct both odometry and inertial drifts. It will be shown that the proposed framework provides a good compromise between efficiency and accuracy. Our main contribution lies in a novel framework for RGB-D-inertial SLAM, especially the non-iterative front-end odometry system.

A preliminary version of this work was presented in (Wang et al. 2017a). It demonstrated that, for the first time, a non-iterative solution is feasible to the RGB-D-inertial odometry system. This paper extends the

initial version comprehensively. First, a flexible framework is proposed for RGB-D-inertial SLAM. Second, the orthogonal reprojection process is extended to dynamic resolution, resulting in a more accurate estimation and detailed map. Third, the non-iterative solution is re-derived in a much simpler way using our recently proposed KCC (Wang et al. 2018b). Last, it is demonstrated that the inertial fusion and non-iterative solution are able to reduce the complexity of RGB-D SLAM dramatically, comparing with several state-of-the-art algorithms. This results in a lightweight RGB-D-inertial SLAM system that can be carried by micro-robot systems.

## 2 Related work

This paper focuses on odometry system, dense mapping, and inertial fusion. The works on monocular/stereo camera will also be reviewed.

### 2.1 Visual odometry and map fusion

Many dense mapping systems resort to ICP and its variants to align point clouds with respect to photometric or geometric constraints. KinectFusion (Newcombe et al. 2011a) is one of the most famous methods, where the pose transformation is obtained by tracking the live depth frame relative to the global model using a coarse-to-fine ICP algorithm with geometric constraints. It permits dense volumetric reconstruction of complex room-sized scenes. Following this work, Kintinuous (Whelan et al. 2012) enables dense mesh-based mapping for extended scale environments and implements a triangular mesh generation module for the map representation. ElasticFusion (Whelan et al. 2015b) combines dense geometric constraints with photometric constraints to achieve robust pose estimation in more challenging scenes. Furthermore, (Whelan et al. 2015a) brings Kintinuous and ElasticFusion together, and adds a method for improving camera-frustum overlap for a greater reconstruction range. The pose estimation is obtained by a dense every-frame volumetric fusion front-end, and the dense surface is corrected by a non-rigid map deformation back-end. DynamicFusion (Newcombe et al. 2015) generalizes the truncated signed distance function to nonrigid case, so that dynamic scenes can be reconstructed and a volumetric 6-D motion field can be estimated. BundleFusion (Dai et al. 2017) applies a local-to-global pose alignment framework and a paralleled sparse-to-dense optimization for sparse feature correspondence and dense geometric and photometric matching. Since the above methods heavily rely on ICP and require powerful GPU to process large amount of data, they are not applicable to systems with low computational power. Instead of matching points directly, DIFODO (Jaimez and Gonzalez-Jimenez 2015) applies a depth flow constraint on the motion model

to derive the linear and angular velocities in a rigid environment.

To decrease the complexity, dense visual odometry (DVO) (Kerl et al. 2013b) proposes a faster, robust method based on a t-distribution-based photometric error model that can be obtained by an iteratively re-weighted least square (IRLS) algorithm. An improved version of DVO (Kerl et al. 2013a) extends the t-distribution model to depth error and proposes an entropy-based similarity measurement for key-frame selection and loop closure detection. The proposed t-distribution model can be reduced to the standard least square minimization problem when error residual is assumed to be normally distributed.

Feature-based method is one of the most widely-used methods in visual odometry. It leverages on an iterative optimization process to minimize reprojection error. The basic idea is that the transformation between camera poses can be recovered by matched features. During the feature-detection step, salient key features that are likely to be matched well with other images are selected. Basically, there are two main approaches to find the corresponding features (Fraundorfer and Scaramuzza 2012). One is to extract features by local search techniques (Forster et al. 2014) and match them between the latest image and key-frame (Mur-Artal et al. 2015). It is suitable for images taken from nearby viewpoints. The other one is to extract features independently and match them based on similarity descriptors (Khan and Wollherr 2015), which can also be used for large motion between viewpoints.

Based on feature tracking, PTAM (Klein and Murray 2007) first proposes to track the camera poses parallel to the mapping thread in a bundle adjustment framework. This enables real-time landmarks tracking and pose estimation at frame-rate. ORB-SLAM (Mur-Artal et al. 2015) carries forward this work by designing a robust system that uses the same features for all SLAM tasks: tracking, mapping, relocalization, and loop closing. This leads to a more reliable and complete solution. One of the earliest dense feature-based SLAM systems is RGB-D mapping (Henry et al. 2012) where Generalized-ICP and RANSAC are combined to optimize the pose graph. RGB-D-SLAM (Endres et al. 2014) exploits feature correspondences by sparse bundle adjustment and has become one of the most successful feature-based methods in RGB-D SLAM family.

In contrast with feature-based methods, direct methods match raw image pixels directly. Since there is no descriptor, only local search techniques can be used to find the corresponding pixels, hence it needs well-designed initialization for large motion between two viewpoints. DTAM (Newcombe et al. 2011b) develops a real-time camera tracking and reconstruction system where pose estimation is from the minimization of photometric error on every pixel. Within a depth probabilistic framework,

SVO (Forster et al. 2014) achieves semi-dense maps with accurate motion estimation by minimizing photometric error outside non-negligible gradients regions. LSD-SLAM (Engel et al. 2014) proposes to build consistent large-scale maps by aligning image directly based on photoconsistency and estimating a filtering-based semi-dense depth map. By incorporating disparity sources from fixed-baseline, Stereo-LSD (Engel et al. 2015) avoids the scale-drift problem and builds a consistent map for large scale environments using stereo cameras. DSO (Engel et al. 2017) proposes a direct sparse odometry system that minimizes the photometric error on sampled pixels with large intensity gradients.

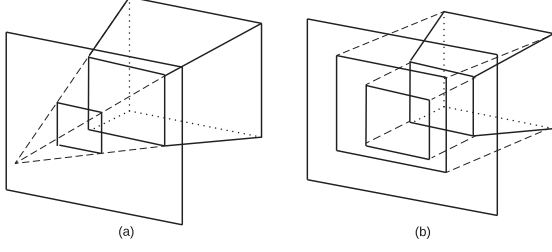
The deep learning based visual odometry methods have also been studied. In (Costante et al. 2016), the authors explore the convolutional neural networks (CNN) to learn visual features for ego-motion estimation. DeepVO (Mohanty et al. 2016) proposes a CNN-based architecture for estimating the object's pose under a known environment. CNN-SLAM (Tateno et al. 2017) combines the direct methods with CNN-predicted depth maps to overcome the absolute scale problem of monocular SLAM. (DeTone et al. 2017) proposes to train two CNNs, one of which operates on single image and extracts salient 2-D points, while another one operates on pairs of the points and estimates the homography.

## 2.2 Inertial fusion

Inertial sensors provide additional constraints for pose estimation and help improve the performance of visual odometry systems. The most computationally efficient inertial fusion approach may be the loosely coupled methods, that process inertial and visual measurements separately. (Weiss and Siegwart 2011) presents an extended Kalman filter (EKF)-based constant complexity framework that treats the visual odometry system as a black box. This leads to a relatively low computational cost for additional processing. MSF-EKF (Lynen et al. 2013) presents a generic and modular multi-sensor fusion framework to deal with delayed absolute or relative measurements. Alternatively, attitude estimation can be extracted and fused into visual estimation algorithms. (Konolige et al. 2010) shows that angular precision of long term visual odometry can be improved a lot by loosely coupled fusion.

In contrast to loosely coupled methods, tightly-coupled methods estimate the states jointly. MSCKF (Mourikis and Roumeliotis 2015) derives a measurement model that is able to express geometric constraints from static image features. It jointly estimates the IMU states and a sliding window of camera poses by an EKF estimator, resulting in a linear complexity in the number of features. MSCKF 2.0 (Li and Mourikis 2013) improved MSCKF and corrected the inconsistent linearized system model that has incorrect observability properties. OKVIS (Leutenegger





**Figure 2.** Differences between perspective and orthogonal projections. The perspective projection (a) projects 3-D points on the principle point, while orthogonal projection (b) projects points on the orthogonal plane. A quadrangular frustum pyramid can be a rectangle when the vertex coincides with the principle point. While for orthogonal image, the size ratio of the two rectangles will not change. This simple property is used for data decoupling.

In direct methods, new images are aligned with several key frames  $I_r$  by minimizing the photometric error  $\mathcal{H}(\mu)$ :

$$\mathcal{H} := \sum_{i,r} \rho(I_r(u'_i) - I(\pi_p(\mathbf{E}'_{C_r W} \pi_p^{-1}(u_i, \mathbf{E}_{C W}))). \quad (3)$$

Similar expressions can be found in (Kerl et al. 2013a; Engel et al. 2014; Newcombe et al. 2011b; Forster et al. 2014; Omari et al. 2015). Note that direct RGB-D SLAM (Kerl et al. 2013a) also minimizes depth error over pixels. The non-robustness to illumination change is one of the potential problems to match pixel intensities directly (Newcombe et al. 2011b). Since (2) and (3) are highly non-linear, iterative numerical solutions are needed.

Iterative solutions also play a significant role in tightly-coupled inertial fusion with the above methods. The inertial residual is imposed into the objective function of ICP (1), reprojection error (2), and photometric error (3), resulting in the RGB-D-inertial (Laidlow et al. 2017), reprojection-inertial (Leutenegger et al. 2015), and direct-inertial (Concha et al. 2016) optimization problems, respectively. Similarly, iterative numerical solutions are needed.

### 3.2 Non-iterative framework

Our proposed non-iterative framework for the front-end shown in Fig. 1 consists of several blocks and will be explained in the Section 4 to 9. A procedure of decoupling-estimation-recoupling is proposed as follows: 3-D point clouds are first decoupled into several subspaces, where data can be matched independently with lower complexity; then the camera motion is obtained from recoupling the estimation in the subspaces. Based on the single key-frame training using kernel cross-correlator (KCC), the translation of the camera pose is predicted by finding the maximum response of the correlation output. This enables the fast motion estimation on ultra-low power processors. Finally, 3-D maps are fused with non-key-frames by a moving

average, so that the missing information of key-frames is complemented with complexity  $\mathcal{O}(n)$ . Unless otherwise stated, the complexity analysis in this paper is in terms of the number of points  $n$  in an image.

## 4 Data decoupling

This section introduces the point cloud decoupling, from 6 DoF to 2 DoF. As shown in Fig. 1, we are introducing a loop where the first block is dependent on the output of the last block, i.e. the rotation estimation from the last cycle is needed for data decoupling. At the beginning, the camera is assumed to be located at the origin.

### 4.1 6 DoF to 3 DoF

It has been shown that even the low-cost, low-precision inertial measurement unit (IMU) can significantly increase performance of the attitude estimation (Konolige et al. 2010). To avoid the traditional optimization problem and decouple the rotational and translational movement, the attitude estimation in Section 9 is applied directly. It is obtained from fusion of back-end and the attitude and heading reference system (AHRS)<sup>1</sup>. A point cloud is a set of points with pose  $\mathbf{G} \in \mathbf{SE}(3)$  and is defined with a specified color to represent an external surface of an object or environment. Assume the  $k_{th}$  key-frame is denoted as  $\mathcal{K}_k(\mathbf{G}_k)$ , a point cloud  $\mathcal{P}_i(\mathbf{G}_i)$  can be rotated to align with the key-frame by its orientation  $\mathbf{R} \in \mathbf{SO}(3)$ . The aligned point cloud  $\tilde{\mathcal{P}}_i$  is obtained through

$$\tilde{\mathcal{P}}_i(\mathbf{t}_i) = \mathbf{R}_k(\mathbf{R}_i)^{-1} \mathcal{P}_i(\mathbf{G}_i), \quad (4)$$

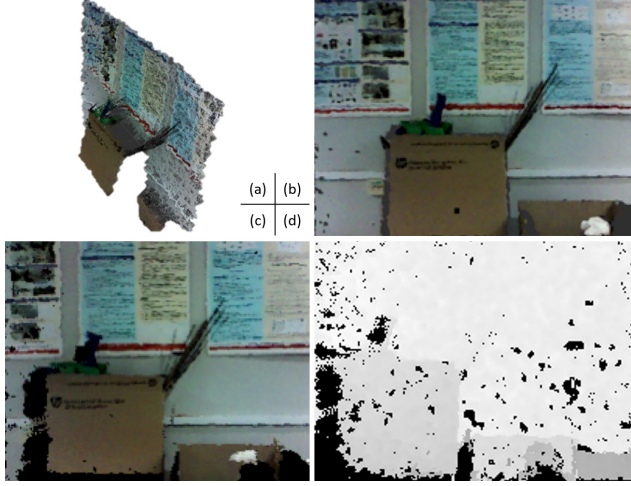
where

$$\mathbf{G}_i = \begin{bmatrix} \mathbf{R}_i & \mathbf{t}_i \\ \mathbf{0} & 1 \end{bmatrix} \text{ with } \mathbf{R}_i \in \mathbf{SO}(3) \text{ and } \mathbf{t}_i \in \mathbb{R}^3. \quad (5)$$

In this sense, the original 6 DoF  $\mathbf{G}_i$  is reduced to 3 translational DoF. From now on, we only need to estimate the translation between point clouds  $\mathcal{P}_i$  and  $\mathcal{K}_k$ .

The idea of estimation after decoupling is proved to be feasible in (Makadia et al. 2006), which leverages on orientation histograms for rough attitude estimation. However, it requires a fine alignment in the final step by running ICP and the translation estimation is based on linear correlation of two point clouds via 3-D Fourier transform with complexity  $\mathcal{O}(n_x n_y n_z \log(n_x n_y n_z))$ , where  $n_x, n_y, n_z$  are the number of voxels in each dimension, respectively. It will be shown that our method is only of complexity  $\mathcal{O}(n \log n)$ , where  $n = n_x n_y$  is the number of points in the image.

One of the potential problems to decouple point clouds by rotation estimation is that the distorted inertial measurements may have a negative influence on translation estimation. For example, magnetic drift or attitude bias can suddenly change in severe situations. In this paper, we



**Figure 3.** A point cloud (a) captured by a depth camera is rectified by the posterior attitude estimation. Image (b) is a perspective image. It is reprojected to an orthogonal color image (c) and depth image (d). Black holes can be seen in (c) and (d), since some missing points in (b) are also projected to the orthogonal image plane.

assume that they are short-term stable, i.e. the bias and drift estimation can roughly follow their real values between two consecutive key-frames. The accumulated long-term drift is corrected by the visual local and global loop closure. Since they are running in the back-end and only performed on key-frames, the system complexity is still dominated by the odometry part. It will be shown that this provides a good compromise between performance and computational complexity, since the non-iterative solution to the odometry system reduces the complexity dramatically yet still with comparable accuracy.

## 4.2 3 DoF to 2 DoF

Computation requirements can be further reduced if the 3-D translational movements are also decoupled. The principle is that the geometry properties in the 3 axes must be kept. We propose to apply orthogonal projection instead of respective projection on the aligned point cloud  $\tilde{\mathcal{P}}_i(\mathbf{t}_i)$  to get orthogonal color and depth images  $\mathbf{x}^C, \mathbf{x}^D \in \mathbb{R}^{n_x \times n_y}$ . A 3-D point  $\mathbf{p} = (x, y, z)^T \in \mathcal{P}$  on the visible surface is mapped to image coordinates  $\mathbf{u} = (u, v)^T \in \Omega$  via the orthogonal projection model  $\pi_a : \mathbb{R}^3 \mapsto \mathbb{R}^2$ , which is defined as:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{1}{r_k} \begin{bmatrix} x \\ y \end{bmatrix}_{\tilde{c}}, \quad (6)$$

where  $r_k \in \mathbb{R}^+$  is the projection resolution that can be set adaptively. The subscript  $\tilde{c}$  denotes that the point coordinates are expressed in the aligned camera frame. The

orthogonal depth image is defined as:

$$\mathbf{x}^D(\mathbf{u}) = [z]_{\tilde{c}}. \quad (7)$$

It stores the point distances to the orthogonal plane versus the perspective depth image that stores the distances to the camera center. This slight difference is illustrated in Fig. 2. The perspective projection cannot keep the geometric relationship, since an object becomes smaller as its distance from the viewpoint increases. While in orthogonal projection, the size ratio of objects is kept the same, regardless their distances. This simplifies the estimation process, since we can now estimate the 2-D translation of orthogonal images followed by 1-D depth translation estimation perpendicular to the plane.

Before that, one thing that needs to be considered is the projection resolution  $r_k$  which is physically the size of covered space by each pixel. The field of view of orthogonal projection is a cuboid space with the central axis coinciding with the camera optical axis. The covered space is determined by the projection resolution  $r_k$  and the image size  $n_x \times n_y$ . Since points outside of the cuboid will be cropped,  $r_k$  is to be set large enough to make the cuboid cover most of the points. Meanwhile, there will be a large black margin on the image, if  $r_k$  is set too large, since not all the pixels can be covered by the projected points. Moreover, the minimum cuboid envelope of a point cloud may change dramatically because of the camera movements.

To be robust to changing scenes, we implement a searching procedure to find the resolution  $r_k$ , so that the cuboid space covers 80% points in the cloud (set empirically, most of the boundary points are outliers). To be efficient, it traverses the cloud by sampling one point in every 25 points. Note that in this process, we keep the central axis of the cuboid envelope coincided with the camera's optical axis. Although the sampling is not precise, we find that the approximation is acceptable and its complexity is only  $\mathcal{O}(n/25)$ , so that the computational time can even be ignored compared to the runtime in the following sections. To further reduce repeated computations, the resolution  $r_k$  is updated only when a key-frame is created. Another advantage of adaptive resolution is that when the camera is near to the scene surfaces, more details can be preserved with smaller  $r_k$ .

Since the point clouds are generated by pinhole cameras, when they are reprojected on the orthogonal plane, some black holes may appear inside the image. This is because some missing points for pinhole cameras are also projected on the orthogonal plane. We will show that they can be filled by the map fusion process presented in Section 8. Fig. 3 shows an example that a 6 DoF point cloud is rectified by rotation estimation and reprojected to orthogonal images.

In the preliminary version (Wang et al. 2017a), a further step for image vectorization is conducted, so that image

translation becomes vector shift. In this paper, we adopt the image translation prediction directly, which is of same complexity with the preliminary version.

## 5 Key-frame training

We propose to apply a non-linear cross-correlator to predict the image translation. Cross-correlators are useful tools and have been used for optical flow (Wang et al. 2018a) and object tracking (Bolme et al. 2010; Henriques et al. 2015). The correlator used in this paper is based on our previous work on kernel cross-correlator (KCC) (Wang et al. 2018b). We start with a brief definition of KCC.

The superscript  $\mathcal{C}, \mathcal{D}$  of orthogonal key-frame and test images  $\mathbf{z}^{\mathcal{C}, \mathcal{D}}, \mathbf{x}^{\mathcal{C}, \mathcal{D}}$  will be left out in the absence of ambiguity. The convolution theorem states that the correlation operation becomes element-wise conjugate multiplication in Fourier domain. Denote the 2-D fast Fourier transformation (FFT)  $\mathcal{F}(\cdot)$  as  $\hat{\cdot}$ , i.e.  $\hat{\mathbf{x}} = \mathcal{F}(\mathbf{x})$ , so that the cross-correlation of two matrices  $\mathbf{g} = \mathbf{x} * \mathbf{h}$  is equivalent to  $\hat{\mathbf{g}} = \hat{\mathbf{x}} \odot \hat{\mathbf{h}}^*$ , where the operator  $\odot$  and superscript  $*$  denote the element-wise multiplication and complex conjugate, respectively. Assume the image size is  $n_x \times n_y$ . The bottleneck of correlation is to compute the forward and backward FFTs, hence the complexity of the entire process has an upper bound  $\mathcal{O}(n \log n)$  where  $n = n_x \times n_y$  is the number of elements. Define the kernel function as  $\kappa : \mathbb{R}^{n_x \times n_y} \times \mathbb{R}^{n_x \times n_y} \mapsto \mathbb{R}$ , such that  $\kappa(\mathbf{x}_i, \mathbf{x}_j) \in \mathbb{R}$ . Assume  $\mathbf{z}_{ij} = \mathcal{S}_{(i,j)}(\mathbf{z})$  is the circular translation of the key-frame  $\mathbf{z}$  with  $(i, j)$  elements. Given a test image  $\mathbf{x} \in \mathbb{R}^{n_x \times n_y}$  and its desired correlation output  $\mathbf{g} \in \mathbb{R}^{n_x \times n_y}$ , the translational case of kernel cross-correlator is defined as:

$$\hat{\mathbf{g}} = \hat{\kappa}_{\mathbf{z}}(\mathbf{x}) \odot \hat{\mathbf{h}}^*, \quad (8)$$

where  $\kappa_{\mathbf{z}}(\mathbf{x})$  is the kernel matrix, with element in  $i_{th}$  row and  $j_{th}$  column as  $\kappa(\mathbf{x}, \mathbf{z}_{ij})$ . More details about full definition of KCC can be found in (Wang et al. 2018b,a).

Given the correlation output  $\mathbf{g}$ , the training objective is to find a filter  $\mathbf{h}$  ( $\mathbf{h} \in \mathbb{C}^{n_x \times n_y}$ ) to satisfy (8). To be efficient, training is conducted in the Fourier domain to take advantage of the simple element-wise operation, i.e. the sum of squared error  $F(\mathbf{h}^*)$  in (9) between the correlation output and the desired output is minimized, i.e.  $\min F(\mathbf{h}^*)$ .

$$F(\mathbf{h}^*) = \|\hat{\kappa}_{\mathbf{z}}(\mathbf{x}) \odot \hat{\mathbf{h}}^* - \hat{\mathbf{g}}\|^2 + \lambda \|\sqrt{\hat{\kappa}_{\mathbf{z}}(\mathbf{z})} \odot \hat{\mathbf{h}}^*\|^2, \quad (9)$$

where the second term is a regularization to prevent overfitting and  $\sqrt{\cdot}$  is element-wise square root operation. Solving the optimization problem requires setting the first derivative to zero, i.e.

$$\nabla F_{\mathbf{h}^*} = 0. \quad (10)$$

Since all the operations in (10) are element-wise, we can simply find a closed-form solution:

$$\hat{\mathbf{h}}^* = \frac{\hat{\mathbf{g}}}{\hat{\kappa}_{\mathbf{z}}(\mathbf{z}) + \lambda}, \quad (11)$$

where the operator  $\div$  denotes the element-wise division.

In this sense, the pattern of translational motion of the key-frame  $\mathbf{z}$  is encoded in the filter  $\mathbf{h}$ . In the experiments, the radial basis function kernel (12) is applied.

$$\kappa(\mathbf{x}, \mathbf{z}_{ij}) = h(\|\mathbf{x} - \mathbf{z}_{ij}\|^2). \quad (12)$$

However, since the complexity of (12) is  $\mathcal{O}(n)$ , the kernel vector  $\kappa_{\mathbf{z}}(\mathbf{x})$  is of complexity  $\mathcal{O}(n^2)$ , which may be on-line infeasible. To compute it efficiently, we first expand the norm in (12):

$$h(\|\mathbf{x} - \mathbf{z}_{ij}\|^2) = h(\|\mathbf{x}\|^2 + \|\mathbf{z}_{ij}\|^2 - 2 \text{Tr}(\mathbf{x}^T \mathbf{z}_{ij})), \quad (13)$$

where  $\text{Tr}(\cdot)$  is the trace operator. Since  $\|\mathbf{x}\|^2$  and  $\|\mathbf{z}_{ij}\|^2$  are constant, the kernel matrix can be calculated as:

$$\kappa_{\mathbf{z}}(\mathbf{x}) = h(\|\mathbf{x}\|^2 + \|\mathbf{z}\|^2 - 2 [\text{Tr}(\mathbf{x}^T \mathbf{z}_{ij})]_{n_x n_y}), \quad (14)$$

where  $[\text{Tr}(\mathbf{x}^T \mathbf{z}_{ij})]_{n_x n_y}$  is a  $n_x \times n_y$  matrix with element  $\text{Tr}(\mathbf{x}^T \mathbf{z}_{ij})$  on position  $(i, j)$ . Because of the correlation theory, we know that  $\mathbf{x} * \mathbf{z} = [\text{Tr}(\mathbf{x}^T \mathbf{z}_{ij})]_{n_x n_y}$ . Substitute it into (14), we have

$$\kappa_{\mathbf{z}}(\mathbf{x}) = h(\|\mathbf{x}\|^2 + \|\mathbf{z}\|^2 - 2 \cdot \mathbf{x} * \mathbf{z}) \quad (15a)$$

$$= h(\|\mathbf{x}\|^2 + \|\mathbf{z}\|^2 - 2 \cdot \mathcal{F}^{-1}(\hat{\mathbf{x}} \odot \hat{\mathbf{z}}^*)). \quad (15b)$$

In this sense, we don't need to calculate  $\mathbf{z}_{ij}$  explicitly, resulting in lower space and computational complexity. The calculation of (15b) is dominated by the forward and backward FFTs, hence the computational complexity of kernel vector  $\kappa_{\mathbf{z}}(\mathbf{x})$  is reduced to  $\mathcal{O}(n \log n)$ . For implementation purpose, the norm in (15b) is computed as (16) in frequency domain using the Parseval's theorem, thus the original images  $\mathbf{z}$  and  $\mathbf{x}$  don't need to be stored.

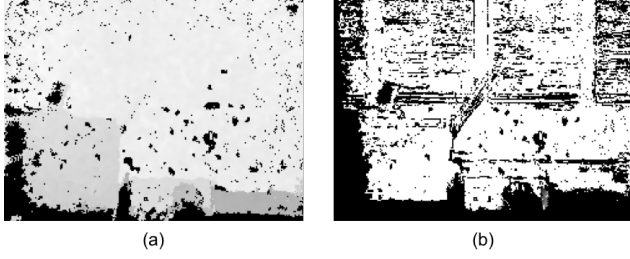
$$\kappa_{\mathbf{z}}(\mathbf{x}) = h\left(\frac{1}{n} \|\hat{\mathbf{x}}\|^2 + \frac{1}{n} \|\hat{\mathbf{z}}\|^2 - 2 \cdot \mathcal{F}^{-1}(\hat{\mathbf{x}} \odot \hat{\mathbf{z}}^*)\right). \quad (16)$$

One of the relevant work of KCC is kernelized correlation filter (KCF) (Henriques et al. 2015). The difference is that KCF is built on ridge regression and is limited to non-weighted kernel functions and circulant training samples. While KCC can be applied to predict affine transformations. More details about this is presented in (Wang et al. 2018b).

## 6 Translation estimation

Based on KCC, the translational pattern of an orthogonal image can be estimated by regarding it as a test sample.





**Figure 4.** The left image (a) is an example of orthogonal depth image from Fig. 3. The corresponding well-matched points are shown in the right image (b) by pixel intensities. The higher the brightness is, the more confidence the matches have.

### 6.1 Image translation estimation

To make the correlation output distinctive, we simply set the first element of  $\mathbf{g}$  as 1, while all the others as 0, i.e.  $g_{[0,0]} = 1$ , where the bracket  $[\cdot]$  is for element indexing (starting from 0). The explanation is that the shift of value 1 in correlation output corresponds to the shift of test image. Because of noises and occlusion, it is impossible to get the exact peak value 1. Instead, the location of the maximum value is used to find the translation:

$$[\tilde{x}, \tilde{y}] = \arg \max_{i,j} \underbrace{\mathcal{F}_{[i,j]}^{-1}(\hat{\kappa}_{\mathbf{z}}(\mathbf{x}) \odot \hat{\mathbf{h}}^*)}_{\mathbf{g}_{[i,j]}(\mathbf{x})} \quad (17)$$

Then the predicted translation  $[\mathbf{t}_x, \mathbf{t}_y]^T$  of orthogonal image is obtained:

$$[\mathbf{t}_x, \mathbf{t}_y]^T = r_k \cdot [\tilde{x}, \tilde{y}]^T. \quad (18)$$

The complexity of (17) is bounded by the calculation of kernel vector and the inverse FFT, hence the process of image translation estimation is of complexity  $\mathcal{O}(n \log(n))$ .

We next present the estimation of covariance  $\sigma_{\mathbf{t}_x}^2$  and  $\sigma_{\mathbf{t}_y}^2$ . Intuitively, the value of each position in the correlation output indicates the estimation confidence of the corresponding translation, thus the estimation covariance can be computed as the covariance of relative weighted translation to the peak. In the experiments, we find that the normalized correlation output  $\mathbf{g}(\mathbf{x}) / \sum \mathbf{g}(\mathbf{x})$  can be approximated by a Gaussian distribution, in which the center is located at the peak (Wang et al. 2017b). Therefore, we approximate the covariance by using the peak value of a standard 2-D Gaussian function, which is  $1/(2\pi\sigma^2)$ , so that

$$\sigma_{\mathbf{t}_x}^2 = \sigma_{\mathbf{t}_y}^2 = \frac{r_k^2 \cdot \sum \mathbf{g}(\mathbf{x})}{2\pi \cdot \max \mathbf{g}(\mathbf{x})}. \quad (19)$$

### 6.2 Depth translation estimation

The overlap of orthogonal image  $\mathbf{x}$  can be matched with key-frame  $\mathbf{z}$ , if  $\mathbf{x}$  is shifted back with  $[\tilde{x}, \tilde{y}]$  elements.

Inspired by this, the camera motion in the depth direction can be estimated in (20a) which averages the depth differences of the matched pixels. To eliminate the influence of dynamic points, it only takes the well-matched pixels that is defined by the set  $\mathcal{W}$  in (20b).

$$\mathbf{t}_z = \frac{1}{|\mathcal{W}|} \sum_{(i,j) \in \mathcal{W}} \left( \mathcal{S}_{(-[\tilde{x}, \tilde{y}])}(\mathbf{x}_{[i,j]}^{\mathcal{D}}) - \mathbf{z}_{[i,j]}^{\mathcal{D}} \right), \quad (20a)$$

$$\mathcal{W} = \left\{ i, j \mid \rho \left( \mathcal{S}_{(-[\tilde{x}, \tilde{y}])}(\mathbf{x}_{[i,j]}^{\mathcal{C}, \mathcal{D}}) - \mathbf{z}_{[i,j]}^{\mathcal{C}, \mathcal{D}} \right) < T_{c,d} \right\}, \quad (20b)$$

where the operator  $|\cdot|$  returns the number of elements in a set and  $\rho(\cdot)$  is a general objective function ( $L_1$ -norm in our tests). The moving objects naturally cannot be matched with shifted image  $\mathcal{S}_{(-[\tilde{x}, \tilde{y}])}(\mathbf{x})$ , thus the parameters  $T_c$  and  $T_d$  are designed to control the elimination of dynamic points.

One of the advantages of (20) is that it only requires the average differences of depth image which is extremely fast to compute and all the well-matched points are able to contribute to the estimation, resulting in the robustness to depth noises. The estimated variance can be obtained via direct calculation:

$$\sigma_{\mathbf{t}_z}^2 = \text{Var}_{(i,j) \in \mathcal{W}} \left( \mathcal{S}_{(-[\tilde{x}, \tilde{y}])}(\mathbf{x}_{[i,j]}^{\mathcal{D}}) - \mathbf{z}_{[i,j]}^{\mathcal{D}} \right). \quad (21)$$

Hence, the complexity of depth translation estimation is  $\mathcal{O}(n)$ . Fig. 4 presents an example for orthogonal depth image and the corresponding well-matched points. Until now, the translation of a point cloud relative to the key-frame is obtained in all three directions.

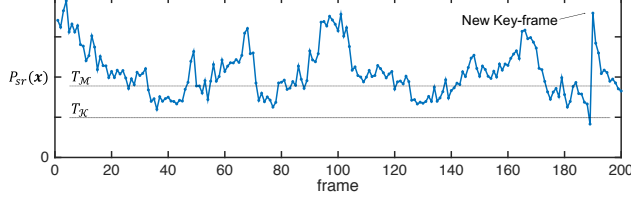
## 7 Key-frame selection

As the camera moves, the overlap between current frame and key-frame may decrease. Hence the peak value of the correlation output may not be distinct enough to determine the translation. In this case, we need to create new key-frames to have better matching quality. Since the new frames will be only matched with their nearest key-frames, estimation error will be accumulated. Therefore, we need to be very careful in creating key-frames. This section presents the conditions for key-frame selection. Section 9 will show that the accumulated error is corrected by loop closure.

Key-frame selection has been studied. For example, DVO (Kerl et al. 2013a) uses the logarithmic determinant of the error distribution covariance matrix to determine key-frames. PTAM (Klein and Murray 2007) creates several conditions: the distance to the last key-frame is large enough; at least twenty key-frames are passed. ORB-SLAM (Mur-Artal et al. 2015) adds the condition: a minimum percentage of tracked features must be reached. However, these conditions are designed in an ad-hoc way and is not suitable for our scenario, since we use different techniques.

We argue that the condition for creating key-frames should be able to represent the matching quality. However,





**Figure 5.** The first 200 frames of a sequence are extracted to show the changes of PSR with respect to frame number. Only when the PSR of the correlation output goes below  $T_K$ , a new key-frame will be created. The point clouds will be fused with the key-frame when PSR is higher than  $T_M$ .

the overlap of two frames is not suitable, since it may not be proportional to the matching quality because of the existence of dynamic objects. Considering the computational cost, we use a very simple criterion in (22), i.e. peak to sidelobe ratio (PSR), which is a measurement of peak strength (Bolme et al. 2010). It splits the correlation output  $\mathbf{g}(\mathbf{x})$  into the peak and the sidelobe which contains the rest of pixels excluding the peak.

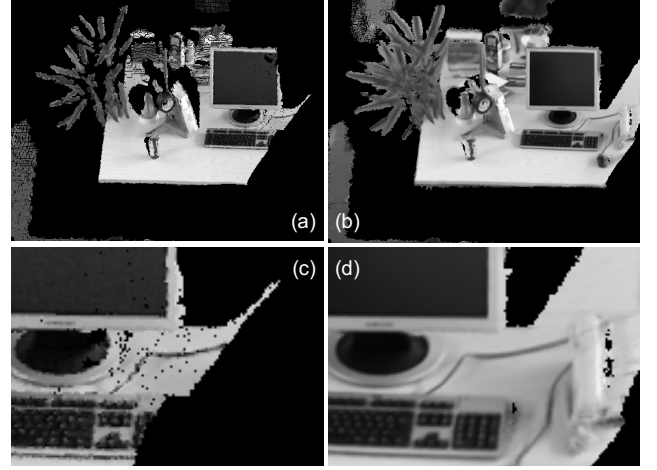
$$P_{sr}(\mathbf{x}) = \frac{\max \mathbf{g}(\mathbf{x}) - \mu_s}{\sigma_s} < T_K, \quad (22)$$

where  $\mu_s$  and  $\sigma_s$  are the mean and standard deviation of the sidelobe. A straightforward explanation is that the peak strength of the correlation output indicates the matching confidence level. In this sense, the desired output with single peak has infinite peak strength.  $P_{sr}(\mathbf{x})$  is regarded as a trigger to insert a new key-frame into the map when it is smaller than  $T_K$ . The criterion (22) is not only able to control the minimum confidence of each matching, but also save computational time, especially when the camera is kept still or moving slowly since there is no new training data is required. Only the calculation of mean and standard deviation of the sidelobe are performed in (22), hence the complexity for key-frame selection is  $\mathcal{O}(n)$ .

Another important capability of (22) is to control the inertial drift compensation. In case of sudden magnetic interference or dynamic inertial biases, (22) will result in creation of more key-frames; (19) and (21) will result in larger estimation covariances. As a result, the back-end optimization will have more information for finer relocalization, which has a greater weight in pose graph, resulting in higher chance of recovering the true inertial states. More details about back-end optimization and loop closure will be presented in Section 9.

## 8 Map refinement and fusion

To reduce the complexity, only the key-frames are selected to represent a map. As mentioned earlier, not all the pixels of orthogonal images can be reprojected by valid points,



**Figure 6.** (a) is a new key-frame. (b) is the refined key-frame in real-time. Tests show that the "black holes" in the original orthogonal image can be filled by subsequent matched images based on (23). (c) and (d) are the same part of the new and refined key-frame in (a) and (b), respectively. Note that the keyboard is well complemented. The moving average operation will not make it blurred, as long as the image translation is estimated correctly.

and some black holes may appear. A very simple idea is to complement the missing information of the key-frame  $\mathbf{z}$  by the matched non-key-frames  $\mathbf{x}_k$ . This is useful when the criterion (22) is not satisfied, i.e. the camera keeps still or moves slowly.

We first define a weight vector  $\mathbf{w} \in \mathbb{R}^{n_x \times n_y}$  for each frame. The element of  $\mathbf{w}$  presents the weight of the corresponding pixel to be fused. Each time a frame  $\mathbf{x}_k$  is acquired, its corresponding weight vector  $\mathbf{w}_k^x$  is initialized as a binary matrix, i.e.  $\mathbf{w}_k^x \leftarrow \{0, 1\}^n$  where 1 or 0 indicates whether that pixel can be fused or not.

The weight vector  $\mathbf{w}^z$  for the key-frame is the element-wise summation of  $\mathbf{w}_k^x$  that has been fused. Initially,  $\mathbf{w}^z$  is simply set as  $\mathbf{w}_k^x$  when  $\mathbf{x}_k$  is selected as a new key-frame. Therefore, the map fusion process can be expressed as a moving average in (23) which is performed in both color and depth images.

$$\mathbf{s}_k \leftarrow \mathbf{w}^z + \mathcal{S}_{(-[\tilde{x}, \tilde{y}])}(\mathbf{w}_k^x) + e, \quad (23a)$$

$$\mathbf{z}^c \leftarrow \frac{(\mathbf{w}^z \odot \mathbf{z}^c + \mathcal{S}_{(-[\tilde{x}, \tilde{y}])}(\mathbf{w}_k^x \odot \mathbf{x}_k^c))}{\mathbf{s}_k}, \quad (23b)$$

$$\mathbf{z}^D \leftarrow \frac{(\mathbf{w}^z \odot \mathbf{z}^D + \mathcal{S}_{(-[\tilde{x}, \tilde{y}])}(\mathbf{w}_k^x \odot (\mathbf{x}_k^D - \mathbf{t}_z)))}{\mathbf{s}_k}, \quad (23c)$$

$$\mathbf{w}^z \leftarrow \mathbf{w}^z + \mathcal{S}_{(-[\tilde{x}, \tilde{y}])}(\mathbf{w}_k^x), \quad (23d)$$

where  $e$  is a small scalar ( $10^{-7}$ ) to prevent division by 0.

Because of the dynamic points and outliers, some valid points in  $\mathbf{x}_k$  can not be matched with the key-frame. Hence, we remove the unmatched points before (23) is performed. The elements of  $\mathbf{w}_k^x$  corresponding to the unmatched points

are set to 0 as in (24), so that they cannot be fused into the key-frames.

$$\mathbf{w}_{k[i,j]}^{\mathbf{x}} \leftarrow 0, \text{ if } (i, j) \notin \mathcal{W}, \quad (24)$$

where the set  $\mathcal{W}$  defined in (20b) contains the indexes of all matched points. To obtain a higher quality map, we perform (23) only when  $P_{sr}(\mathbf{x}_k) > T_{\mathcal{M}}$  where  $T_{\mathcal{M}}$  is a parameter to control the confidence of map fusion and  $T_{\mathcal{M}} > T_{\mathcal{K}}$ .

The PSR value plays an important role in our algorithm. Fig. 5 presents an example for the changes of PSR in terms of frames. This sequence is extracted from the first 200 frames of freiburg1\_desk from TUM dataset (Sturm et al. 2012). It can be seen that when the matching confidence is lower than  $T_{\mathcal{K}}$ , a new key-frame will be created, so that the matching confidence returns to a high value.

Since all the operations in (23) are element-wise, the complexity of map fusion process is  $\mathcal{O}(n)$ . Fig. 6 gives an example where a key-frame is refined by non-key-frames. It demonstrates that the missing information of key-frames can be complemented, so that more details are preserved.

## 9 Loop closure

### 9.1 Relocalization

The relocalization is checked among key-frames. To “memorize” the visual appearances, we extract and save 1000 ORB features (Rublee et al. 2011) for every key-frames. Since ORB feature descriptor is binary, it allows quick comparison by direct Hamming distance. Moreover, in order to perform quick image retrieval for relocalization, Bag-of-Words model (Nister and Stewenius 2006) is applied to re-represent ORB key-frame features by visual words. Moreover, these visual words are taken from a pretrained dictionary that is similar to the one deployed in ORB-SLAM2. The vocabulary, pretrained with Bovis 2008-09-01 dataset (Bonarini et al. 2006), contains 1 million words, arranged in 6 levels, each with 10 clusters (Mur-Artal and Tardós 2017). The relocalization mechanism is implemented based on the excellent work of DBoW2. Firstly, potential match pairs  $\langle b_t, b_{t-j} \rangle$ , where  $b_t$  is the bag of key-frame  $t$ , are identified by a normalized similarity score (Galvez-Lopez and Tardos 2012).

$$\eta(b_t, b_{t-j}) = \frac{s(b_t, b_{t-j})}{s(b_t, b_{t-1})}, \quad (25)$$

where  $s$  is L1-score between two bags:

$$s(b_i, b_j) = 1 - \frac{1}{2} \left| \frac{b_i}{|b_i|} - \frac{b_j}{|b_j|} \right|. \quad (26)$$

These candidates then undergo a second test to determine if their relative transformation and PSR value are within a threshold.

### 9.2 Pose graph optimization

To correct accumulated error, we optimize a pose graph upon loop closure detection. A pose graph, with key-frames as  $\mathbf{SE}(3)$  nodes interconnected by the relative transform and covariance estimation, is incrementally constructed as new key-frames are inserted. Once confident relocalization is available, we minimize the sum of squared error over the global key-frame trajectory using the Lavenberg-Marquardt algorithm. Our pose-graph optimization scheme leverages on the existing implementation of  $g^2o$  (Kummerle et al. 2011). Different from the preliminary version (Wang et al. 2017a), where the pose is recoupled by loosely coupled fusion, the back-end pose graph can correct both inertial and odometry drift. Moreover, since only key-frames are involved, the runtime is dominated by the odometry part.

In the experiments, we find that such feature-based method is sometimes sensitive to the pseudo-features which are produced by light reflections via pieces of glass, resulting in wrong trajectory optimization. To overcome this problem, we implement a simple false relocalization detection by comparing with the front-end odometry. If the results of graph optimization is too far away from the odometry, we simply reject such relocalization.

## 10 Experimental results

Throughout this paper, a set of non-iterative algorithms have been designed to reduce the complexity of the front-end of a RGB-D-inertial SLAM system. We will evaluate the system both quantitatively and qualitatively in terms of dense mapping, trajectory estimation and runtime efficiency. We also test the performance on micro-devices.

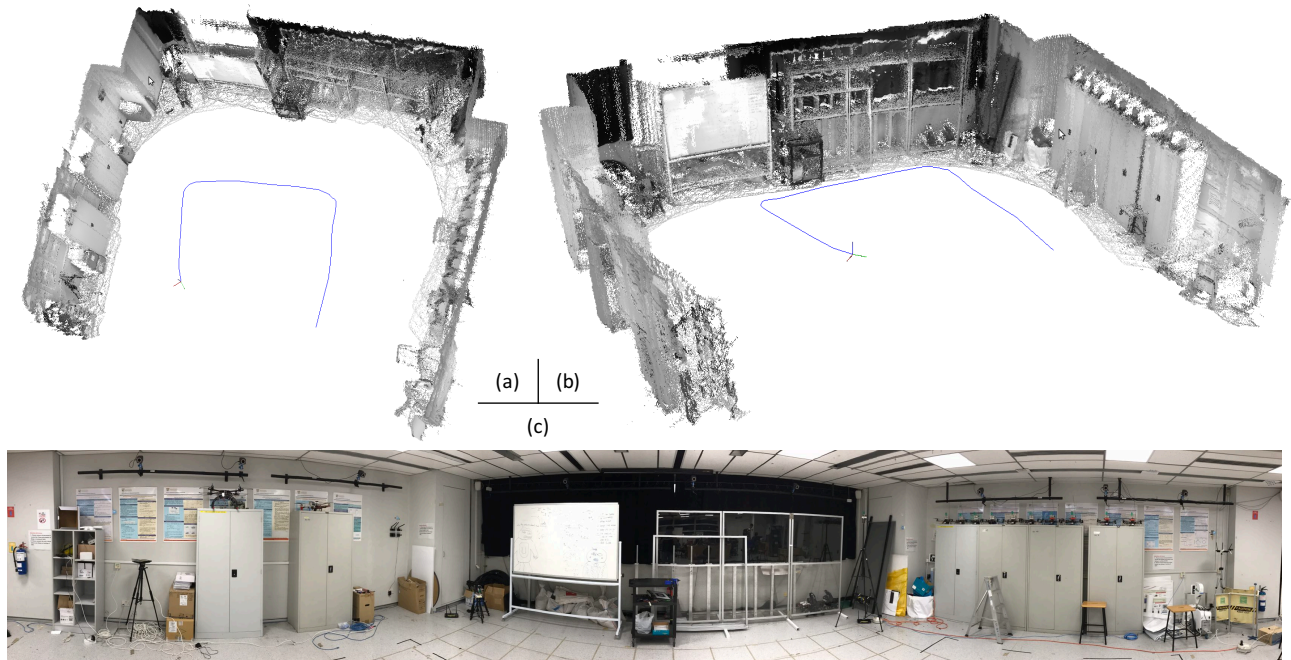
### 10.1 Implementation

The framework is implemented using FFTW3 (Frigo and Johnson 2005) and Eigen (Guennebaud and Jacob 2010) libraries for FFT and matrix calculation. The system runs in ROS (Quigley et al. 2009) and is pre-calibrated using the Kalibr toolbox (Furgale et al. 2013). The inertial and visual measurements are simply synchronized using the ROS timestamp. Due to the communication channel delay, this will introduce a short time difference. However, we find that the performance is acceptable. We expect that a hardware triggering synchronization produces better performance.

In the experiments, the Gaussian kernel is applied in (12) with a standard deviation of 0.2 pixels. The regularization term  $\lambda$  in solution (11), the matching difference parameters  $T_c$  and  $T_d$  in (20b) are all set to 0.1. It is found that these parameters are not sensitive to the test environments, since different scenarios and sensors are tested and the results are not much affected by different choices of these parameters. The PSR thresholds  $T_{\mathcal{K}}$  and  $T_{\mathcal{M}}$  are set to 50 and 100, respectively.

**Table 1.** Simulation performance of ATE RMSE, mean, and median error on TUM dataset. ‘Dynamic’ means whether dynamic objects are contained in the corresponding sequence.

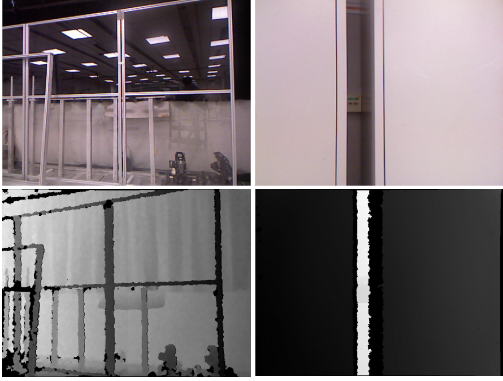
Dataset	RMSE (m)	Mean (m)	Median (m)	Std (m)	$\bar{v}$ (m/s)	$\bar{\omega}$ (°/s)	Dynamic
freiburg1_rpy	0.062	0.054	0.049	0.031	0.062	50.15	
freiburg1_xyz	0.011	0.010	0.008	0.006	0.244	8.92	
freiburg1_desk	0.025	0.024	0.022	0.009	0.413	23.32	
freiburg1_desk2	0.055	0.048	0.041	0.028	0.426	29.31	
freiburg1_plant	0.103	0.091	0.079	0.049	0.365	27.89	
freiburg1_room	0.128	0.112	0.098	0.062	0.334	29.88	
freiburg2_rpy	0.025	0.021	0.018	0.013	0.014	5.77	
freiburg2_xyz	0.012	0.011	0.010	0.005	0.058	1.72	
freiburg3_cabinet	0.078	0.076	0.076	0.021	0.216	10.25	
freiburg3_large_cabinet	0.106	0.096	0.091	0.066	0.362	8.75	
freiburg3_sitting_rpy	0.036	0.027	0.021	0.023	0.042	23.84	✓
freiburg3_sitting_xyz	0.031	0.029	0.028	0.012	0.132	3.56	✓
freiburg3_sitting_static	0.006	0.005	0.006	0.003	0.011	1.70	✓
freiburg3_walking_xyz	0.056	0.044	0.037	0.035	0.208	5.49	✓
freiburg3_walking_static	0.024	0.018	0.014	0.016	0.012	1.39	✓
freiburg3_walking_halfsphere	0.115	0.094	0.078	0.066	0.221	8.27	✓
freiburg3_long_office_household	0.033	0.031	0.030	0.011	0.249	10.19	
freiburg3_structure_texture_near	0.019	0.018	0.017	0.006	0.141	7.68	
freiburg3_structure_texture_far	0.014	0.013	0.012	0.005	0.193	4.32	

**Figure 7.** Figure (a) and (b) are the dense reconstruction of our lab. The panorama of this room is given in Figure (c). This environment is quite challenging for feature-based methods, since there are many feature-less area and light reflections produced by pieces of glass.

## 10.2 Simulation on TUM dataset

We first evaluate the performance of trajectory estimation using the TUM RGB-D benchmark (Sturm et al. 2012)

that provides synchronized ground truth from a motion capture system, which is also used to simulate inertial measurements in the tests. The accuracy in terms of



**Figure 8.** The examples of color (first row) and depth (second row) images, in which lots of specular reflections and featureless regions can be found.

absolute trajectory (ATE) root-mean-square error (RMSE), mean error, and median error are listed in Table 1, where the standard deviation of RMSE, maximum error, average velocity and angular rate are also given. More details about the calculation of these metrics can be found in (Sturm et al. 2012). In comparison, all estimated camera poses are associated with the ground truth by timestamp.

Comparable performance on this dataset can be found in state-of-the-art works, e.g. Volumetric Fusion (Whelan et al. 2015a), RGB-D SLAM2 (Endres et al. 2014), ElasticFusion (Whelan et al. 2016), and ORB-SLAM2 (RGB-D) (Mur-Artal et al. 2015; Mur-Artal and Tardós 2017). Although we cannot see a significant improvement on accuracy (we get better results in some sequences, but the numbers are still on the same magnitude), our framework can run much faster ( $2\times-4\times$ ), and perform well on some scenarios which are challenging for other methods, as seen in the following sections.

### 10.3 Dense mapping

In this section, we present the qualitative performance of in-door dense reconstruction. Fig. 7 presents the 3-D map of our lab, that is generated in real-time and shown from different point of view in (a) and (b), respectively. The full view of this room is shown in Fig. 7 (c) where large featureless area and visual pseudo-features produced by pieces of glass can be seen. This environment is quite challenging for feature-based methods, since the pseudo-features will move along with the camera movement, which will produce wrong pose estimation. For our method, the kernel cross-correlator matches the point cloud as a whole, regardless the distinct features. Therefore, our method is insensitive to the visual corner outliers and work better in such situations. Note that the back-end of our system sometimes also produces false relocalization due to the

feature outliers, while most of them are rejected by the front-end odometry stated in Section 9.2.

In (Wang et al. 2017b), we also constructed the dense map for the same environment, which requires the aids from external ultra-wideband (UWB) anchors. Due to the comprehensive improvements in this paper, e.g. the dynamic resolution mechanism, the external sensors are not needed any more. The performance of other methods in such environment is presented in next section.

## 10.4 Performance on benchmark

**10.4.1 Dataset and platform** The experimental benchmark<sup>2</sup> contains 7 sequences that are totally 39.7 GB and recorded in an indoor environment. The ground truth is obtained from a highly accurate motion capture system (Vicon). Except for the ground truth, this benchmark include measurements from Kinect at 30Hz, inertial measurements from myAHRS+<sup>3</sup> at 100Hz, which is a low cost high performance inertial sensor containing a 16-bit 3-axis gyroscope, a 16-bit 3-axis accelerometer, and a 13-bit 3-axis magnetometer. Some of the image examples are shown in Fig. 8. They are quite challenging for vision-based methods because there are lots of specular reflections and featureless regions. All the experiments are conducted on a standard PC running Ubuntu with an Intel Core i7-4700 CPU and 8G RAM.

**10.4.2 Baseline Algorithms** The proposed odometry system will be compared with some state-of-the-art methods, i.e. RGB-D SLAM2 (Endres et al. 2014), ElasticFusion (Whelan et al. 2016), and ORB-SLAM2 (RGB-D) (Mur-Artal et al. 2015; Mur-Artal and Tardós 2017). Their performance is obtained using the same platform mentioned in Section 10.4.1 based on their open source implementation.

Although these methods do not include inertial fusion, they are still excellent baseline and have been extensively tested by the robotic community. The implementation of RGB-D-inertial odometry method, e.g. (Laidlow et al. 2017) is unavailable, hence we are unable to compare. The traditional tightly-coupled or loosely-coupled inertial fusion methods typically introduce additional complexity due to the additional filtering or optimization process for the large number of inertial measurements. However, we will show that the proposed non-iterative inertial fusion method reduce the complexity of pure visual system dramatically. For accuracy, it will be shown that the proposed method doesn't reduce the performance of pure visual methods and even performs better in some scenarios that are challenging for traditional methods.

Since ElasticFusion has no ROS interface, a simple ROS wrapper is implemented to read images through ROS communication protocol, while the runtime measurement will not include the communication time. Some of the above methods optimize the whole trajectory in the back-end,

**Table 2.** Root mean square error (RMSE) comparison on the benchmark. Only instant pose estimation is accepted.

Benchmark	ours		ORB-SLAM2		ElasticFusion		RGB-D SLAM2	
	RMSE	std.	RMSE	std.	RMSE	std.	RMSE	std.
01	0.106	0.038	0.323	0.095	0.439	0.165	0.122	0.060
02	0.373	0.230	-	-	0.602	0.185	0.575	0.417
03	0.123	0.055	0.142	0.079	0.444	0.171	0.191	0.057
04	0.076	0.037	0.058	0.029	0.123	0.058	0.124	0.055
05	0.093	0.040	0.513	0.219	0.281	0.121	-	-
06	0.113	0.045	0.128	0.057	0.355	0.139	0.543	0.155
07	0.021	0.009	0.025	0.015	0.024	0.012	0.031	0.022

**Table 3.** Mean error and maximum error comparison on the benchmark. Only instant pose estimation is accepted.

Benchmark	ours		ORB-SLAM2		ElasticFusion		RGB-D SLAM2	
	MEAN	MAX	MEAN	MAX	MEAN	MAX	MEAN	MAX
01	0.099	0.261	0.308	0.508	0.407	0.688	0.106	0.248
02	0.294	1.020	-	-	0.572	1.067	0.397	1.527
03	0.110	0.300	0.118	0.380	0.410	0.778	0.182	0.389
04	0.066	0.211	0.051	0.150	0.109	0.256	0.111	0.470
05	0.084	0.183	0.464	1.003	0.254	0.557	-	-
06	0.103	0.209	0.114	0.352	0.327	0.591	0.520	0.840
07	0.019	0.053	0.021	0.065	0.021	0.050	0.022	0.186

**Table 4.** Real-time efficiency comparison on Kinect dataset with PC platform. All the data are give by average update rate (Hz).

Dataset	ours	ORB-SLAM2	RGB-D SLAM2
01	73.4	41.1	18.7
02	82.2	-	20.1
03	81.6	41.5	18.3
04	80.7	45.6	20.9
05	77.2	42.7	21.1
06	77.8	46.6	21.6
07	85.1	50.0	23.3
overall	79.7	44.6	20.6

and output the optimized trajectory only after the program is terminated. This is not suitable for low-latency robotic systems. To evaluate and compare real-time performance, we only accept the instant pose estimation.

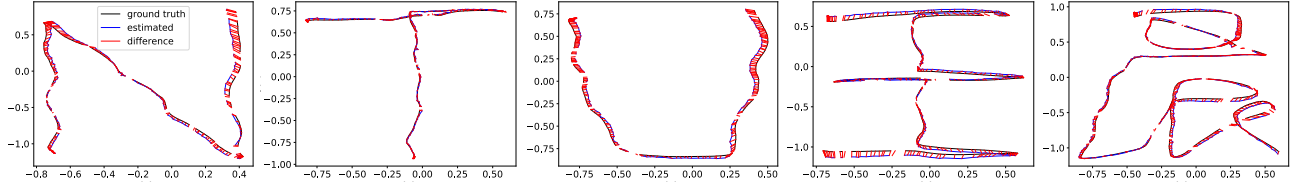
**10.4.3 Accuracy Comparison** The accuracy is evaluated in terms of ATE RMSE and absolute mean error (MEAN). The comparison is reported in Table 2 and Table 3. It can be seen that the proposed method achieves similar accuracy with all the other methods. We notice that the accuracy of feature-based methods, ORB-SLAM2 and RGB-D SLAM2 vary a lot in different datasets, this may be due to the existence of specular reflection and features-less regions. This phenomenon indicates that the feature-based methods

are sensitive to feature outliers. In contrast, ElasticFusion performs similar for different datasets. The results of some sequences that are obviously wrong are not reported, this is because we find those methods fail to give a well pose optimization due to feature outliers.

It is noticed that the accuracy of ORB-SLAM2 (RGB-D) is higher when true visual local loop closure is available frequently, while our method works better in feature-less and pseudo-feature environments. One of the reasons may be that the KCC used in our method is more robust to object distortion and occlusion. In this sense, the proposed non-iterative framework is a good alternative solution to the front-end of RGB-D-inertial SLAM system.

**10.4.4 Efficiency Comparison** The efficiency performance is evaluated in terms of update rate (runtime), which is reported in Table 4. The proposed framework outperforms all the other methods. Note that the reported runtime of our method is the summation of both tracking and mapping process; while that of the other methods only contain the tracking time. This is because the tracking thread is independent of the mapping in other methods, while they are processed sequentially in ours. The runtime of our method varies for different sequences according to the number of trainings. If landmarks change rapidly, our method has to train new models, resulting in a little bit increasing of runtime.





**Figure 9.** The trajectory estimation an ultra-low power CPU Atom x5-Z8350 with Scenario Design Power (SDP) of 2W.



**Figure 10.** A dense mapping of office cubicles. It is constructed in real-time on the ultra-low power CPU Atom x5-Z8350. The map is sent to a remote server in real-time only for visualization purpose (no post-processing).

Note that we don't report the efficiency of ElasticFusion in Table 4, since it provides only GPU implementation, while the other methods are running on pure CPU. The platform mentioned in Section 10.4.1 contains a NVIDIA Quadro 2000 GPU, in which the average update rate for ElasticFusion is about 5.1 Hz. We believe that ElasticFusion is able to run faster if more powerful GPU is provided, while the efficiency comparison is not meaningful.

**Table 5.** Average runtime (ms) on the ultra low power platform.†

Processor	Method	Tracking	Mapping	Update
x5-Z8350	ours	26.9	5.4	32.3
	ORB2	167	499	$\geq 167$

†All tests are using the RGB-D image with size  $640 \times 480$ . The mapping process of ORB-SLAM2 is parallel to tracking.

## 10.5 Performance on micro-device

**10.5.1 Platform** The real-time performance on low power device is crucial for micro-robot systems. As shown before, the proposed system requires the least computational resources, thus we are able to test it on micro-devices. We choose the Intel robotic development kit, that includes RealSense R200 RGB-D camera and a credit card sized computational board equipped with an ultra-low power processor Atom x5-Z8350. We also integrate it with an inertial sensor myAHRS+ which is mentioned in Section 10.4.1. Running at 1.44GHz with 2G RAM, this platform is very difficult for most of the state-of-the-art dense mapping algorithms to run in real-time (30Hz).

**10.5.2 Performance** Limited by the on-board memory, we cannot evaluate the performance using benchmarks. Instead, we will test the real-time performance in terms of average runtime, trajectory estimation, and dense mapping.

(i) We list the average tracking and mapping runtime in Table 5, where the performance of ORB-SLAM2 (RBG-D) is also given for comparison. It can be seen that our system requires much less runtime compared with ORB-SLAM2. The tracking and mapping process of ORB-SLAM2 is parallel to each other and cannot be simply summed up.

(ii) Some examples of the overhead trajectory estimation is shown in Fig. 9, where the ground truth is from the Vicon system. The estimated poses are associated with the truth values by the red lines according to the ROS timestamp. We cannot report the accuracy comparison with ORB-SLAM2, since it cannot work in real-time (a lot of dropped frames).

(iii) An example for real-time dense mapping of our working area (cubicles) is shown in Fig. 10. The map is constructed and fused real-time by the on-board processor. Meanwhile, the key-frames (orthogonal images) are sent to a remote server for visualization without post-processing. Some black holes can be seen inside the map, the reasons could be that the inside IR sensors seem to have no

response to some specific dark material, resulting in no depth information. Since the key-frame fusion algorithm in (23) requires the complementary information from non-key-frames, it cannot work in such situation.

## 11 Conclusion

In this paper, we proposed a non-iterative front-end for the RGB-D-inertial SLAM system. Combining with non-iterative front-end and a lightweight iterative back-end, our system dramatically reduces the computational complexity. It was demonstrated that the odometry system can be accelerated based on an on-line training process. To the best of our knowledge, the proposed framework may be the first non-iterative solution to the front-end of a dense mapping system. We also demonstrated that it is able to achieve much faster speed and comparable accuracy with the state-of-the-art methods.

## Acknowledgements

The authors would like to thank the help from Dr. Chun-lin Chen, Mr. Junjun Wang, and Mr. Xu Fang in some experiments.

## Notes

1. <https://en.wikipedia.org/wiki/AHRS>
2. [https://wangchen.online/ni\\_slam](https://wangchen.online/ni_slam)
3. [https://wangchen.online/myahrs\\_driver](https://wangchen.online/myahrs_driver)

## References

- Bailey T and Durrant-Whyte H (2006) Simultaneous Localization and Mapping (SLAM): Part II. *Robotics & Automation Magazine* 13(3): 108–117.
- Besl PJ and McKay ND (1992) A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14: 1–18.
- Bolme D, Beveridge JR, Draper BA and Lui YM (2010) Visual Object Tracking Using Adaptive Correlation Filters. In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 2544–2550.
- Bonarini A, Burgard W, Fontana G, Matteucci M, Sorrenti DG and Tardos JD (2006) Rawseeds: Robotics advancement through web-publishing of sensorial and elaborated extensive data sets. In: *In proceedings of IROS*, volume 6.
- Clark R, Wang S, Wen H, Markham A and Trigoni N (2017) VINet: Visual-Inertial Odometry as a Sequence-to-Sequence Learning Problem. *arXiv.org* : arXiv:1701.08376.
- Concha A, Loianno G, Kumar V and Civera J (2016) Visual-Inertial Direct SLAM. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1331–1338.
- Costante G, Mancini M and Valigi P (2016) Exploring Representation Learning With CNNs for Frame-to-Frame Ego-Motion Estimation. *IEEE Robotics and Automation Letters* 1(1): 18–25.
- Dai A, Nießner M, Zollhöfer M and Izadi S (2017) BundleFusion: Real-time Globally Consistent 3D Reconstruction using On-the-fly Surface Reintegration. *ACM Transactions on Graphics* 36(3): 1–18.
- Davison AJ, Reid ID, Molton ND and Stasse O (2007) MonoSLAM: Real-Time Single Camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29(6): 1052–1067.
- DeTone D, Malisiewicz T and Rabinovich A (2017) Toward Geometric Deep SLAM. *arXiv.org* : arXiv:1707.07410.
- Endres F, Hess J, Sturm J, Cremers D and Burgard W (2014) 3-D Mapping With an RGB-D Camera. *IEEE Transactions on Robotics* 30(1): 177–187.
- Engel J, Koltun V and Cremers D (2017) Direct Sparse Odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence* .
- Engel J, Schöps T and Cremers D (2014) LSD-SLAM: Large-Scale Direct Monocular SLAM. In: *European Conference on Computer Vision*. Switzerland: Springer International Publishing, pp. 834–849.
- Engel J, Stuckler J and Cremers D (2015) Large-Scale Direct SLAM with Stereo Cameras. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 1935–1942.
- Engelhard N, Endres F and Hess J (2011) Real-time 3D visual SLAM with a hand-held RGB-D camera. *Proc. of the RGB-D Workshop on 3D Perception in Robotics at the European Robotics Forum, Vasteras, Sweden* 180.
- Forster C, Carlone L, Dellaert F and Scaramuzza D (2015) IMU Preintegration on Manifold for Efficient Visual-Inertial Maximum-a-Posteriori Estimation. In: *Robotics: Science and Systems*. Robotics: Science and Systems Foundation.
- Forster C, Pizzoli M and Scaramuzza D (2014) SVO: Fast Semi-Direct Monocular Visual Odometry. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 15–22.
- Fraundorfer F and Scaramuzza D (2012) Visual Odometry: Part II: Matching, Robustness, Optimization, and Applications. *Robotics & Automation Magazine* 19(2): 78–90.
- Frigo M and Johnson SG (2005) The design and implementation of FFTW3. *Proceedings of the IEEE* 93(2): 216–231. Special issue on “Program Generation, Optimization, and Platform Adaptation”.
- Furgale P, Rehder J and Siegwart R (2013) Unified temporal and spatial calibration for multi-sensor systems. In: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, pp. 1280–1286.
- Galvez-Lopez D and Tardos JD (2012) Bags of Binary Words for Fast Place Recognition in Image Sequences. *IEEE Transactions on Robotics* 28(5): 1188–1197.
- Guennebaud G and Jacob B (2010) Eigen v3. <http://eigen.tuxfamily.org>.



- 
- Gutierrez-Gomez D, Mayol-Cuevas W and Guerrero JJ (2016) Dense RGB-D Visual Odometry Using Inverse Depth. *Robotics and Autonomous Systems* 75: 571–583.
- Henriques JF, Caseiro R and Martins P (2015) High-Speed Tracking with Kernelized Correlation Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37(3): 583–596.
- Henry P, Krainin M, Herbst E, Ren X and Fox D (2012) RGB-D Mapping: Using Kinect-Style Depth Cameras for Dense 3D Modeling of Indoor Environments. *International Journal of Robotics Research* 31(5): 647–663.
- Jaimez M and Gonzalez-Jimenez J (2015) Fast Visual Odometry for 3-D Range Sensors. *IEEE Transactions on Robotics* 31(4): 809–822.
- Ji Y, Yamashita A and Asama H (2015) RGB-D SLAM Using Vanishing Point and Door Plate Information in Corridor Environment. *Intelligent Service Robotics* 8(2): 105–114.
- Kerl C, Sturm J and Cremers D (2013a) Dense Visual SLAM for RGB-D Cameras. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 2100–2106.
- Kerl C, Sturm J and Cremers D (2013b) Robust odometry estimation for RGB-D cameras. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 3748–3754.
- Khan S and Wollherr D (2015) IBUILD: Incremental bag of Binary words for appearance based loop closure detection. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 5441–5447.
- Klein G and Murray D (2007) Parallel Tracking and Mapping for Small AR Workspaces. *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality* : 225–234.
- Konolige K, Agrawal M and Solà J (2010) Large-Scale Visual Odometry for Rough Terrain. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Kummerle R, Grisetti G, Strasdat H, Konolige K and Burgard W (2011) G2O: a General Framework for Graph Optimization. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 3607–3613.
- Laidlow T, Bloesch M, Li W and Leutenegger S (2017) Dense RGB-D-Inertial SLAM with Map Deformations. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 6741–6748.
- Leutenegger S, Lynen S, Bosse M, Siegwart R and Furgale P (2015) Keyframe-Based Visual-Inertial Odometry Using Nonlinear Optimization. *International Journal of Robotics Research* 34(3): 314–334.
- Li M and Mourikis AI (2013) High-Precision, Consistent EKF-Based Visual-Inertial Odometry. *International Journal of Robotics Research* 32(6): 690–711.
- Lynen S, Achtelik MW, Weiss S, Chli M and Ingénieur mécanicien RYS (2013) A Robust and Modular Multi-Sensor Fusion Approach Applied to MAV Navigation. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.
- Ma L, Falquez JM, McGuire S and Sibley G (2016) Large scale dense visual inertial slam. *Field and Service Robotics* .
- Makadia A, Patterson AI and Daniilidis K (2006) Fully Automatic Registration of 3D Point Clouds. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1 (CVPR'06)*. IEEE, pp. 1297–1304.
- Mohanty V, Agrawal S, Datta S, Ghosh A, Sharma VD and Chakravarty D (2016) DeepVO: A Deep Learning approach for Monocular Visual Odometry. *arXiv.org* : arXiv:1611.06069.
- Mourikis AI and Roumeliotis SI (2015) A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 3565–3572.
- Mur-Artal R, Montiel JMM and Tardos JD (2015) ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics* 31(5): 1147–1163.
- Mur-Artal R and Tardós JD (2017) ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Transactions on Robotics* : 1–8.
- Newcombe RA, Davison AJ, Izadi S, Kohli P, Hilliges O, Shotton J, Molyneaux D, Hodges S, Kim D and Fitzgibbon A (2011a) KinectFusion: Real-Time Dense Surface Mapping and Tracking. In: *2011 IEEE International Symposium on Mixed and Augmented Reality*. IEEE, pp. 127–136.
- Newcombe RA, Fox D and Seitz SM (2015) DynamicFusion: Reconstruction and Tracking of Non-Rigid Scenes in Real-Time. In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 343–352.
- Newcombe RA, Lovegrove SJ and Davison AJ (2011b) DTAM: Dense Tracking and Mapping in Real-Time. In: *IEEE International Conference on Computer Vision*. IEEE, pp. 2320–2327.
- Nister D and Stewenius H (2006) Scalable recognition with a vocabulary tree. In: *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, volume 2. Ieee, pp. 2161–2168.
- Omari S, Bloesch M, Gohl P and Siegwart R (2015) Dense Visual-Inertial Navigation System for Mobile Robots. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 2634–2640.
- Quigley M, Conley K, Gerkey B, Faust J, Foote T, Leibs J, Wheeler R and Ng AY (2009) Ros: an open-source robot operating system. In: *ICRA workshop on open source software*, volume 3. Kobe, Japan, pp. 1–6.
- Raguram R, Frahm JM and Pollefeys M (2008) A Comparative Analysis of RANSAC Techniques Leading to Adaptive Real-Time Random Sample Consensus. In: *Computer Vision – ECCV 2008*. Berlin, Heidelberg: Springer, Berlin, Heidelberg,

- pp. 500–513.
- Rublee E, Rabaud V, Konolige K and Bradski G (2011) ORB: An efficient alternative to SIFT or SURF. In: *IEEE International Conference on Computer Vision*. IEEE, pp. 2564–2571.
- Sturm J, Engelhard N, Endres F, Burgard W and Cremers D (2012) A benchmark for the evaluation of rgb-d slam systems. In: *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, pp. 573–580.
- Tateno K, Tombari F, Laina I and Navab N (2017) CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction. *arXiv.org* : arXiv:1704.03489.
- Valencia R and Andrade-Cetto J (2018) *Mapping, planning and exploration with Pose SLAM*. Springer.
- Wang C, Ji T, Nguyen TM and Xie L (2018a) Correlation Flow: Robust Optical Flow using Kernel Cross-Correlators. In: *International Conference on Robotics and Automation (ICRA)*. IEEE.
- Wang C, Yuan J and Xie L (2017a) Non-Iterative SLAM. In: *2017 18th International Conference on Advanced Robotics (ICAR)*. Hong Kong: IEEE, pp. 83–90.
- Wang C, Zhang H, Nguyen TM and Xie L (2017b) Ultra-Wideband Aided Fast Localization and Mapping System. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 1602–1609.
- Wang C, Zhang L, Xie L and Yuan J (2018b) Kernel Cross-Correlator. In: *AAAI Conference on Artificial Intelligence*.
- Weiss S and Siegwart R (2011) Real-Time Metric State Estimation for Modular Vision-Inertial Systems. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 4531–4537.
- Whelan T, Kaess M, Fallon M and Johannsson H (2012) Kintinuous: Spatially extended kinectfusion. *RSS Workshop on RGB-D Advanced Reasoning with Depth Cameras*.
- Whelan T, Kaess M, Johannsson H, Fallon M, Leonard JJ and McDonald J (2015a) Real-Time Large-Scale Dense RGB-D SLAM with Volumetric Fusion. *International Journal of Robotics Research* 34(4-5): 598–626.
- Whelan T, Leutenegger S, Salas Moreno R, Glocker B and Davison A (2015b) ElasticFusion: Dense SLAM Without A Pose Graph. In: *Robotics: Science and Systems*. Robotics: Science and Systems Foundation.
- Whelan T, Salas-Moreno RF, Glocker B, Davison AJ and Leutenegger S (2016) ElasticFusion: Real-time dense SLAM and light source estimation. *International Journal of Robotics Research*.