Autonomous Robots (2020) 44:115–130
https://doi.org/10.1007/s10514-019-09881-0

DVL-SLAM: sparse depth enhanced direct visual-LiDAR SLAM

Young-Sik Shin[1] · **Yeong Sang Park**[1] · **Ayoung Kim**[1]

Received: 11 June 2018 / Accepted: 20 July 2019 / Published online: 6 August 2019

Abstract
This paper presents a framework for direct visual-LiDAR SLAM that combines the sparse depth measurement of light detection and ranging (LiDAR) with a monocular camera. The exploitation of the depth measurement between two sensor modalities has been reported in the literature but mostly by a keyframe-based approach or by using a dense depth map. When the sparsity becomes severe, the existing methods reveal limitation. The key finding of this paper is that the *direct* method is more robust under sparse depth with narrow field of view. The direct exploitation of sparse depth is achieved by implementing a joint optimization of each measurement under multiple keyframes. To ensure real-time performance, the keyframes of the sliding window are kept constant through rigorous marginalization. Through cross-validation, loop-closure achieves the robustness even in large-scale mapping. We intensively evaluated the proposed method using our own portable camera-LiDAR sensor system as well as the KITTI dataset. For the evaluation, the performance according to the LiDAR of sparsity was simulated by sampling the laser beam from 64 to 16 and 8. The experiment proves that the presented approach is significantly outperformed in terms of accuracy and robustness under sparse depth measurements.

Keywords Direct visual SLAM · Camera-LiDAR · Sparse depth

1 Introduction

For the last two decades, simultaneous localization and mapping (SLAM) has received steadily increasing attention in robotics research. It has been advanced by studies using various sensor systems, from the monocular camera to the dense RGB-D camera. For the 3D perception of the outdoor field robotic system, the sensor system that combines LiDAR and camera has attracted attention as a practical sensor type. This sensor system reconstructs a 3D spatial structure through SLAM, which enables various tasks for field robots, such as path planning, obstacle avoidance and robot manipulation. However, there is little focus on how to enhance visual SLAM by exploiting the relatively sparse depth measurements from LiDAR.

Most field robotic systems in the literature rely on multiple sensors (e.g., cameras, LiDAR and radar) for navigation and perception. In particular, the expectation for the commercialization of autonomous vehicles has led to the introduction of new types of LiDAR, such as solid-state LiDAR with a sparser and narrower field of view (FOV), and the increase in demand is expected to lead to mass production of cost-effective LiDAR. The depth measurement of LiDAR elaborates on visual images in SLAM by eliminating scale ambiguity and thus enhancing geometric information to the image. However, when LiDAR provides only sparse measurements, the co-visible volume in the FOV of the two sensors becomes small, thereby, encumbers the LiDAR measurements association for enhancing visual SLAM.

In recent years, there has been considerable progress in the field of visual SLAM using a monocular camera. The classical visual feature-based method has already matured resulting in a stable visual SLAM method. Recently, *direct* methods have been introduced that directly use the intensity of the image (Forster et al. 2014; Engel et al. 2014, 2017; Newcombe et al. 2011b; Kerl et al. 2013b; Steinbrücker et al. 2011). These methods showed remarkable motion estima-

A portion of the paper was presented in part in Shin et al. (2018).

✉ Ayoung Kim
ayoungk@kaist.ac.kr

Young-Sik Shin
youngsik.shin@kaist.ac.kr

Yeong Sang Park
pys0827k@kaist.ac.kr

[1] Korea Advanced Institute of Science and Technology, Daejeon, Korea

tion performance despite not using the correspondence of visual features. However, there are some significant challenges to visual SLAM with the monocular system. As the camera moves along the optical axis, the pixel position of the visual feature is almost invariant, which allows estimation of the rotation but estimation of the translational motion is up to scale. In order to overcome the scale ambiguity of the monocular system, these methods inevitably require a bootstrap to initialize the 3D environment (Gauglitz et al. 2012; Civera et al. 2008; Forster et al. 2014). Eventually, the quality of this initialization dominates the scale error with the real world.

There were approaches to integrate IMU measurements to overcome the limitations of the monocular system. The use of the IMU allows metric scale observations but requires scalable initialization due to lack of depth measurement. In addition, the motion and environment are weakly observable or present difficulty in initializing depth information in a degenerate condition (Yang and Shen 2017; Qin et al. 2018). Also, a stereo system is used to solve the scale ambiguity of this monocular vision system. However, precise calibration is required to estimate precise depth measurement from a stereo camera. In addition, a configuration with a large baseline is essential to obtain depth information at long distances (Sibley et al. 2007) and triangulating the depth of a long distance requires a sensor configuration of the large baseline. In stereo cameras, the disparity error affects the estimated depth, and as the length of the baseline increases, the overlap between the two cameras decreases and the amount of available information decreases (Pinggera et al. 2014). The error of the depth measurement obtained from LiDAR is relatively small and is not affected by extrinsic calibrating quality as with a stereo camera. The independent use of a stereo camera or LiDAR to perform metric-level SLAM has attracted a great deal of attention.

More recent efforts are found from the learning-based approaches where visual SLAM methods have incorporated the deep learning-based depth estimation to solve the inherent issue of the monocular camera (Wang et al. 2017; Tateno et al. 2017; Yang et al. 2018). Although learning-based depth prediction provides dense depth, the error increases as the depth estimation range increases, and still yields a significant error in the meter level (Yang et al. 2018; Kuznietsov et al. 2017). Also, they need a pre-trained model for depth prediction to deploy them in SLAM applications and additional adjustment is required to use the model directly for camera models with different intrinsic parameters. These approaches have only been evaluated in the driving sequence or in the indoor scenarios, further investigation is needed to see how generalized they are in various environments and platforms (Yang et al. 2018). For this reason, we aim to combine LiDAR's ability to perform highly accurate depth measurements with motion estimation of direct visual SLAM.

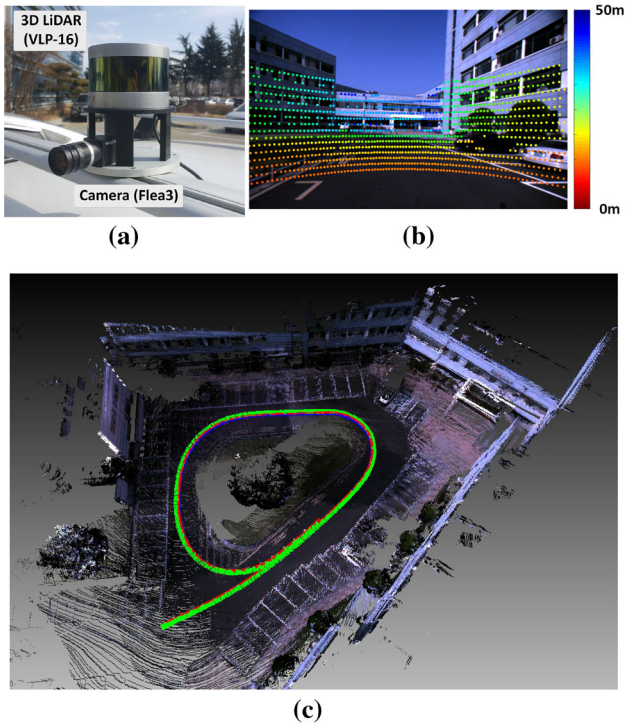

Fig. 1 A sensor system combined with camera and LiDAR for motion estimation and 3D mapping. **a** A camera-LiDAR sensor system that acquires the image and sparse depth measurement. The data acquired by the sensor is projected onto the image as shown in **b**. Using this sensor configuration, the proposed method performs precise pose estimation and 3D mapping. The points are color-coded with respect to the scene depth. **c** Estimated trajectory and 3D reconstruction result

This paper proposes a visual SLAM system[1] that combines the sparse depth measurement of LiDAR with the intensity of the camera image and utilizes the direct method without using visual features. Note that, as shown in Fig. 1, only sparse depth measurements are available from LiDAR in a limited area of the imagery. Although these sparse measurements can be associated with visual features, their utility is diminished by the lack of meaningful features in blurred images. For this reason, we conclude that the direct method is effective for handling sparse depth measurements with the images. In addition, *direct* SLAM is known to provide more accurate motion estimation results in low-resolution cameras (Engel et al. 2017). However, the narrow FOV of sparse 3D LiDAR remains an issue. It only allows for an association of depth and image in the partial image region. To overcome this limitation, we integrate the depth of neighboring frames into the keyframe and perform motion estimation using multiple keyframes. We also perform loop detection and correction to obtain a drift-free result on a large-scale.

This paper proposes a *direct* visual SLAM method for a Camera-LiDAR system under sparse depth. The main

[1] Code is available at http://github.com/irapkaist/dvl_slam.

characteristics of the proposed method can be explained as follows:

- We propose a *direct* visual SLAM for when the sparse depth of LiDAR is available within a narrow FOV. Since the sparse measurement is directly used in the image domain, it requires no visual features.
- A window-based optimization overcomes the limitation caused by the sparsity of the LiDAR and different FOV of heterogeneous sensors.
- In large-scale environments, loop-closing techniques are used for the global map consistency. We provide a robust loop constraint by performing cross-validation on loop candidates.
- The proposed method is thoroughly evaluated through real experiments using our own sensor system and public datasets containing various environments.

The rest of this paper is organized as follows. Section 2 discusses research related to visual SLAM from single camera to depth-enabled configuration. Section 3 presents an overview of the proposed direct visual-LiDAR SLAM. Section 4 explains the details of the proposed method. The experimental results are shown in Sect. 5 and we discuss the conclusion of this study.

2 Related work

2.1 Feature-based visual SLAM

The visual SLAM is a technique for performing motion estimation and mapping using sequential camera images. There has been a lot of studies in visual SLAM, most of which concentrate on using corner-like visual features such as Scale Invariant Feature Transform (SIFT), peeded Up Robust Features (SURF) and Oriented FAST and Rotated BRIEF (ORB) (Klein and Murray 2007; Kitt et al. 2010; Strasdat et al. 2011; Mur-Artal et al. 2015).

Davison et al. (2007) successfully performed extended Kalman filter (EKF)-based visual SLAM in real-time using only a monocular camera in a room-sized environment. However, these filtering-based approaches are computationally inefficient because they have to perform operations in consecutive frames and accumulate larger linearization errors than Bundle Adjustment (BA)-based approaches. On the other hand, keyframe-based approaches construct a map using only selected frames, and BA optimization accurately perform, although the computation complexity is somewhat higher. Parallel Tracking and Mapping (PTAM), proposed by Klein and Murray (2007), is the most representative keyframe-based visual SLAM system. It is the first study to divide the visual SLAM system into parallel threads for camera tracking and mapping. They have demonstrated successful performance in a small environment with an aim toward augmented reality application. Mur-Artal et al. (2015) proposed a more representative feature-based SLAM method called ORB-SLAM. They added a loop thread to the tracking and mapping framework to make them more robust and accurate in a large-scale environment. The methods proposed in Gálvez-López and Tardós (2012) and Strasdat et al. (2011) were used to deal with loop detection and scale drift. However, the drawback of these feature-based methods it that they have difficulty finding correspondences when the environment is a simple and repeated pattern or is a featureless condition. Eventually, these environmental conditions result in the motion estimation failure.

2.2 Direct visual SLAM

Recently, a *direct* method has been introduced without requiring the concrete feature correspondence, and still showing remarkable performance. Instead of using the reprojection errors of correspondences, the *direct* method considers the intensity difference of a pixel as a photometric error model (Engel et al. 2014, 2017; Forster et al. 2014; Newcombe et al. 2011b; Pizzoli et al. 2014). However, good initialization is required for convergence of the optimization, and a massive computation using the GPU is needed to estimate the motion and the full-depth map in real-time (Newcombe et al. 2011b; Pizzoli et al. 2014). In order to reduce this computational burden, Forster et al. (2014) and Engel et al. (2014) proposed semi-dense approaches. They used a certain number of patches, or only used areas of image gradient magnitude above a threshold, to reduce computation complexity, thus ensuring that the direct method runs on the CPU in real-time. Engel et al. (2017) proposed a state-of-the-art direct visual odometry method called Direct Sparse Odometry (DSO). DSO performs window-based optimization including all relevant parameters (e.g., camera intrinsics, motion and inverse depth value). They maintain a constant number of points and frames to achieve real-time performance and low drift without loop-closure. However, the issue of scale ambiguity remains a challenge like any monocular SLAM, and the performance is often determined by bootstrapping to initialize the depth.

2.3 RGB-D SLAM

The RGB-D camera has received great attention in the field of visual SLAM (Newcombe et al. 2011a; Kerl et al. 2013a, b; Whelan et al. 2015). These cameras provide not only RGB images but also dense depth measurement for each pixel within a limited range. Endres et al. (2014) formulated the motion estimation problem given the RGB image and depth information as a 3D feature matching problem. Huang et al.

(2011) tracked the visual feature associated with depth and used it for motion estimation. Henry et al. (2012) combined visual feature-based methods with Iterated Closest Point (ICP). They employed visual features with associated depth to obtain an initial alignment, and then jointly optimize 2D and 3D alignment errors. Newcombe et al. (2011a) proposed the KinectFusion algorithm that uses an implicit surface model of the scene, known as the truncated signed distance function (TSDF). The dense model of the scene is adopted for motion estimation with ICP. Also, there are studies applying the direct method (Steinbrücker et al. 2011; Kerl et al. 2013a, b). These methods minimize photometric errors that aim to find the optimal rigid body motion to register two consecutive RGB-D frames. Whelan et al. (2015) proposed a motion estimation that combined ICP using TSDF volume with the photometric error. These methods, using RGB-D cameras as Kinect, require dense depth measurement for visual SLAM. The use of such sensors is limited in open environments due to limited short depth measurements (i.e., up to 5 m) in the environments and a triangulation is often leveraged to overcome these limitations (Concha and Civera 2017).

2.4 LiDAR-based SLAM

The LiDAR sensor is widely adopted in many computer vision and robotics studies due to their precise range measurement and the invariance to the perceptual aliasing. The ICP-based methods mainly selected for LiDAR, which are classical scan matching methods, have evolved and focused on dense point cloud data in the latest studies to efficiently utilize them (Segal et al. 2009; Serafin and Grisetti 2015). However, such methods fail to perform 3D mapping due to the sparsity of vertical measurements when providing sparse measurements from low-cost 3D LiDAR. For example, a 16-beam 3D LiDAR has 16 rings in the vertical direction and provides thousands of points per ring. Because of the vertical sparsity, estimation of pitch motion and the vertical translation are particularly problematic, causing the failure in the 3D mapping. Zhang and Singh (2017) proposed a motion estimation method by extracting sharp edges and a planar surface from LiDAR points. They have achieved significant accuracy by extending this geometric feature-based motion estimation method to map-to-scan matching. However, this method does not guarantee its performance in environments without geometric features.

Another LiDAR-based SLAM combines sweeping 2D LiDAR with the inertial measurement unit (IMU) (Bosse et al. 2012; Park et al. 2018). These sensor systems are asynchronous and operate at a high rate, and 3D mapping becomes more challenging with the motion distortion of LiDAR. Bosse et al. (2012) successfully solved the challenge by applying a continuous-time trajectory to SLAM. However, the continuous-time SLAM suffers from computational complexity in long-term operation because of a global batch trajectory optimization. Park et al. (2018) proposed a map-centric LiDAR fusion method. Surfel-based fusion and map deformation not only maintain global consistency but also reduce map noise. By limiting continuous-time trajectory optimization to local mapping, the method also guarantees real-time performance.

2.5 Visual-LiDAR SLAM

Lu (2016) proposed a visual-LiDAR slam method called VELO which is a SLAM method using multiple cameras and LiDAR. Lu tightly coupled the sparse visual odometry and the LiDAR scan matching an objective function and achieved globally consistent 3D mapping, including loop closure. The method performs frame-to-frame motion estimation by combining ICP constraints from LiDAR and 3D-3D, 3D-2D and 2D-2D visual constraints from consecutive stereo images. However, to perform both visual feature matching and ICP, expensive tree-based data structures are required. Moreover, the performance of this method is not guaranteed because it does not take into account the motion distortion of LiDAR points as the sensor moves.

Zhang et al. (2017) proposed a method called Depth Enhanced Monocular Odometry (DEMO) that handles sparse depth measurement from LiDAR. DEMO deals with sensor configuration similar to ours (i.e., a monocular camera with LiDAR). However, the method associates the visual feature with depth measurement and uses it to estimate the motion of the camera. Gräter et al. (2018) proposed a combination of a feature-based visual odometry method and a high-accuracy depth measurement for precise vehicle motion estimation. We enabled metric-scale motion estimation by interpolating tracked visual features with depth measurements and utilizing a keyframe-based bundle adjustment to improve local accuracy. However, to associate visual features and depth measurements, the expensive 64-ray LiDAR has been utilized and its accuracy is not guaranteed when using more sparse 3D LiDAR. The proposed method is similar to theirs in that the limitation of the monocular camera is overcome by combining it with LiDAR; however, our focus is on using the *direct* method to handle depth sparsity without using visual features. We also differ from other methods using dense depth (Newcombe et al. 2011a; Kerl et al. 2013b; Steinbrücker et al. 2011; Whelan et al. 2015) by handling sparse measurements of LiDAR directly on a single camera. These sensor configurations are applicable to most field robotic systems, but they have not been actively studied because of the differences between heterogeneous sensors. To the best of our knowledge, our method is the first method to apply sparse measurement from LiDAR to the direct method.

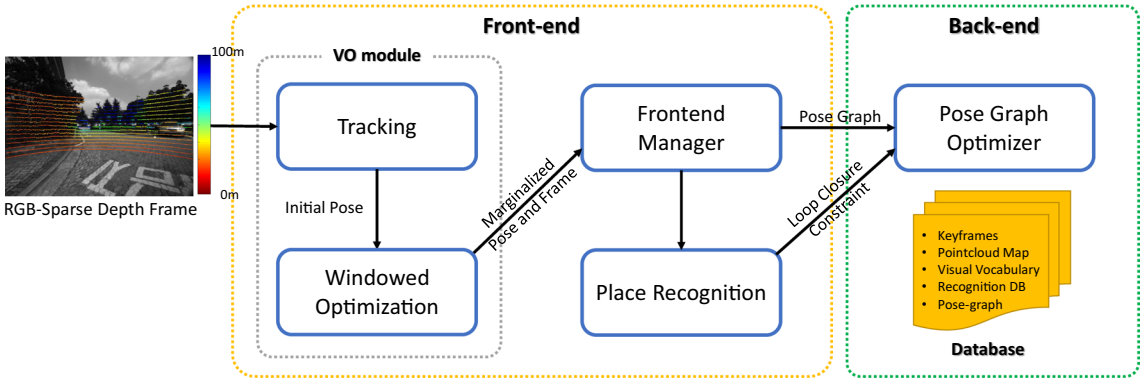

Fig. 2 Block diagram of the direct visual-LiDAR SLAM. The proposed method consists of the front-end for data association and back-end solving for the global pose-graph optimization including loop constraint

3 Algorithm overview

The proposed method consists of a front-end and a back-end similar to the general SLAM approaches. The input of the algorithm is sequential image data and a sequential LiDAR scan. Given an image with an associated sparse depth, only the image is used for the tracking process, so we compensate for the distortion of LiDAR caused by motion after the initial tracking is done. Then the module takes a point sampling strategy for fast and robust motion estimation. It keeps the complexity of the constant time by reducing the number of points used in the algorithm. The front-end focuses on accurate motion estimation using windowed optimization and data association for loop-closing as shown in Fig. 2. LiDAR provides precise depth measurement of up to 100 m. We can only optimize the pose of multiple keyframes because the disparity error on the image cannot guarantee the accuracy of depth measurement of the long distance and is computationally expensive. The associated data received from the front-end is used for global pose graph optimization in the back-end. The description of each part of the overall system is as follows:

Tracking The visual odometry (VO) module performs the tracking process for fast motion estimation without visual features. The initial motion is estimated using only a small number of sampled points, and the sampled points are directly tracked by the intensity in the image domain. The tracking occurs at a fast rate without requiring corner-like feature extraction and matching, unlike indirect methods. In addition, using only LiDAR-associated points eliminates a triangulation phase to estimate the depth. This simplicity benefits the various robotics systems with cameras and LiDAR. Similar to other direct approaches, the image pyramid and constant motion model are used in this process (Forster et al. 2014; Engel et al. 2017; Kerl et al. 2013b).

LiDAR undistortion When initial motion is obtained by the tracking process and LiDAR scan data is obtained, we can compensate for that distortion by motion interpolation. In the case of VLP-16, since scan data can be obtained in each packet unit, we compensate for distortion of LiDAR points using interpolated motion based on poses of camera frames. When undistortion of LiDAR points is performed, we associate them with the closest camera frame.

Windowed optimization Once the tracking process has been successfully performed, the current frame is added to the sliding window followed by optimization to improve local accuracy within the window. This refinement allows the motion from the tracking process to maintain the low drift.The oldest frames exceeding a user-defined number (N_w) are marginalized. The marginalized keyframe is then fed into the back-end pose graph.

Place recognition The final component of the front-end is the place recognition module. This module confirms whether the position of the marginalized keyframe is revisited and then integrates depth information from adjacent frames to perform a two-view alignment. This depth integration solves ambiguity when a large displacement situation occurs between two candidate frames.

Pose graph optimizer The back-end module optimizes the global pose graph, including the current motion and the loop constraints provided by the front-end. We used *g2o* (Kuemmerle et al. 2011) with an incremental sparse linear algebra for the online operation of the optimization.

4 Direct visual-LiDAR SLAM

4.1 Notation

A frame $\mathcal{F}_i$ consists of an image $\mathcal{I}_i$ and a pointcloud $\mathcal{P}_i$ from a camera-LiDAR sensorsystem. Each point $\mathbf{p}_k = (x, y, z)^\top \in$

$\mathbb{R}^3$ of $\mathcal{P}_i$ is observed as a pixel $\mathbf{u_k} = (u, v) \in \mathbb{R}^2$ in the coordinates of the image plane through a projection function $\pi(\mathbf{p}_k)$. If the tracking process has been performed, then each frame has its associated pose as a transformation matrix $\mathbf{T}_{wi} \in SE(3)$ in the world coordinate.

Lie algebra elements $\boldsymbol{\xi} \in \mathbb{R}^6$ are used as minimal representations for pose optimization on $SE(3)$ throughout this paper, where the element $\xi = [\mathbf{w}, \mathbf{v}]$ appears as both an angular and linear velocity vector. The lie algebra element is mapped to $SE(3)$ by exponential mapping, and its inverse is obtained by log mapping. This relationship is used to update the pose matrix through the increment obtained by lie algebra: $se(3) \boxplus SE(3) \rightarrow SE(3)$.

$$\xi \boxplus \mathbf{T} := \exp(\hat{\xi})\mathbf{T} \tag{1}$$

When tracking between frame $\mathcal{F}_i$ and frame $\mathcal{F}_j$, the relative pose transformation can be denoted by $\mathbf{T}_{ji} = \mathbf{T}_{wj}^{-1}\mathbf{T}_{wi}$ and this relationship can be used to transform the coordinates of the points as follows:

$$\mathbf{p'}_j = \mathbf{T}_{ji}\mathbf{p}_i = \begin{bmatrix} \mathbf{R}_{ji} & \mathbf{t}_{ji} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{p}_i \tag{2}$$

where $\mathbf{p}_i$ is a point on the frame $\mathcal{F}_i$ and $\mathbf{p'}_j$ is an observed point on the frame $\mathcal{F}_j$. $\mathbf{R}_{ji} \in SO(3)$ and $\mathbf{t}_{ji} \in \mathbb{R}^3$ are a rotation matrix and translational vector.

The widely adopted pinhole camera model is used to project the transformed point $\mathbf{p'}_j$ of frame $\mathcal{F}_i$ to frame $\mathcal{F}_j$. Let $\pi(\cdot)$ be the projection function onto the image plane:

$$\pi(\mathbf{p'}_j) = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \end{bmatrix} \begin{bmatrix} \frac{x'_j}{z'_j} \\ \frac{y'_j}{z'_j} \\ 1 \end{bmatrix} \tag{3}$$

where f_x and f_y are focal length and c_x and c_y are the principal points.

4.2 Frame management

4.2.1 Frame generation

The proposed method defines points of LiDAR and the associated image as a frame. Given a camera image in the next sequence, we perform tracking while optimizing the 6-DOF motion. To associate the data of two sensors as a frame, the relative position between two sensors must be known. The camera operates at a high frequency (30 Hz), while the 3D LiDAR operates at a low frequency (10 Hz). The proposed method compensates for the distortion of LiDAR points by motion based on the poses from camera images obtained in the tracking process. Given the closest two camera poses $\mathbf{T}_k$ and $\mathbf{T}_{k+1}$, the linear interpolated pose $\mathbf{T}_\tau$ is defined as:

$$\mathbf{T}_\tau = \mathbf{T}_k \exp(\lambda \xi_{k,k+1}) \tag{4}$$

where their timestamps satisfy $\tau_k < \tau < \tau_{k+1}$. $\xi_{k,k+1} = \log(\mathbf{T}_k^{-1}\mathbf{T}_{k+1})$ is used to interpolate pose T_τ with interpolation ratio $\lambda = (\tau - \tau_k)/(\tau_{k+1} - \tau_k)$. Finally, based on interpolate poses, LiDAR scan points are compensated and associated with the nearest camera frame. Our system records the time at which the laser passes the center of the camera, and uses the image taken at the closest time to associate the camera frame and LiDAR. Once the two sensor data are associated, the 3D points of the LiDAR coordinates are transformed into camera coordinates, allowing an image and 3D points of input frames to be placed in the camera coordinates.

4.2.2 Keyframe management

The sliding window maintains a fixed number of keyframes to achieve the constant processing time for the VO module. The most recent keyframe in the sliding window is tracked by the current image of the input frame. The strategy of using a sliding window in the odometry module improves local accuracy and reduces computational load. When managing the keyframes of the sliding window, the following two criteria are used:

- When the tracking process is successful, a new keyframe can be added to the window by checking the ratio of projectable points. Since the FOV of the camera and the LiDAR is fixed, the reduction of the projected points means a loss of information. We prevent this loss of information by maintaining a certain ratio of projectable points. Any frame with a ratio less than γ_{kf} is immediately registered in the sliding window.
- The viewpoint change can be resolved through the ratio of the projectable points, but the influence of the illumination change remains. Even if the viewpoint change is not detected, we add the keyframe after checking the brightness change every time using a from (5).

4.2.3 Keyframe marginalization

To alleviate the computational burden of the windowed optimization, our VO module maintains its frames by marginalizing the oldest keyframe from the sliding window. After the tracking process, whenever a new keyframe is added, marginalization is performed according to the following strategy:

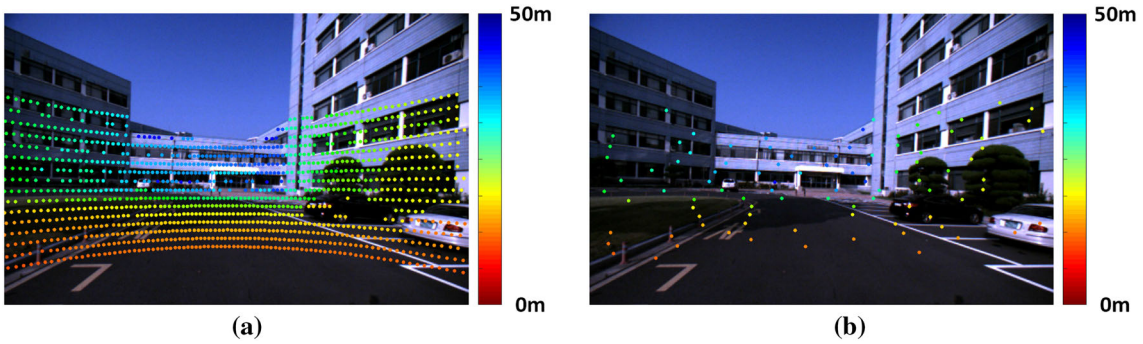

Fig. 3 Salient point selection. **a** The entire LiDAR measurement projected onto the image without selection. **b** The selected salient points. For computational efficiency, only salient points are used for motion estimation

– The window maintains at least two keyframes. If the ratio of visible points exceeds γ_{kf}, a new frame is added to the sliding window.
– If the number of keyframes exceeds the window size (N_w) defined by the user, a new keyframe is added and the oldest keyframe is marginalized.

We use the Schur complement to marginalize the oldest keyframe. The remaining Hessian blocks can be used by the optimizer repeatedly when a new keyframe is added. The marginalized keyframe is excluded from the window, but it is added as a node of the global pose graph and used in the mapping phase. Single camera based direct methods optimize both the points and the poses after the tracking process Engel et al. (2017). Since the range measurements of LiDAR are fairly accurate, our method only estimates the poses in the window for computational efficiency.

4.2.4 Point management

Tracking all points acquired from LiDAR is unnecessary. The measurements from LiDAR may include points in the low texture region, where the points are not easily trackable on images. In addition, since there is less overlapping area than the dense RGB-D sensor, using a patch of sparse pattern like `pattern 4` in DSO helps to ensure robustness rather than pixel-wise tracking and the use of these sparse pattern-patch guarantees sufficient performance while requiring slightly fewer pixels (Engel et al. 2017). However, such patch-based tracking requires appropriate management, as the computational complexity increases with the size of the patch.

Although LiDAR measurements are already sparse, we use even fewer, but more informative, measurements through salient point selection to increase computation efficiency. To sample the salient points, we focus on (i) the spatial distribution of LiDAR data in the spherical coordinate and (i) the gradient of the image. We divide azimuth and elevation by two degrees and use them as a spatial bin. Then, the point with the largest gradient in the image is selected as the salient point. The points are spatially well-distributed because they are selected from spatial bins with a constant interval. When the gradient of the point is too small, such point is excluded due to their low information. Figure 3 shows an example of selected salient points. The LiDAR measurements are uniformly distributed in the image and are thus more sparse.

4.3 Tracking

The proposed method performs tracking to obtain an initial guess of the camera pose whenever a new frame is acquired. Inspired by the sparse direct method (Engel et al. 2017), we focus on fast and robust motion estimation without handling visual features.

For ease of notation, the latest keyframe is referred to as the reference frame $\mathcal{F}_i$ and the current frame is referred to as the target frame $\mathcal{F}_j$. To estimate relative transformation between two frames, the points $\mathbf{P}_i$ in $\mathcal{F}_i$ are projected into the target frame $\mathcal{F}_j$ and the difference of intensity at the patch of the sparse pattern Ω is defined as a photometric error. Although these patch-based approaches require more computation than pixel-wise approaches, they ensure robustness against image degradation, such as motion blur and noise. In addition, we used the affine illumination model with parameters a and b to cope with the changes in illumination (Jin et al. 2001). Parameters a and b of the affine illumination model in Eq. (9) are solved in closed form at each iteration by Engel et al. (2015). The photometric residual is defined as:

$$r(\mathbf{p}_i) = \mathcal{I}_j(\pi(\mathbf{T}_{ji}\mathbf{p}_i)) - (a\mathcal{I}_i(\pi(\mathbf{p}_i)) + b). \tag{5}$$

Finally, the objective function for tracking is defined using the residuals of the points $\mathbf{P}_i$ in the frame $\mathcal{F}_i$ as follows:

$$\underset{\mathbf{T}_{ji}}{\operatorname{argmin}} \frac{1}{2} \sum_{\mathbf{p}_k \in \mathbf{P}_i} \sum_{i \in \Omega(\mathbf{p}_k)} \rho(r(\mathbf{p}_i)) \left(r(\mathbf{p}_i)^2\right), \tag{6}$$

where the function $\rho(\cdot)$ is a robust weighting function based on the t-distribution, as shown in Lange et al. (1989).

$$\rho(r) = \frac{\nu + 1}{\nu + (\frac{r-\mu_r}{\sigma_r})^2} \tag{7}$$

Here, ν is the degree of freedom of the t-distribution that was set to 5 as in Kerl et al. (2013b) and Li and Lee (2016). μ_r and σ_r represent the mean and the standard deviation of all intensity residuals. To cope with the situation in which the photometric consistency of two frames is not maintained due to illumination change caused by auto-exposure, we use median value as μ_r and median absolute deviation as σ_r:

$$\begin{aligned} \mu_r &= \text{median}\left(\{r_i\}_{i=1}^{N_r}\right) \\ \sigma_r &= 1.4826\,\text{median}(\{|r_i - \mu_r|\}_{i=1}^{N_r}) \end{aligned} \tag{8}$$

where r_i is the residual value, i is its index and N_r is the number of residuals. This median value is effective in the situation where the mean is biased by outliers.

To solve this nonlinear objective function of the tracking process in Eq. (6), the Gauss–Newton method was utilized while recalculating the weights for each iteration. From the initial relative pose $\mathbf{T}_{ji}$, the solved increment $\boldsymbol{\xi} \in se(3)$ updates it incrementally through $\boxplus$ operator. For more efficient computation, we employ an inverse compositional approach. This solves ξ in the reference frame instead of solving it in the target frame, and updates $\mathbf{T}_{ji}$ in the reference frame. The residual is then rewritten in the following form:

$$r(\mathbf{p}_i) = \mathcal{I}_j(\pi(\mathbf{T}_{ji}\mathbf{p}_i)) - (a\mathcal{I}_i(\pi(\mathbf{T}(\xi)\mathbf{p}_i)) + b). \tag{9}$$

To find the optimal increment, we evaluate the derivative at $\xi = 0$. Since the inverse compositional approach linearizes at the same point during the iteration, we can precompute jacobian, thereby significantly improving the computation speed. The increment ξ is calculated in the target frame using Eq. (10):

$$\delta\xi = -(\mathbf{J}^T\mathbf{W}\mathbf{J})^{-1}\mathbf{J}^T\mathbf{W}\mathbf{r}, \tag{10}$$

where $\mathbf{J}$ is the Jacobian, $\mathbf{W}$ is the t-distribution weight matrix and $\mathbf{r}$ is stacked residuals. Once ξ has been obtained, the update step is inversely applied as shown below.

$$\mathbf{T}_{ji} \leftarrow \mathbf{T}_{ji}\exp(-\hat{\xi}) \tag{11}$$

In addition, we leverage a coarse-to-fine scheme and a constant velocity model to handle local minima and large-displacement situations.

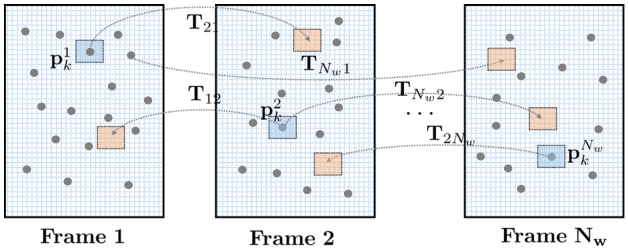

Fig. 4 Illustration of window-based optimization. The window consists of N_w^{th} keyframes, and each keyframe has its image and pointcloud. Gray dots represent the pointcloud and rectangles represent images. A point $\mathbf{p}_k$ in keyframe is projected into all keyframes with covisibility. A photometric residuals are then calculated between the patch at the existing point and the other patch at a projected point

4.4 Window-based optimization

This section details the pose refinement through window-based optimization. If the current frame satisfies the keyframe creation criteria, a sliding window-based refinement is performed on the multiple keyframes. As in the tracking module, the photometric error model is adopted. The only difference is that the error model is applied to multiple frames. The refinement in the multiple frames overcomes the lack of information, such as a situation where the measurement is biased in a single frame. However, this sliding window based approach requires more computation as the number of frames increases. To ensure constant computations, we kept a fixed number of 3 to 10 frames, depending on the sparsity of the measurement.

Each keyframe ($\mathcal{F}_i$) in the window ($\mathbf{W}_\mathcal{F}$) has its own associated points ($\mathbf{P_i}$) and pose ($\mathbf{T_i}$). As shown in Fig. 4, a point ($\mathbf{p}_k$) is projected into other keyframes depending on the covisibility in the window. This procedure creates a pose-graph that contains points as latent variables; we want to optimize the pose parameters $\Xi = \{\mathbf{T}_0, \ldots, \mathbf{T}_{N_W}\}$ associated with those keyframes. Because range measurements from LiDAR are fairly accurate, we only perform the optimization on the poses of keyframes. To solve this refinement problem, we define a photometric error in multiple frames and use the patch of sparse pattern Ω as in the previous section.

$$\underset{\Xi}{\text{argmin}}\,\frac{1}{2}\sum_{\mathcal{F}_w \in \mathbf{W}_\mathcal{F}}\sum_{\mathbf{p}_k \in \mathbf{P_i}}\sum_{i \in \Omega(\mathbf{p}_k)} w(r(\mathbf{p}_k))\left(r(\mathbf{p}_k)^2\right) \tag{12}$$

Differing from Eq. (6), the error of point measurement appears in multiple frames. The t-distribution weight $w(\cdot)$ is applied for robustness.

4.5 Loop-closure

Although exploiting sliding window technique allows odometry estimation with low drift, it is still an issue that errors

eventually accumulate deteriorating the consistency of the trajectory in a large-scale environment. To alleviate this issue, it is necessary to optimize the entire trajectory by incorporating loop-closing. To accomplish this task, we use a global pose-graph. Whenever a revisiting area is found through the place recognition module, the loop-closing module provides the constraint to the pose-graph and performs optimization.

4.5.1 Place recognition

The place recognition module provides a data association for the reobserved area to obtain a consistent map in a large-scale environment. In our framework, the place recognition module depends on the DBoW proposed by Gálvez-López and Tardós (2012). Whenever a new keyframe is provided by the VO module, the frame is represented as bag-of-words vector $\mathbf{v}$. Then the keyframe database computes the following normalized similarity score to find a matching candidate for the query vector $\mathbf{v_q}$:

$$\eta(\mathbf{v}_c, \mathbf{v}_q) = \frac{s(\mathbf{v}_c, \mathbf{v}_q)}{s(\mathbf{v}_c, \mathbf{v}_1)} \tag{13}$$

where $\mathbf{v}_q$ is a bag-of-binary words vector of the previously added keyframe. To calculate the similarity $s(\mathbf{v}_c, \mathbf{v}_q)$, the L_1-score is used as shown below:

$$s(\mathbf{v}_c, \mathbf{v}_q) = 1 - \frac{1}{2}\left|\frac{\mathbf{v}_c}{|\mathbf{v}_c|} - \frac{\mathbf{v}_q}{|\mathbf{v}_q|}\right| \tag{14}$$

In this module, we exploit the pre-trained vocabulary from Mur-Artal et al. (2015). If the score in Eq. (13) is less than threshold $\alpha = 0.7$, we reject the candidate. Otherwise, we consider the keyframe as a potential matching candidate for the loop-closing. Finally, we validate the temporal consistency by checking the similarity score between the candidate and its neighboring keyframes of the current keyframe.

4.5.2 Loop-closing

The place recognition module provides a matching candidate for the constraints of the pose-graph. Rather than performing loop-closing whenever a candidate is found, we perform cross-validation by comparing the relative pose bidirectionally. Given a new query keyframe $\mathcal{F}_q$ and a candidate keyframe $\mathcal{F}_c$, we can estimate the relative pose from Eq. (6). By changing the role of the reference frame in that equation, we can independently compute two relative poses $\mathbf{T}_{qc}$ and $\mathbf{T}_{cq}$. If the difference between these two relative poses is subtle, this cross-validation leads to a strong agreement and the relative pose is added to the pose-graph as a loop constraint.

$$\| \log\left(\mathbf{T}_{qc}\mathbf{T}_{cq}\right)\| < \epsilon \tag{15}$$

Table 1 Parameters for keyframe management

	Description	KITTI	Indoor	Outdoor
γ_{kf}	Ratio for keyframe creation	0.8	0.8	0.8
t_{kf}	Time for keyframe creation (s)	1.0	1.0	1.0
N_w	Window size (number of frames)	5	7	5

This procedure replaces performing geometric verification with random sample consensus (RANSAC) in the feature-based indirect method. Finally, Dynamic Covariance Scaling (DCS) is applied to the loop edge to prevent failures due to unreliable constraints (Agarwal et al. 2013).

5 Experimental evaluations

The proposed method was validated in various environments where the camera and the LiDAR can be used simultaneously. The proposed method was evaluated using both the publicly available open dataset and our own portable camera-LiDAR sensor system. A consumer-level laptop with an Intel Core i7-7700HQ and 8 GB RAM was used for all experiments. The parameters applied for the front-end of the algorithm are introduced in Table 1.

5.1 Visual odometry evaluation on KITTI

To verify visual odometry performance, the proposed algorithm was applied on the KITTI odometry benchmark dataset (Geiger et al. 2012) by excluding loop-closure. In the KITTI dataset, we only used the monochrome image acquired in the left camera and the pointcloud from LiDAR for validation of the proposed method. The dataset provides a 1230 × 370 resolution image, and LiDAR has 64 laser beams with an FOV that is 30° in vertical and 360° in horizontal. The first

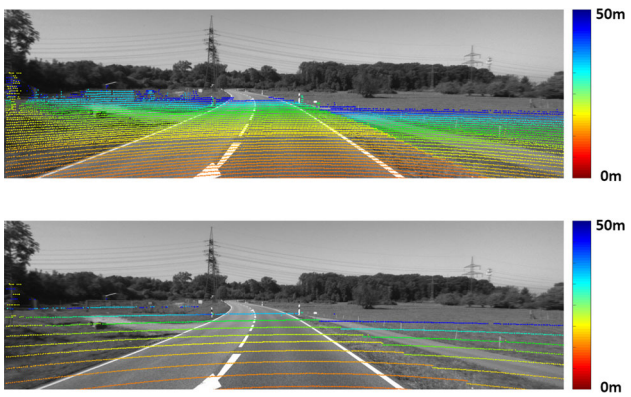

Fig. 5 From a rich LiDAR data (64 laser beam from HDL64-E) in the original data, we simulate a sparse LiDAR by sampling laser. The points are color-coded by their depth value

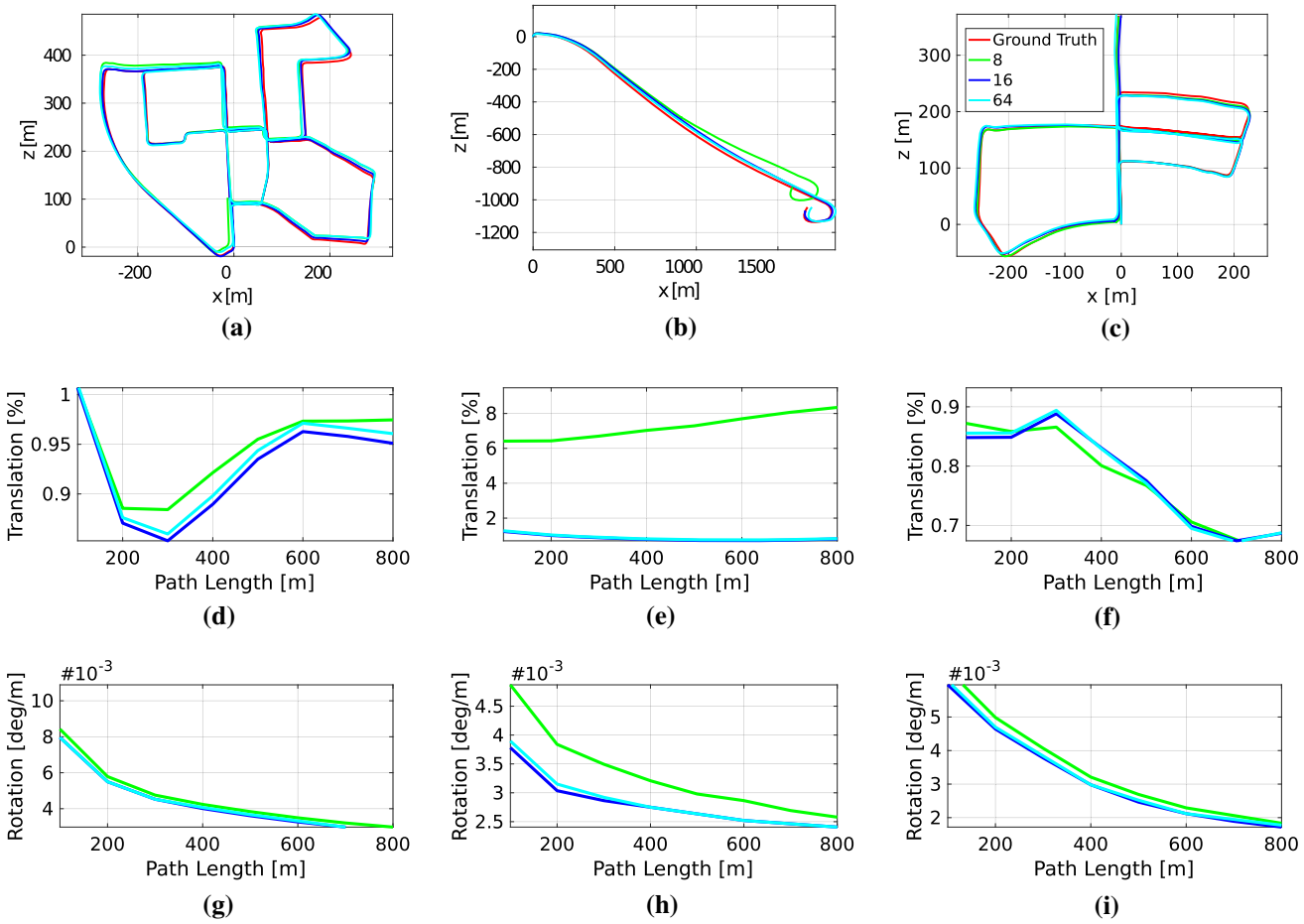

Fig. 6 The sample result of the propose method in the KITTI dataset. From the left column, the result represents sequence *00*, *01* and *05*. The results represent a comparison 64 ray LiDAR as well as with simulated 16 and 8 ray LiDAR. Top row shows the estimated trajectory in **a**–**c**. The middle shows the translational error according to the path length in **d**–**f**, and **g**–**i** of the bottom shows the rotational error

set of experiments focused on evaluating the accuracy of the proposed visual odometry and verifying the robustness according to the sparsity of the laser beam. For this purpose, 64 laser beams were equally sampled to 16 and 8 rays, as shown in Fig. 5. The evaluation contains no loop-closure in order to compare only the motion estimation performance of the visual odometry.

The KITTI odometry benchmark provides 11 training sets with ground truth. Figure 6 shows the sample result from several sequences of the training set using the proposed method. The top row in Fig. 6 represents the estimated trajectory, and the middle and the bottom show translational and rotational error with respect to travel distance. The result contains the results from sequence *00*, *01* and *05*. Note that sequence *01* is a highway dataset obtained from a vehicle moving at high speeds in a dynamic environment. Compared to other results, the 8 rays have the most deteriorated estimated trajectory in sequence *01*. Because sequence *01* is a highway dataset, repeated patterns in the highly dynamic environment resulted in a larger error when applying 8 rays. However, the proposed method successfully handled the sparse range measurement in other test cases.

The quantitative comparison results for all training sets are shown in Table 2. The table includes the results according to the sparsity of the laser beam. The results of DEMO are directly referred from Zhang et al. (2017). The translational error is the average of the errors measured by dividing the entire trajectory into segments of 100 m, 200 m, $\cdots$, 800 m. The proposed method shows an average translational error of 0.98% at 64 laser beams presenting better performance than DEMO. The result of 16 laser beams shows 0.98% similar to the results of 64 laser beams. In the case of 8 laser beams, estimation yielded a meaningful accuracy except for the highway case (sequence *01*).

We present the error analysis according to path length and vehicle speed, as shown in Fig. 7. In addition, we included the results from three laser beams of 64, 16 and 8 rays to examine the sparsity of the LiDAR. Figure 7 presents the translational error on the top and the rotational error on the bottom. Despite the length of the accumulated path increases, the translational

Table 2 Comparison result of KITTI dataset

Seq no.	Path len. (m)	Environment	DEMO (64 ray)	Ours (64 ray)	Ours (16 ray)	Ours (8 ray)
0	3714	Urban	1.05	0.93	**0.93**	0.95
1	4268	Highway	1.87	1.47	**1.09**	7.25
2	5075	Urban + country	**0.93**	1.11	1.26	0.95
3	563	Country	0.99	**0.92**	0.98	0.94
4	397	Country	1.23	**0.67**	1.08	1.08
5	2223	Urban	1.04	0.82	**0.77**	0.77
6	1239	Urban	0.96	0.92	**0.75**	0.75
7	695	Urban	**1.16**	1.26	1.30	1.42
8	3225	Urban + country	**1.24**	1.32	1.30	1.30
9	1717	Urban + country	1.17	**0.66**	0.72	0.72
10	919	Urban + country	1.14	0.70	**0.57**	0.57
avg			1.16	0.98	**0.98**	1.52

The result represent an averaging translational error at each segment length
The bold type indicates the best accuracy

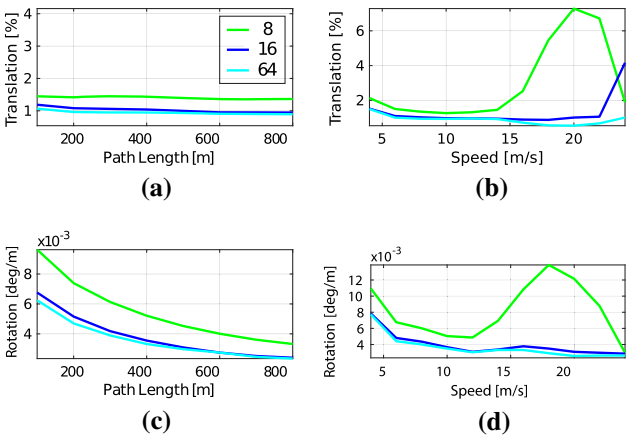

Fig. 7 Error comparison with respect to the sparsity of LiDAR in the whole training sequences. The first row shows the translational error and the second row shows the rotational error. The left column and the right column show the errors according to path length and vehicle speed

Fig. 8 Sensor configuration for outdoor experiment. **a** A camera-LiDAR sensor on vehicle and **b** a VRS-GPS that is only used for validation

error is steadily bounded as in Fig. 7a, and the rotational error is reduced in Fig. 7b. The translational error is relatively small for the entire route because most of the motion in the vehicle dataset is locally facing forward. The right column shows the error according to the speed of the vehicle. As in Fig. 7d, the faster the vehicle drives the smaller the rotational error becomes. This phenomenon seems to occur due to the nature of the non-holonomic constrained platform. Nevertheless, if the measurement is too sparse, such as 8 ray LiDAR, the error tends to increase according to the vehicle speed in Fig. 7b, d.

5.2 Evaluation on custom-built sensor system

The proposed method was evaluated by using our custom-built camera-LiDAR sensor system in Fig. 1a. The sensor system consists of a camera and a 3D LiDAR. The camera provides an RGB image with a resolution of 800 × 600 running at the frame rate of 40 Hz with horizontal FOV 90°. The 3D LiDAR used in the sensor system is equipped with VLP-16 from Velodyne. The VLP-16 provides a relatively sparse pointcloud over 16 laser beams, which results in very small overlapping areas considering the FOV of the camera and the LiDAR. Note that we performed experiments using only the LiDAR measurements in the overlapping area.

In order for the proposed method to perform with optimal performance in the sensor system, precise calibration must be guaranteed between the heterogeneous sensors. We first performed a calibration to obtain intrinsic parameters of the camera to project the 3D point. The extrinsic calibration was then followed to obtain the relative transformation between the camera and the LiDAR in a conventional manner from Kassir and Peynot (2010). The proposed method was validated by the sensor system in the outdoor and the indoor environment, including loop-closure. In the outdoor experiments, the sensor system was mounted on a car-like

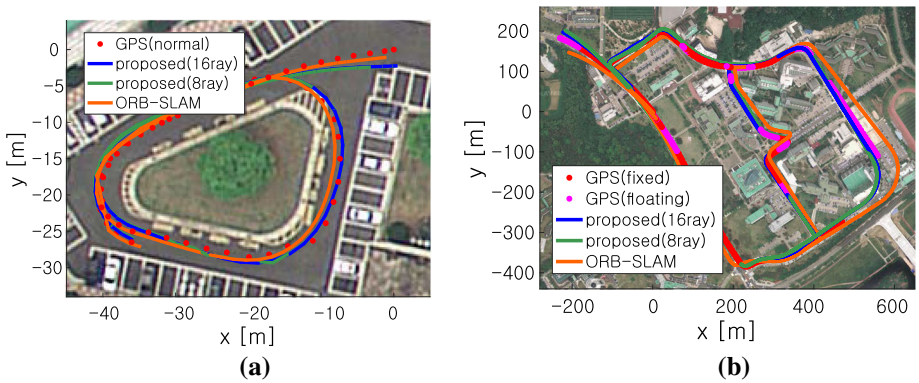

Fig. 9 Comparison of estimated trajectories from the proposed method, GPS and ORB-SLAM. **a** A parking lot experiment and **b** a large-scale campus experiment. Red and magenta represent the GPS position, and blue is the proposed method using original 16-ray LiDAR. Green is proposed method from sampled 8-ray LiDAR. Orange represents the results of ORB-SLAM (Color figure online)

vehicle moving in campus areas. On the other hand, the indoor experiment was conducted in the corridor inside the campus building, whereas the sensor system was moved by means of a handheld. Similar to KITTI evaluation, the parameters for the visual front-end are listed in Table 1. For each experiment, we compared the trajectory using two methods. One is a proposed method, and the other is a single camera-based ORB-SLAM. Since the ORB-SLAM used a monocular camera, we performed the scale compensation using the alignment errors of the ground truth and the estimated position, then compared them (Zhang and Scaramuzza 2018).

$$\underset{s,\mathbf{R},\mathbf{t}}{\operatorname{argmin}} \sum \parallel \mathbf{p}_{gt} - s\mathbf{R}\hat{\mathbf{p}} - \mathbf{t} \parallel \tag{16}$$

where s, $\mathbf{R}$ and $\mathbf{t}$ are similarity transformation parameters, $\mathbf{p}_{gt}$ is ground truth position and $\hat{\mathbf{p}}$ is estimated position. As the scale is known in the proposed method, it is fixed to $s = 1$. However, we could not use the ground truth trajectory in the indoor experiment; we only reported the result of the proposed method.

5.2.1 Outdoor experiment

The outdoor experiments are performed using our own camera-LiDAR sensor system mounted on a car-like vehicle. As shown in Fig. 8, the vehicle was equipped with our portable sensor system together with an additional virtual reference station (VRS)-global positioning system (GPS). The VRS-GPS provides an average error of 20 mm in the fixed state and up to 1 m in the floating state. In addition, only a few tens of meters of accuracy is guaranteed in the normal state. By assuming the VRS-GPS measurements as ground truth, we computed the position error of the proposed method. To fairly compare the performance of the proposed method against other single sensor-based methods, experiments were conducted on a parking lot surrounded by buildings and campus as the large-scale environment.

Figure 9 shows the satellite image and the vehicle route from GPS. In the parking lot experiment, the total travel distance is approximately 0.15 km and contains one loop. The second experiment was conducted throughout the campus that is a large-scale environment with a total travel distance of 3.97 km and incorporating two loops. We compared two methods against the baseline positions measured by VRS-GPS.

Table 3 Absolute position error between the VRS-GPS and the estimated pose

	16-Ray (m)	8-Ray (m)	ORB-SLAM (m)
Parking lot	1.09	1.17	1.79
Campus	8.77	11.58	17.42

The proposed method shows better performance than ORB-SLAM in 8-ray and 16-ray results

In the parking lot experiment, we compared the trajectories with the normal state GPS as ground truth. Because the surrounding buildings are degrading the GPS signal, we could only detect signals in the normal state. Figure 9a shows trajectory estimation and its comparison to the ground truth in the parking lot. Figure 9b represent the comparison result in the campus experiment. The proposed method is consistent and accurate in spite of the large-scale environment.

Table 3 shows the absolute position error between the position estimated by the two methods and the VRS-GPS. To simulate the situation of the sparser LiDAR measurement, we further compared the 16 laser beams of the VLP-16 sampled into 8 beams. The campus experiment, involved a discontinuous interval of the GPS measurement, and we calculated the error using only available GPS measurements. The proposed method shows an absolute position error of 1.09 m and 8.77 m in the two experiments. In particular, large-scale campus experiments showed about half of the ORB-SLAM position errors. Also, when the method was run with 8 laser beams, the errors were 1.17 m and 17.42 m.

Our method includes loop-closing and allows consistent trajectory results on a large-scale. In addition, the method has pointcloud measurements in all frames, so as to produce a colored pointcloud directly using the estimated trajectory. Figure 10a reveals the pointcloud map obtained from the outdoor

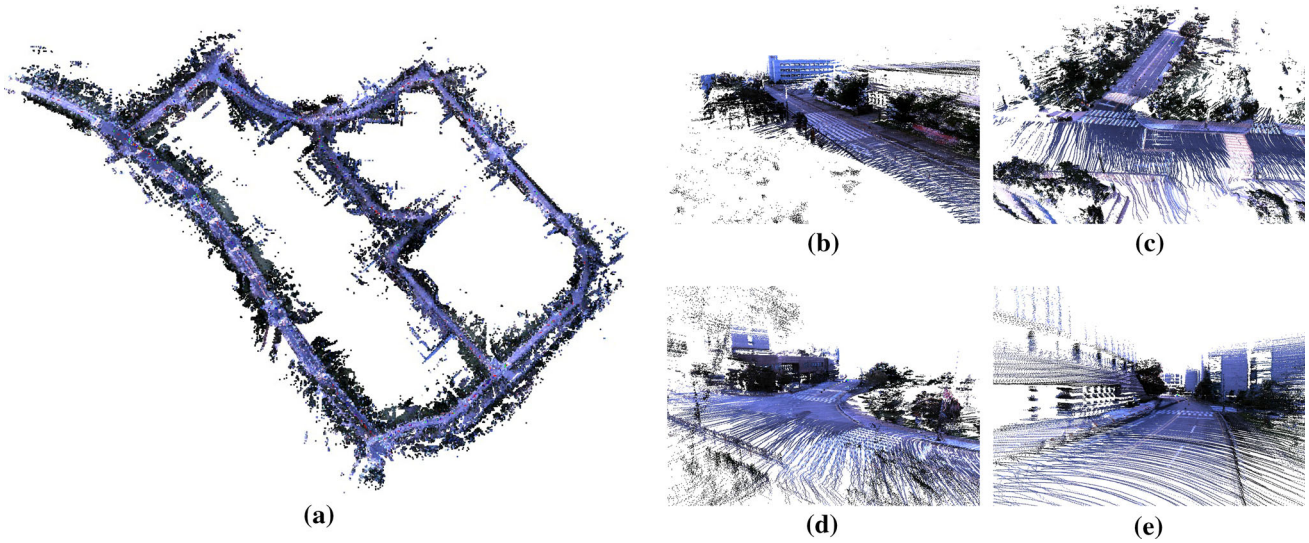

Fig. 10 3D reconstruction result on the campus experiment. **a** The pointcloud obtained by performing loop-closing in the entire area of the experiment. **b–e** Depict the close-up of the generated pointcloud

experiment. The resulting pointcloud map presents enough accuracy, even though no additional geometric point matching was performed. Because of the results obtained only with visual odometry and loop-closing results, further improvement may be possible if point-matching or map merging algorithms are performed as in Zhang and Singh (2015).

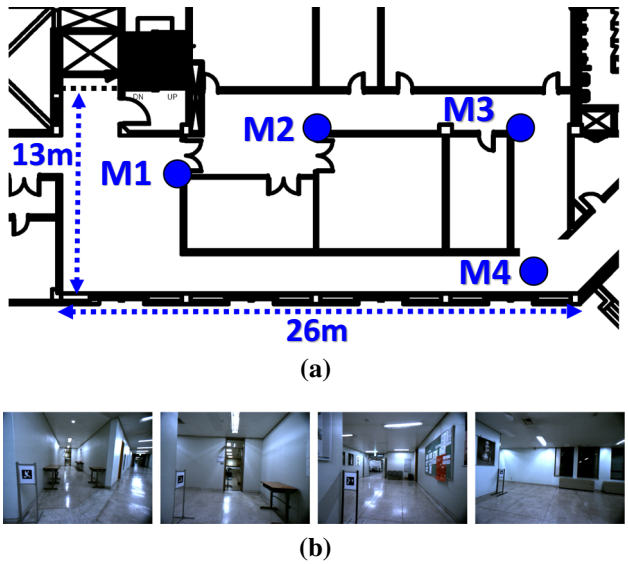

Fig. 11 Indoor experimental environment. **a** The floor plan with the location where the ArUCo marker is installed. The location where the marker was installed is shown as blue dots. **b** The corridor environment with the marker in which the experiment was performed (Color figure online)

5.2.2 Indoor experiment

The next experiment was performed in an indoor environment using our camera-LiDAR sensor system. The indoor environment is a typical GPS-denied environment and ground-truth is not provided. In order to evaluate the performance of the proposed method in this experiment, we installed ArUCo markers from Garrido-Jurado et al. (2014) at four locations as shown in Fig. 11a. Figure 11b shows the scene in which the markers are installed in the corridor. We evaluate the performance of the proposed method by comparing the relative poses measured from the markers against their corresponding estimated poses. We traveled along the loop three times, resulting in a total path length of 180 m. Since the absolute position of the markers in the indoor experiment is not known accurately, we compared the Root Mean Squared Error (RMSE) of the relative position between the measured position by the markers and the estimated position by proposed method whenever the markers were re-observed on the revisit. Single sensor-based methods were only qualitatively compared because of the failure of the trajectory estimation.

As shown in Fig. 11a, the markers were placed at the four corners in the corridor (M1 to M4). They are recog-

Table 4 Relative position error between the measured relative pose by the marker and the relative pose in the estimated trajectory

	M1 (m)	M2 (m)	M3 (m)	M4 (m)
Loop 1–2	0.1263	0.0712	0.1074	0.0535
Loop 2–3	0.0655	0.0633	0.2540	0.0668
Loop 3–1	0.1323	0.0770	0.1837	0.0636

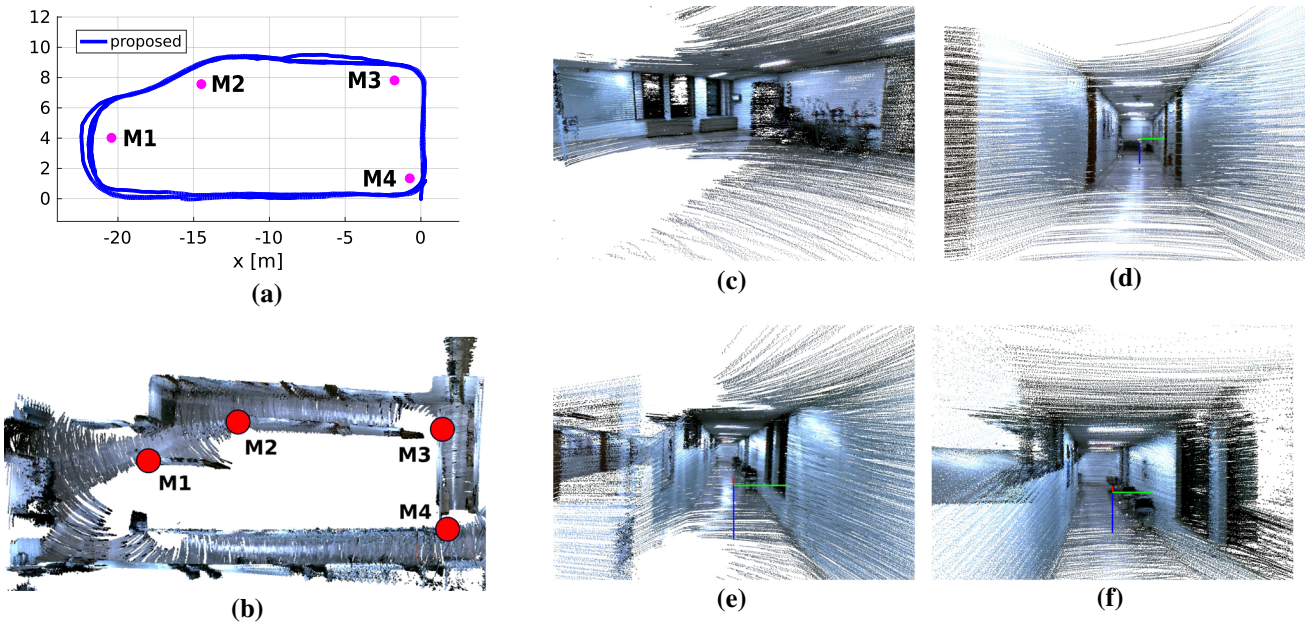

Fig. 12 Comparison of estimated trajectories and 3D reconstruction result on the indoor experiment. Magenta points depict marker position. **b** The pointcloud obtained by performing loop closing in the entire area of the experiment. **c**–**f** The close-up of the generated pointcloud (Color figure online)

nized by the camera to measure the relative poses between the marker and the camera. Table 4 summarizes the relative positional error between the measured pose by the ArUCo marker and the estimated pose by the proposed method. When the marker is viewed too far away from the camera, the degradation of accuracy is reported (Malbezin et al. 2002). Hence, we compared the results using the measured pose when the marker was recognized only at a close distance. While the planned path provides three revisits, the average relative error was kept small (0.105 m). This is because the proposed method includes loop-closing via pose graph optimization, while successfully maintaining the consistency of the map and reducing the drift of the entire trajectory. In one case of M3, the reported error is somewhat larger between the second and third revisit. This is due to the placement of the M3 marker since the marker is located right around the corner and is suddenly recognized immediately after the corner during the experiment. This leads to the error in marker-based pose measurement.

Figure 12 represents the comparison of estimated trajectories from each method and 3D reconstruction results. Figure 12a depicts the trajectory result with the estimated position of the markers. Whenever a marker is observed in the image, the position of the marker is measured based on the estimated pose. Three times of revisits produced a number of positional estimation of each marker from multiple scenes, and we plotted the averaged measurement of each marker. The inferred trajectory from the proposed method is consistent due to the loop constraint during three laps. Figure 12b shows the 3D reconstruction results. The reconstructed 3D pointcloud map may be sparse compared to dense RGB-D camera cases but may present excellent quality in terms of global consistency as in Fig. 12c–f. Even though no further refinement through point matching was followed, the proposed method yielded a consistent pointcloud map of the environment.

6 Conclusion

In this work, we introduced a direct visual SLAM method for camera-LiDAR sensor systems where sparse depth is available. The proposed method presented robust SLAM results, even if extremely sparse depth measurements (8-ray) were available in a large-scale environment. Sliding window-based pose estimation prevents local level drift, and the appearance-based place recognition module and the pose graph optimizer maintain consistency at the global level. The method was evaluated using our own sensor system and KITTI benchmark dataset. Our method provided accurate trajectory and pointcloud maps of real-world metrics without performing point matching at all.

Despite the sensor-wise advantage of direct exploitation of a LiDAR, the displacement between the two sensors may cause occluded points between multiple frames. As also indicated by Huang and Stachniss (2019), degradation of robustness and accuracy under large occlusion can be further improved by explicitly handling occlusion. In the proposed approaches, the point sampling strategy

and outlier rejection scheme ameliorated this occlusion effect. Targetting a more cluttered environment, our future work is at ensuring accuracy and robustness under various moving objects and occlusion by incorporating semantic information.

In future work, we plan to extend our approach to robust camera-LiDAR SLAM systems for illumination changes using geometric information from LiDAR. Accuracy can be further improved by including the extrinsic parameters of the two sensors in the optimizer.

Acknowledgements This research was supported by MOLIT (19TSRD-B151228-01) and by MOTIE (No. 10067202). Y. Shin is financially supported via 'Innovative Talent Education Program for Smart City' by MOLIT.

References

Agarwal, P., Tipaldi, G. D., Spinello, L., Stachniss, C., & Burgard, W. (2013). Robust map optimization using dynamic covariance scaling. In *IEEE international conference on robotics and automation (ICRA)* (pp. 62–69).

Bosse, M., Zlot, R., & Flick, P. (2012). Zebedee: Design of a spring-mounted 3-d range sensor with application to mobile mapping. *IEEE Transactions on Robotics*, *28*(5), 1104–1119.

Civera, J., Davison, A. J., & Montiel, J. M. M. (2008). Inverse depth parametrization for monocular slam. *IEEE Transactions on Robotics*, *24*(5), 932–945.

Concha, A., & Civera, J. (2017). Rgbdtam: A cost-effective and accurate rgb-d tracking and mapping system. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 6756–6763).

Davison, A. J., Reid, I. D., Molton, N. D., & Stasse, O. (2007). Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(6), 1052–1067.

Endres, F., Hess, J., Sturm, J., Cremers, D., & Burgard, W. (2014). 3-d mapping with an RGB-D camera. *IEEE Transactions on Robotics*, *30*(1), 177–187.

Engel, J., Koltun, V., & Cremers, D. (2017). Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*(99), 611–625.

Engel, J., Schöps, T., & Cremers, D. (2014). LSD-SLAM: Large-scale direct monocular SLAM. In *European conference on computer vision (ECCV)*.

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

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

Gálvez-López, D., & Tardós, J. D. (2012). Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, *28*(5), 1188–1197.

Garrido-Jurado, S., noz Salinas, R. M., Madrid-Cuevas, F., & Marín-Jiménez, M. (2014). Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, *47*(6), 2280–2292.

Gauglitz, S., Sweeney, C., Ventura, J., Turk, M., & Höllerer, T. (2012). Live tracking and mapping from both general and rotation-only camera motion. In *Proceedings of 11th IEEE international symposium on mixed and augmented reality (ISMAR)*, Atlanta, Georgia, (pp. 13–22).

Geiger, A., Lenz, P., & Urtasun, R. (2012). Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Conference on computer vision and pattern recognition (CVPR)*.

Gräter, J., Wilczynski, A., & Lauer, M. (2018). LIMO: Lidar-monocular visual odometry. In *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 7872–7879).

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. *The International Journal of Robotics Research*, *31*(5), 647–663.

Huang, A. S., Bachrach, A., Henry, P., Krainin, M., Maturana, D., Fox, D., et al. (2011). Visual odometry and mapping for autonomous flight using an RGB-D camera. In *International symposium on robotics research (ISRR)* (pp. 1–16).

Huang, K., & Stachniss, C. (2019). Accurate direct visual-laser odometry with explicit occlusion handling and plane detection. In *Proceedings of the IEEE international conference on robotics and automation (ICRA), accepted* (To appear).

Jin, H., Favaro, P., & Soatto, S. (2001). Real-time feature tracking and outlier rejection with changes in illumination. In *Proceedings 8th IEEE international conference on computer vision. ICCV 2001* (Vol. 1, pp. 684–689).

Kassir, A., & Peynot, T. (2010). Reliable automatic camera-laser calibration. In G. Wyeth & B. Upcroft (Eds.), *Australasian conference on robotics and automation (ACRA), ARAA*. Queensland: Brisbane.

Kerl, C., Sturm, J., & Cremers, D. (2013a). Dense visual slam for RGB-D cameras. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 2100–2106).

Kerl, C., Sturm, J., & Cremers, D. (2013b). Robust odometry estimation for RGB-D cameras. In *IEEE international conference on robotics and automation* (pp. 3748–3754).

Kitt, B., Geiger, A., & Lategahn, H. (2010). Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme. In *IEEE intelligent vehicles symposium (IV)* (pp. 486–492).

Klein, G., & Murray, D. (2007). Parallel tracking and mapping for small AR workspaces. In *2007 6th IEEE and ACM international symposium on mixed and augmented reality* (pp. 225–234).

Kuemmerle, R., Grisetti, G., Strasdat, H., Konolige, K., & Burgard, W. (2011). G2o: A general framework for graph optimization. In *Proceedings of the IEEE international conference on robotics and automation, Shanghai, China* (pp. 3607–3613).

Kuznietsov, Y., Stuckler, J., & Leibe, B. (2017). Semi-supervised deep learning for monocular depth map prediction. In *The IEEE conference on computer vision and pattern recognition (CVPR)*.

Lange, K. L., Little, R. J., & Taylor, J. M. (1989). Robust statistical modeling using the t distribution. *Journal of the American Statistical Association*, *84*(408), 881–896.

Li, S., & Lee, D. (2016). Fast visual odometry using intensity-assisted iterative closest point. *IEEE Robotics and Automation Letters*, *1*(2), 992–999.

Lu, D. (2016). *Vision-enhanced lidar odometry and mapping*. Master's thesis, Carnegie Mellon University, Pittsburgh, PA.

Malbezin, P., Piekarski, W., & Thomas, B. H. (2002). Measuring ARTootKit accuracy in long distance tracking experiments. In *The 1st IEEE international workshop agumented reality toolkit* (p. 2).

Mur-Artal, R., Montiel, J. M. M., & Tardós, J. D. (2015). ORB-SLAM: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, *31*(5), 1147–1163.

Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., et al. (2011a). Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality* (pp. 127–136).

Newcombe, R. A., Lovegrove, S. J., & Davison, A. J. (2011b). DTAM: Dense tracking and mapping in real-time. In *2011 international conference on computer vision* (pp. 2320–2327).
Park, C., Moghadam, P., Kim, S., Elfes, A., Fookes, C., & Sridharan, S. (2018). Elastic lidar fusion: Dense map-centric continuous-time slam. In *Proceedings of the IEEE international conference on robotics and automation, Brisbane* (pp. 1206–1213).
Pinggera, P., Pfeiffer, D., Franke, U., & Mester, R. (2014). Know your limits: Accuracy of long range stereoscopic object measurements in practice. In *European conference on computer vision* (pp. 96–111).
Pizzoli, M, Forster, C., & Scaramuzza, D. (2014). Remode: Probabilistic, monocular dense reconstruction in real time. In *2014 IEEE international conference on robotics and automation (ICRA)* (pp. 2609–2616).
Qin, T., Li, P., & Shen, S. (2018). Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, *34*(4), 1004–1020.
Segal, A., Haehnel, D., & Thrun, S. (2009). Generalized-ICP. In *Proceedings of robotics: Science and systems, Seattle, USA*.
Serafin, J., & Grisetti, G. (2015). NICP: Dense normal based point cloud registration. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 742–749).
Shin, Y. S., Park, Y. S., & Kim, A. (2018). Direct visual slam using sparse depth for camera-lidar system. In *Proceedings of the IEEE international conference on robotics and automation, Brisbane* (pp. 5144–5151).
Sibley, G., Matthies, L., & Sukhatme, G. (2007). Bias reduction and filter convergence for long range stereo. In S. Thrun, R. Brooks, H. Durrant-Whyte (Eds.), *Robotics research* (pp. 285–294). Berlin: Springer.
Steinbrücker, F., Sturm, J., & Cremers, D. (2011). Real-time visual odometry from dense RGB-D images. In *2011 IEEE international conference on computer vision workshops (ICCV Workshops)* (pp. 719–722).
Strasdat, H., Davison, A. J., Montiel, J. M. M., & Konolige, K. (2011). Double window optimisation for constant time visual slam. In *2011 international conference on computer vision* (pp. 2352–2359).
Tateno, K., Tombari, F., Laina, I., & Navab, N. (2017). Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *The IEEE conference on computer vision and pattern recognition (CVPR)*.
Whelan, T., Kaess, M., Johannsson, H., Fallon, M., Leonard, J. J., & McDonald, J. (2015). Real-time large-scale dense RGB-D slam with volumetric fusion. *The International Journal of Robotics Research*, *34*(4–5), 598–626.
Yang, N., Wang, R., Stuckler, J., & Cremers, D. (2018). Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry. In *The European conference on computer vision (ECCV)*.
Yang, Z., & Shen, S. (2017). Monocular visual-inertial state estimation with online initialization and camera-IMU extrinsic calibration. *IEEE Transactions on Automation Science and Engineering*, *14*(1), 39–51.
Yin, X., Wang, X., Du, X., & Chen, Q. (2017). Scale recovery for monocular visual odometry using depth estimated with deep convolutional neural fields. In *The IEEE international conference on computer vision (ICCV)*.
Zhang, J., Kaess, M., & Singh, S. (2017). A real-time method for depth enhanced visual odometry. *Autonomous Robots*, *41*(1), 31–43.
Zhang, J., & Singh, S. (2015). Visual-lidar odometry and mapping: Low-drift, robust, and fast. In *IEEE international conference on robotics and automation (ICRA)* (pp. 2174–2181).
Zhang, J., & Singh, S. (2017). Low-drift and real-time lidar odometry and mapping. *Autonomous Robots*, *41*(2), 401–416.
Zhang, Z., & Scaramuzza, D. (2018). A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems* (pp. 7244–7251).

Young-Sik Shin received B.S. degree in Electrical Engineering from Inha University, Incheon, Korean, in 2013. He received a M.S. degree in the Robotics Program at Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea. Currently, he is a Ph.D. student in the Department of Civil and Environmental Engineering at KAIST. His research interest is focused on robot navigation, visual localization and SLAM.

Yeong Sang Park is the Ph.D. student in the Department of Civil and Environmental Engineering, KAIST. He received the B.S. in Electronic Engineering and M.S. degree in Electrical Engineering from Inha University, Incheon, Korea, in 2015 and 2017, respectively. His research interests include visual simultaneous localization and mapping (SLAM), navigation.

Ayoung Kim is the assistant professor in the Department of Civil and Environmental Engineering with joint affiliation at KI robotics, KAIST. She received the B.S. and M.S. degrees in Mechanical Engineering from Seoul National University, Seoul, Korea, in 2005 and 2007, respectively, and the M.S. degree in Electrical Engineering and the Ph.D. degree in Mechanical Engineering from the University of Michigan (UM), Ann Arbor, in 2011 and 2012, respectively. She also worked as a postdoctoral researcher in naval Architecture and Marine Engineering, UM in 2013 before works at ETRI as a senior researcher. Her research interests include visual simultaneous localization and mapping (SLAM), navigation, especially targeting long-term and large scale robotic autonomy.