# Ground and Aerial Collaborative Mapping in Urban Environments

Jinhao He, Yuming Zhou, Lixiang Huang, Yang Kong, and Hui Cheng, *Associate Member, IEEE*

***Abstract*—A heterogeneous multi-robot system consisting of Unmanned Ground Vehicles (UGVs) and Unmanned Aerial Vehicles (UAVs) have advantages over a single-robot system in efficiency and flexibility, enabling them to perform a larger range of tasks. To allow heterogeneous platforms to work together in GPS-denied scenarios, it is crucial to build a complete 3D map of the environment. In this letter, a novel method is presented to perform ground and aerial collaborative mapping leveraging visual and range data collected by cameras and 3D LiDAR sensors. In the proposed system, a visual-LiDAR ego-motion estimation module that considers point, line and planar constraints can provide robust odometry information. Thumbnail images representing obstacle outlines are generated and descriptors are extracted using a neural network to help perform data association between separate runs. Map segments and the robot poses are organized together and are updated during a pose graph optimization procedure. The proposed ground-aerial collaborative mapping approach is evaluated on both synthetic and real-world datasets comparing with other methods. Experiment results demonstrate that our method can achieve outstanding mapping results.**

***Index Terms*—Mapping, multi-robot systems, sensor fusion.**

## I. Introduction

HETEROGENEOUS robot systems consisting of both aerial and ground robots have many potential advantages over single-robot systems by improving efficiency, reliability and flexibility to achieve complex tasks such as exploration and rescue in large-scale environments. When unmanned systems try to perform specific tasks or to navigate in the environment successfully, the information about the surroundings is necessary. The technique of Simultaneous Localization and Mapping (SLAM) can be used to provide environmental information. In the SLAM problem, ego-motion of the robot is estimated and the information of the environment is recovered. However, when it comes to heterogeneous robot team scenarios, the problem is more challenging. To cooperate with each other, a robot in a multi-robot system needs to know not only its own states but also the perception information like localization results and obstacle detection results from other robots, represented under the same reference system. This requirement would be difficult to satisfy when GPS is disabled. Fortunately, if we are able to build a 3D map , robot poses and obstacles positions can thus be represented under a common map frame and utilized by every member of the robot team. One solution to this problem is to deploy multiple robots doing SLAM [1].

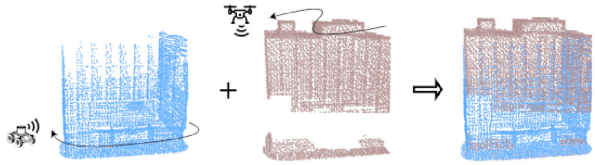


Fig. 1. An intuitive illustration of ground-aerial collaborative mapping problem. Ground-based map and aerial-based map are merged into a complete map.

Notice that the map built by homogeneous platforms has limited sensing range, a ground-based map can cover most of the area near the ground but the information at higher places remains unknown. Without knowledge about the flight environment, the ground-based map cannot be used directly by a drone. In this case, ground and aerial collaborative mapping systems are beneficial to help build a complete and consistent 3D map of the environment efficiently. The ground-aerial collaborative mapping problem is illustrated by Fig. 1

In recent years, many research efforts have been made to perform ground-aerial map reconstruction. Based on the sensors being used, existing methods fall into two main categories [2]: image-based methods and LiDAR-based methods.

Image-based methods treat the ground-aerial collaborative mapping task as an image feature matching problem, which has been widely used in structure-from-motion (SfM) or visual SLAM systems. In these methods [3]–[5], the features are extracted and matched between images to estimate the camera poses, and the multi-view stereo technique is used to recover the depth information of the image. The dense 3D map representation can be generated using surface reconstruction techniques [6], [7] with camera poses and depth point clouds as input. Considering that images captured by ground and aerial robots have large differences in viewpoint and scale, ground-aerial image matching might not be straight forward. Zhou *et al.* [8] propose a scale-invariant image matching approach to tackle the large scale variation of views. Feature matching is done by first determining the related scale levels in scale space and then considering image similarity measurement and feature correspondences simultaneously. Some other works focus on generating a synthetic image from the aerial viewpoint by directly warping the ground images based on their depth maps [9] or by projecting the ground model under the projective matrices of the aerial camera [10] and then perform feature matching between the synthetic images and the aerial images.

Manuscript received June 16, 2020; accepted October 1, 2020. Date of publication October 21, 2020; date of current version October 28, 2020. This letter was recommended for publication by Associate Editor S. Behnke and T. Stoyanov upon evaluation of the Reviewers' comments. *(Corresponding author: Hui Cheng.)*

The authors are with the School of Data and Computer Science, Sun Yat-Sen University, Guangzhou, Guangdong 510006, China (e-mail: hejh27@mail2.sysu.edu.cn; zhouym34@outlook.com; 675904985@qq.com; kongy6@mail2.sysu.edu.cn; chengh9@mail.sysu.edu.cn).

This letter has supplementary downloadable material available at https://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/LRA.2020.3032054

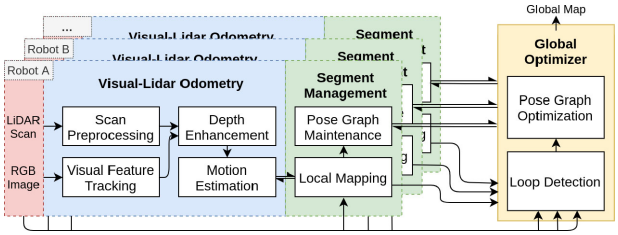
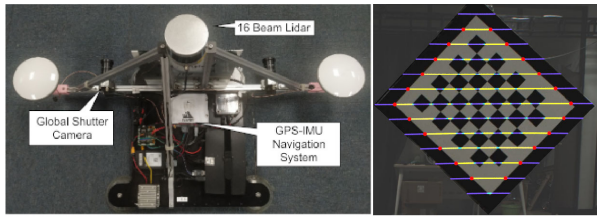



Fig. 2. The framework of the proposed ground-aerial collaborative mapping system. Each robot performs motion estimation using LiDAR scans and RGB images. Local mapping helps refine the estimated motion and construct a local map using raw sensor data. The mapping results from different robots are merged into a global map by the global optimizer.




(a) Sensor Configuration (b) Alignment Result

Fig. 3. The sensor configuration in the proposed system.

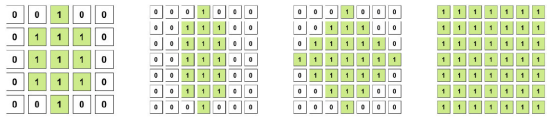



Fig. 4. Different kernels used for depth completion.

LiDAR-based methods operate on LiDAR point clouds, and the key idea is to align ground and aerial point clouds. Since the point clouds collected by LiDAR are more precise than those recovered by SfM systems, the global maps can be presented as merged point clouds directly. In the remote sensing field, some methods extract feature lines [11], [12] or corner points [13] of the building outlines from the ground and aerial laser clouds, and these feature correspondences are used to estimate the rotation and translation between two point clouds. Besides building outlines, features can also be extracted from raw point clouds. Zhang *et al.* [14] extend their previous work [15] of LiDAR SLAM into a collaborative mapping framework. Edge and planar points are extracted from each scan and later serve as correspondences. Surmann *et al.* [16] propose to make use of planar segments to perform registration. Gawel *et al.* [17] present a key-point descriptor matching algorithm to perform registration. There are also methods that rely on global positioning system(GPS) references or human assist like common point selection at preprocessing stage [18], [19].

In this letter, we propose a collaborative mapping system that integrates the ground-aerial point cloud registration algorithm and a pose graph-based collaborative SLAM pipeline together. It allows ground-and-aerial, multi-session collaborative mapping for heterogeneous robot team without the need of GPS or geotagged database. The proposed system contains the following features:

- A tightly-coupled visual-LiDAR odometry method is presented for localization. It uses a fast depth completion algorithm to help landmark point initialization, and considers point, line and planar constraints during the optimization.
- A loop detection method that is inspired by the idea of extracting and matching building outlines from the remote sensing field is proposed. Thumbnail images that represent obstacle outlines in the submaps are generated to help data association.

## II. Proposed Method

The architecture of the proposed method is illustrated in Fig. 2, where the rectangles indicate the key modules of the framework. In this section, the key components will be described in detail.

### A. Sensor Configuration

The sensor configuration used by our ground and aerial platforms is displayed in Fig. 3(a). The LiDAR and image data are time-synchronized and the extrinsics between the two sensors are calibrated in advance. The alignment result is given in Fig. 3(b).

### B. Tightly Coupled Visual-Lidar Odometry

To build a point cloud map of the environment, LiDAR scans are stitched together according to the corresponding LiDAR poses.

Without the help of GPS, sensor poses should be estimated, thus, the proposed ground-aerial collaborative mapping pipeline starts with a tightly coupled visual-LiDAR odometry module. This module receives inputs from a LiDAR sensor and a monocular camera, and provides odometry estimation that satisfies point, line and planar constraints. By combining a monocular camera and the LiDAR sensor for ego-motion estimation, the odometry module is robust under degenerate scenarios like repetitive textures or planar-dominated areas, which are very challenging for vision-only or LiDAR-only methods respectively.

For each incoming image $I_k$, the image features are detected and tracked with KLT sparse optical flow algorithm [20]. Let $F_k$ denote all the feature points extracted from the image frame captured at time $k$. The feature points are parameterized by inverse depth $\lambda$ and pixel coordinate $\mathbf{u} = [u, v]^T$. Let $f_k^i$ denote the $i$-th feature in $F_k$. For incoming LiDAR scans, let ${}^L P_k$ denote the point cloud acquired at time $k$ under LiDAR frame $L$. For simplicity, LiDAR points are transformed to the camera frame using the calibrated extrinsics $T_C^L$ in default, so the left upperscript would be ignored. In order to estimate the ego-motion with these input data, the next step is to find frame-to-frame constraints.

Notice that the depth values are unavailiable for the extracted feature points at the beginning, we propose to fetch depth values from the depth images generated using the aligned LiDAR scans so that no extra nearest search is needed. Recently, various methods [21]–[23] are presented to perform depth completion or single-image depth prediction. In this letter, a modified version of [21] is implemented to generate the depth map used for visual landmark initialization. It is fast and accurate and can run at nearly 100 Hz. The details are presented in Algorithm 1, in which kernels shown in Fig. 4 are used to handle the sparse point cloud data collected by a 16-beam LiDAR. Given LiDAR

**Algorithm 1:** Fast Depth Completion.

**Input:** Input Sparse Depth Image $D$
**Output:** Output $D^*$
1 $D_{inv} = 100.0 - D$;
2 Dilate $D_{inv}$ using $7 \times 7$ diamond-shape kernel // (dilate)
3 Dilate $D_{inv}$ using $5 \times 5$ oval kernel; // (connect scan lines)
4 Morphological close $D_{inv}$ with $7 \times 7$ oval kernel; // (close holes)
5 Dilate $D_{inv}$ using $7 \times 7$ full kernel and keep previous computed valid pixel unchanged // (fill medium holes)
6 Dilate $D_{inv}$ using $31 \times 31$ full kernel and keep previous computed valid pixel unchanged // (fill big holes)
7 $5 \times 5$ medium blur // (smooth result)
8 Keep only valid pixels in step 6;
9 $D^* = 100.0 - D_{inv}$

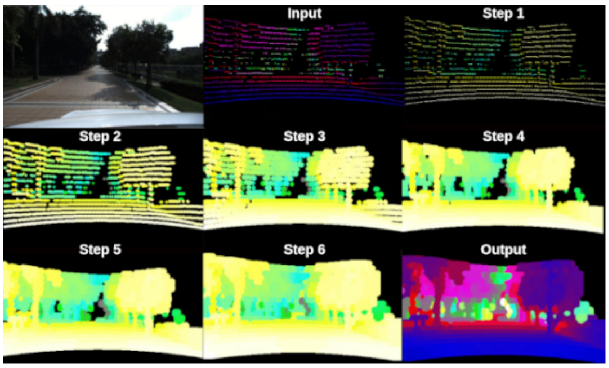



Fig. 5. Example outputs of the key steps in Algorithm 1. Best viewed in color.

scan $P_k$, by projecting them onto the image plane according to the camera intrinsics, we can get a sparse depth image $D_s$ whose pixels correspond to those from the monocular image $I$. A dense depth image $D_k$ is then generated using Algorithm 1. Example results of the key steps of Algorithm 1 is given by Fig. 5. For each $f_k^i$ in $F_k$, if the corresponding pixel on $D_k$ has a valid depth value, $f_k^i$ is initialized with the inverse depth, otherwise, it would keep being tracked and wait for a valid depth observation. The feature $f$ with valid depth can be back-projected to the 3D point $X = [x, y, z]^T$. Since every feature is generated from the precise LiDAR scan directly instead of being triangulated using the previously estimated motion, they can contribute to the motion estimation once being initialized.

During the preprocessing stage, the edge points $\mathcal{E}$ and planar points $\mathcal{P}$ are extracted according to the computed local curvature. These feature points are maintained and matched frame by frame following the similar strategy commonly used in LiDAR SLAM systems [15], [24]. Let $R_k^{k-1} \in SO(3)$ and $t_k^{k-1} \in \mathbb{R}^3$ be the rigid rotation and translation between frame $k-1$ and frame $k$, respectively. 3D points $X_k$ can be warped to $\bar{X}_k$ represented in the camera frame at time $k-1$:

$$\bar{X}_k = R_k^{k-1} X_k + t_k^{k-1} \tag{1}$$

The sets of warped edge points and planar points are denoted as $\bar{\mathcal{E}}$ and $\bar{\mathcal{P}}$, respectively. For each point $\bar{E}_k$ in $\bar{\mathcal{E}}_k$, the distance to the closest line defined by two nearest points in $\mathcal{E}_{k-1}$ is denoted as $d_{\mathcal{E}}(\bar{E}_k)$. For each point $\bar{P}_k$ in $\bar{\mathcal{P}}_k$, the distance to the closest plane defined by three nearest points in $\mathcal{P}_{k-1}$ is denoted as $d_{\mathcal{P}}(\bar{P}_k)$. Readers can refer to [15] and [24] for the definition of $d_{\mathcal{E}}(\bar{E}_k)$ and $d_{\mathcal{P}}(\bar{P}_k)$. The set of initialized feature points with depth values still available at time $k$ is denoted as $\bar{\mathcal{F}}_k$.

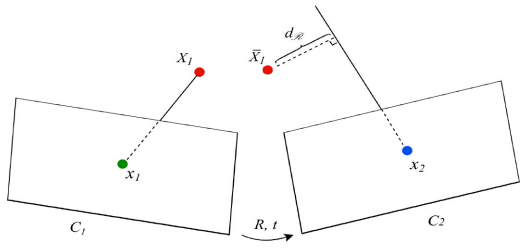



Fig. 6. An intuitive illustration of point-to-ray error. $x_1$ and $x_2$ are feature observations of the same point. $X_1$ is the estimated 3D position of the feature point. The residual $d_{\mathcal{R}}$ is the distance between the warped 3D point $\bar{X}_1$ and the ray that starts from the camera origin of $C_2$ and passes through $x_2$.

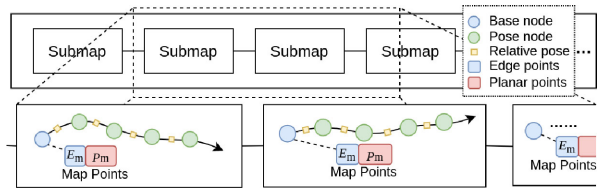



Fig. 7. An intuitive illustration of the maintained submaps in the segment management module.

Relative poses between two synchronized image-LiDAR frames can be estimated by solving the following nonlinear least-square problem:

$$\min_{R,t} \left\{ \sum_{\bar{X}_k^i \in \bar{\mathcal{F}}_k} \rho \underbrace{\left( \| d_{\mathcal{R}} \left( \bar{X}_k^i, X_{k-1}^i \right) \|^2 \right)}_{visual\ point\ constraint} + \sum_{\bar{E}_k^j \in \bar{\mathcal{E}}_k} \rho \underbrace{\left( \| d_{\mathcal{E}}(\bar{E}_k^j) \|^2 \right)}_{lidar\ constraints} + \sum_{\bar{P}_k^m \in \bar{\mathcal{P}}_k} \rho \overbrace{\left( \| d_{\mathcal{P}}(\bar{P}_k^m) \|^2 \right)}^{lidar\ constraints} \right\} \tag{2}$$

where $\rho$ is the Huber norm [25]. To let the point residual have the same metric(in meters) with the LiDAR edge and planar constraints, the visual point constraint is defined as point-to-ray error $d_{\mathcal{R}}(\bar{X}_k^i, X_{k-1}^i)$ instead of the commonly-used reprojection error. An intuitive illustration is given in Fig. 6. Only synchronized image-LiDAR pairs are considered in (2). Since image data usually has a higher frame rate, the images not aligned with the LiDAR data are only used for high-frequency pose propagation. Visual features' inverse depth $\lambda$ is updated by a fixed-lag optimization. The Ceres solver [26] is used for solving the nonlinear problem (2).

### C. Segment Management

The segment management module is responsible for local map construction and pose graph maintenance. Fig. 7 illustrates the data structures in this module: LiDAR point clouds and sensor poses are organized into submaps, each submap contains a pose graph segment and a local point cloud map. Pose graph segment

consists of nodes and edges, where the first inserted pose is labeled the base node. The map point data is separated into edge map $\mathcal{E}_m$ and planar map $\mathcal{P}_m$, represented under the base node frame. Map points are stored into voxels for fast access. For each incoming data pairs (LiDAR point cloud $\mathcal{E}_k$, $\mathcal{P}_k$ and the estimated relative pose to the last sensor frame $k-1$) of frame $k$, we first warp the point clouds to the local map frame using the estimated relative pose, and then perform a scan to map registration every 5 frames by minimizing the distance from each $E_k$ in $\mathcal{E}_k$ and $P_k$ in $\mathcal{P}_k$ to the closest line and plane extracted from $\mathcal{E}_m$ and $\mathcal{P}_m$ respectively to refine the current pose estimation. Detailed description of this scan to map matching process can be found in [15]. After the refinement, the relative pose and the currently estimated map frame pose are added to the pose graph as edge and node respectively. Although the refined pose estimation and the reconstructed local map already have high accuracy, error accumulates over time, and may cause map distortion and inconsistency in the overlapping area. If we only build a single global map, point cloud data is fused to the map scan by scan until the end, and it is difficult to adjust the merged map. A possible solution is to store every scan in the buffer and reconstruct the whole map once the sensor poses change after optimization. However, storing all raw data and rebuilding the map after each optimization step are not suitable for long-running cases. In the proposed system, the whole map is divided into several segments, based on the assumption that the reconstructed maps are of high quality locally. The global map can be adjusted by performing a rigid transformation on the submaps. Since map points are represented under the corresponding base frame in each submap, once the base frame pose being updated, the map points are updated as well. A new submap is created when the distance to the base frame reaches a threshold. If the threshold is too small or too big, there would be too many copies of the local map, or the flexibility of map adjustment would be lost. Extreme cases are creating new submaps every frame or use the same map all the time. The threshold used in the proposed system is 25 m. The latest frame is set to be the base frame in the new submap.

### D. Loop Detection

The loop detection module aims to recognize the current location from the places that other robots or itself has visited and establish data association. The main idea is to find an indicator to evaluate data similarity. Loop detection in the ground and aerial collaborative mapping system can be divided into two subproblems: homogeneous and heterogeneous loop detection. Homogeneous loop detection performs place recognition using data collected by similar platforms with the same hardware configurations. Algorithms [27]–[29] for single-robot loop detection can also be applied to homogeneous platform scenarios. Loop detection on data collected by heterogeneous platforms is much more challenging. Ground and aerial images have large deviation in viewpoint and scale, especially when the aerial camera is mounted down-facingly. For this reason, image-based loop detection methods are not applicable to heterogeneous scenarios.

In this letter, a loop detection method is presented to deal with both homogeneous and heterogeneous scenarios. Inspired by the building outline matching methods [11]–[13] from the remote sensing field, the obstacle (buildings and trees) outlines can be utilized to help detect the loop. Here we assume that there are no big changes in building outlines with elevation. Since a single image or LiDAR scan can only capture limited information about the obstacle outlines, which is insufficient for place recognition, the obstacle outlines are extracted from submap point clouds instead. These outlines are first used to generate thumbnail images of the submap by four steps: First, the ground plane normal $n_p = [n_x, n_y, n_z]^T$ is estimated using RANSAC [30] algorithm. Second, the surfaces parallel to the extracted 3D plane $P_{map}$ are filtered out, and then the remaining point clouds are projected onto $P_{map}$ using the estimated plane normal $n_p$. Third, the plane normal $n_p$ is aligned with the $Z$ coordinate, so that the projected points $C_{map}$ on $P_{map}$ are transformed to $\bar{C}_{map}$ on $X-Y$ plane. Finally, the thumbnail image $B_{map}$ can be generated by simply mapping the $X-Y$ coordinate of each point in $\bar{C}_{map}$ to the pixel coordinate at a specific resolution. Since most of the roof or ground points have already been culled, the valid pixels on $B_{map}$ provide information of the obstacle outlines. The thumbnail images are generated every 40 frames, and each image covers a 50 m $\times$ 50 m area. Some examples of the generated thumbnail images are given in Fig. 8.

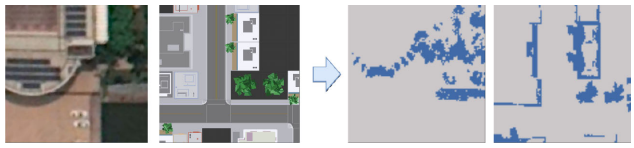


Fig. 8. Examples of the generated submap thumbnail images. The valid pixels on the thumbnail images (right) provides information of obstacle outlines of the environments (left).

When the thumbnail images have been generated, compact descriptors are computed using NetVLAD [29] and these descriptors will be later used for image query. Geometric consistency is also considered during loop detection: Firstly, the base node poses of the existing submaps are maintained in a K-D tree structure and only the $k$ nearest submaps are chosen for similarity test. Next, Normal-Distributions-Transform(NDT) [31] is performed to estimate the rigid transformations between each pair of thumbnail point clouds $C_{map}$ and the matching scores are used for coarse outlier rejection. Lastly, the transformations estimated by NDT are compared to the distance of submap bases so that candidate loops that would bring large deviation are also rejected. Once the loop between two submaps has been detected, the relative pose between the two base frames is computed by calculating the relative transformation between two submap point clouds using Generalized-Iterative-Closest-Point(GICP) [32] algorithm. The relative pose is then added to the pose graph as a constraint edge.

### E. Ground and Aerial Pose Graph Optimization

Let $\mathcal{T} = \{\alpha, \beta, \gamma, \cdots\}$ be the set of all mapping sessions, and $n_\alpha$ denote the number of pose estimations in session $\alpha$. Let $\tilde{x}_{\alpha i} = \begin{bmatrix} \tilde{R}_{\alpha i} & \tilde{t}_{\alpha i} \\ 0 & 1 \end{bmatrix} \in SE(3)$, $\tilde{R}_{\alpha i} \in SO(3)$, $\tilde{t} \in \mathbb{R}^3$ denote the estimated robot pose of the session $\alpha$ at frame $i$, and $\tilde{q}_{\alpha i}$ denote the quaternion representation of $\tilde{R}_{\alpha i}$. The set of all pose estimation is represented as $\mathcal{X}$. Given the relative pose measurement between the $j$th frame of the session $\beta$ and the $i$th frame of the session $\alpha$, the residual on a pose graph edge can be

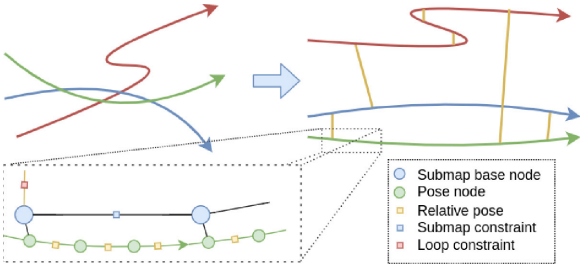



Fig. 9. An intuitive illustration of ground and aerial pose graph optimization, the orange lines indicate the loop closure constraints, the zoom-in view shows part of the pose graph of the green trajectory.

derived by

$$d\left(\alpha i,\ \beta j\right) = \begin{bmatrix} d_t(\alpha i,\ \beta j) \\ d_\theta(\alpha i,\ \beta j) \end{bmatrix} \tag{3}$$

in which

$$d_t(\alpha i,\ \beta j) = t_{\beta j}^{\alpha i} - \tilde{q}_{\alpha i}^{-1} \otimes (\tilde{t}_{\beta j} - \tilde{t}_{\alpha i}) \tag{4}$$

$$d_\theta(\alpha i,\ \beta j) = 2\left[q_{\alpha i}^{\beta j} \otimes (\tilde{q}_{\alpha i}^{-1} \otimes \tilde{q}_{\beta j})\right]_{xyz} \tag{5}$$

where $[q]_{xyz}$ extracts the vector part of a quaternion $q$. Let $\mathcal{L}$ denote the set of all loop closure edges, $from(l)$ and $to(l)$ denote the two pose graph nodes the edge $l$ connected to. Let $s_\beta$ denote the number of submaps in session $\beta$, and $base_\beta(i)$ denote the base pose graph node of submap $i$ in session $\beta$. The pose graph optimization(PGO) problem can be formulated as follows:

$$\begin{aligned} \min_{\mathcal{X}} \Bigg\{ & \sum_{\alpha\in\mathcal{T}, i=0}^{n_\alpha - 1} \rho\underbrace{\left(\|d(\alpha i,\ \alpha i+1)\|^2\right)}_{odometry\ constraint} \\ & + \sum_{l\in\mathcal{L}} \rho\underbrace{\left(\|d(from(l),\ to(l))\|^2\right)}_{loop\ closure\ constraint} \\ & + \sum_{\beta\in\mathcal{T}, j=0}^{s_\beta - 1} \rho\overbrace{\left(\|d(base_\beta(j),\ base_\beta(j+1))\|^2\right)}^{submap\ continuity\ constraint} \Bigg\} \end{aligned} \tag{6}$$

where the odometry, loop closure as well as submap continuity constraints are considered, ensuring that poses and submaps are smoothly connected. $\rho$ is the Huber norm. Fig. 9 gives an intuitive illustration of the pose graph optimization problem. Once the optimization problem is solved, the coordinate of ground and aerial vehicles would be unified and thus the map data from different sessions can be integrated.

Lastly, for each aerial submap, an optional global refinement can be performed by picking the nearest ground neighbour and aerial neighbour and perform GICP refinement using the world-frame pose as the initial guess. The submap pairs with low match scores are also added as pose graph edges. The pose graph optimization problem is solved by Ceres solver [26] in our implementation.

## III. Experimental Results

In this section, each module of the proposed system is evaluated from the experimental aspect under both real-world or synthetic datasets.

(a) Aerial Platform (b) Ground Platform

Fig. 10. Data acquisition platforms used in the experiment.

In real-world experiments, as shown in Fig. 10, a DJI Matrice 600 Pro and a vehicle are used as the data acquisition platforms carrying the sensor module mentioned in Section II-A. The sensor orientation is adjusted to 45° down-facing and 0° front-facing for aerial and ground platforms respectively. The flight height of the aerial platform is around 40 m. GPS-IMU data is used as ground truth for trajectory evaluation.

To enrich the heterogeneous datasets, synthetic data that mimics real-world data is also generated. The simulated sensors are consistent with the real-world configuration. The simulated drones travel and capture sensor data in a 3D city model following preset paths. The city model is generated using Blender. For simplicity, no extra model is used for the simulated ground vehicle because the ground-based data can be obtained by controlling the drones to fly near the ground.

In this section, the necessity of introducing visual constraints in the odometry module is evaluated. Comparisons are also conducted between the proposed loop detection method and recent solutions using RGB images. In addition, Evaluation of the overall collaborative mapping pipeline is presented. All the collected data is processed on the same system with Intel i9-9900 K CPU at 3.6 GHz, a NVIDIA GeForce RTX 2080 Ti GPU, and 64 GB memory.

### A. Evaluation of Odometry Results

The proposed tightly coupled visual-LiDAR odometry module requires time-synchronized RGB images and LiDAR scans as the input. In the experiments, 7 data sequences that include synchronized 20 Hz RGB image data at $1920 \times 1200$ resolution, 10 Hz LiDAR point cloud data and 200 Hz GPS-IMU data are captured. The 7 sequences are collected in the campus, including 4 ground-based sequences and 3 aerial-based sequences with traveling distance ranging from 234 m to 530 m. Challenging scenes such as repetitive textures (Ground 2 and Ground 4) and open environments (Aerial 1) are shown in Fig. 11. GPS-IMU data are used as ground truth for trajectory evaluation.

We use the evo package [33] for odometry evaluation. Two quantitative metrics, absolute pose error (APE) and relative pose error (RPE) are used in the experiments. In APE tests, the translation part (in meters) of the estimated poses $P_{est}$ and the ground truth poses $P_{gt}$ are compared:

$$E_i = P_{gt,i}^{-1}\, P_{est,i} \in \mathrm{SE}(3) \tag{7}$$

$$APE_i = \|\mathrm{trans}(E_i)\| \tag{8}$$

In RPE tests, the relative rotation (in degrees) of the estimated poses $P_{est}$ and the ground truth poses $P_{gt}$ are compared:

$$E_{i,j} = (P_{gt,i}^{-1}\, P_{gt,j})^{-1}(P_{est,i}^{-1}\, P_{est,j}) \in \mathrm{SE}(3) \tag{9}$$

$$RPE_{i,j} = |\mathrm{angle}(\log_{\mathrm{SO}(3)}(\mathrm{rot}(E_{i,j})))| \tag{10}$$

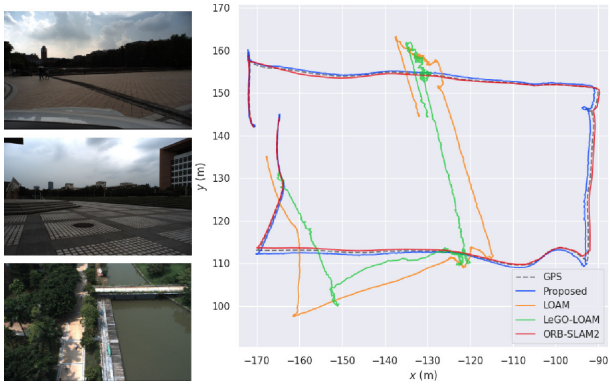



Fig. 11. Left: Some challenging scenes in Ground 2, Ground 4, and Aerial 1 test respectively. Right: A comparison of trajectory estimation results on Aerial 3 test.

TABLE I
COMPARISON OF DIFFERENT ODOMETRY METHODS

| Method | LOAM | | LeGO-LOAM | | ORB-SLAM2 | | Proposed | |
|---|---|---|---|---|---|---|---|---|
| Sequence | APE(m) | RPE(°) | APE | RPE | APE | RPE | APE | RPE |
| Ground 1 | **0.696** | **0.235** | 0.797 | 3.852 | 0.950 | 11.36 | 1.034 | 3.889 |
| Ground 2 | **0.270** | **0.219** | 0.310 | 3.372 | 18.16 | - | 0.354 | 3.347 |
| Ground 3 | **0.400** | **0.180** | 0.477 | 2.701 | 1.313 | 9.184 | 0.805 | 2.691 |
| Ground 4 | **1.613** | **0.539** | **1.271** | 10.65 | - | - | 2.235 | 12.18 |
| Aerial 1 | 29.50 | - | 24.34 | - | 9.671 | 7.183 | 1.219 | **4.362** |
| Aerial 2 | 6.856 | 3.573 | 17.33 | - | 4.607 | 7.349 | 2.396 | **3.199** |
| Aerial 3 | 24.79 | - | 26.58 | - | **0.431** | 2.522 | 0.805 | **0.675** |
| Mean-G | 0.745 | **0.294** | **0.714** | 5.144 | 5.105 | 6.847 | 1.107 | 5.526 |
| Mean-A | 20.38 | 3.573 | 22.75 | - | 4.903 | 5.685 | **1.473** | **2.745** |
| Mean | 9.162 | **0.949** | 10.16 | 5.144 | 5.018 | 6.266 | **1.264** | 4.334 |

The RPE is measured every meter. Three other odometry methods are compared, including LiDAR-based method LOAM [15] and LeGO-LOAM [24] as well as visual-based method ORB-SLAM2 (mono) [34]. The loop detection modules in these methods are disabled for odometry-only comparison. Because the estimated trajectories and the GPS data are represented under different coordinates and the scale is unknown in the monocular version of ORB-SLAM2, the trajectory coordinate, as well as the scale are aligned before the trajectory evaluation.

The experiment results are shown in Table I, where the RMSE(root-mean-square-error) data is listed and the bold fonts indicate the best results. It is shown that the proposed visual-LiDAR odometry module outperforms vision-only or LiDAR-only methods on APE tests on average and also has the best APE and RPE results in aerial tests. Vision-only method ORB-SLAM2 has the largest APE result in Ground 2 test and fails in Ground 4 test due to repetitive textures and rapid movements. LOAM and LeGO-LOAM perform well in Ground tests but they fail to provide reliable pose estimation results in aerial cases. A visual result of Aerial 3 test is presented in Fig. 11. It can be seen that considering both visual and LiDAR constraints in the odometry module can significantly improve the pose estimation robustness and overall accuracy, especially for aerial cases. The proposed visual-LiDAR odometry module provides an effective scheme for both ground and aerial platforms and is beneficial to the ground-aerial collaborative mapping scenarios.

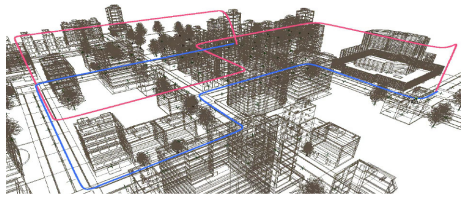


Fig. 12. The ground and aerial trajectories of the simulation data. The texture of the city model is hidden for better visualization of the trajectories.

### B. Evaluations of Ground and Aerial Loop Detection

To quantertively evaluate the performance of the proposed loop detection method, we need more loop data. For homogeneous case, a possible solution is to extend the dataset by the trajectory itself so that all frames can be treated as candidate loops [35]. When it comes to heterogeneous case, the ground and aerial data should have enough trajectory overlapping so that loop can ideally be detected everywhere and the ability to distinguish different scenes can be easily assessed. There are more flexibilities to generate such data under simulation environment, and thus synthetic datasets are used in the experiments. In the simulations, the planned trajectories for the simulated aerial and ground vehicles are the same except the flight altitude, and the aerial/ground data is obtained via controlling the simulated aerial vehicles to fly at higher or lower altitudes respectively. A visualization of the trajectories is given in Fig. 12. The captured data is comprised of 10 Hz RGB images at $752 \times 480$ resolution and 10 Hz LiDAR point clouds.

The captured data is processed using the proposed loop detection scheme. As described in Section II-D, the thumbnail images are generated and the compact descriptors $h$ are then computed. The distance of the compact descriptors is used to measure the similarity of two frames $i$ and $j$ :

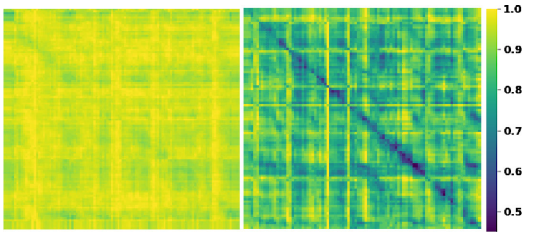
$$d_f(i,j) = \|h_i - h_j\| \tag{11}$$

If the distance is small, the two frames are considered to be similar. After comparing frame $i$ with all candidate frames, the difference scores are then normalized by dividing them by the largest response relative to frame $i$ :

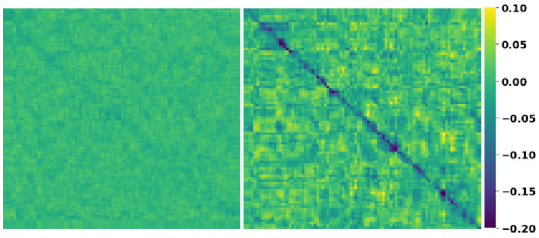
$$\bar{d_f}(i,:) = \frac{d_f(i,:)}{max(d_f(i,:))} \tag{12}$$

Therefore, a difference matrix $D$ of the ground and aerial data can be constructed by computing $d_f$ for each ground-aerial frame pairs. To reduce the effect of the scene ambiguity, the rank of the difference matrix $D$ is reduced by removing $5$ biggest eigenvalues of it [35], [36]. The rank reduced difference matrix $D_r$ is used for detecting candidate loops. Two difference matrices are constructed. The first one is generated using the image descriptors that are directly computed on the captured RGB images. The second one is generated using the image descriptors computed on the submap thumbnails described in Section II-D. A visual comparison of these two matrices and their rank reduced form is presented in Fig. 13.

Since the ground and aerial platforms follow the same trajectory but at different altitudes, the cross-platform loop closure should ideally be detected all the time and the difference matrix should have small values on the diagonals. It can be seen from Fig. 13(b) that the difference matrix generated by the proposed method has more obvious diagonal pixels, indicating that the




(a) The difference matrices.




(b) The difference matrices after rank reduction.

Fig. 13. A visual comparison of difference matrices generated using NetVLAD (left) and proposed method (right).

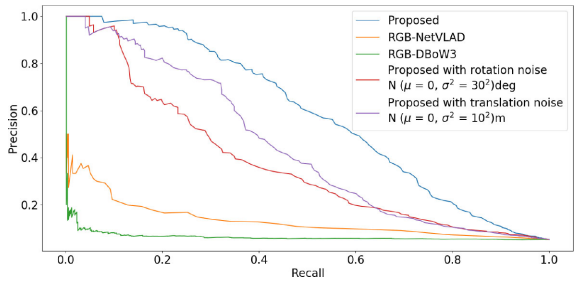
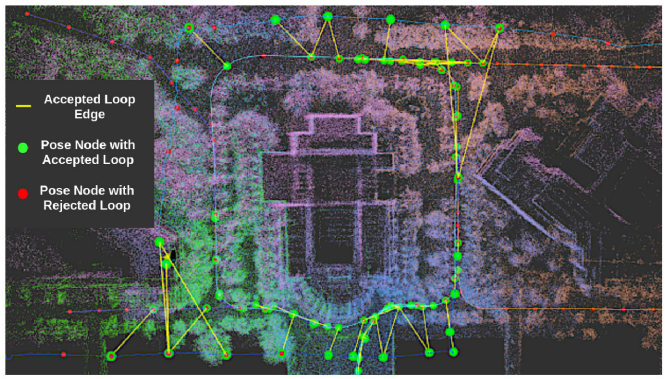



Fig. 14. The precision-recall curve comparison of the ground and aerial loop detection algorithms.

proposed loop detection method is superior to recognize similar scenes from ground and aerial heterogeneous data.

To better assess the cross-platform loop detection performance, we take all possible values in the two reduced rank matrices as the discrimination thresholds, and take adjacent frames within a window of length 3 as ground truth loops. The obtained precision-recall(P-R) curves are shown in Fig. 14. An additional P-R result of DBoW3 [27] method on raw RGB image data is also generated for comparison. Also, tests for not-well-generated thumbnails are performed by adding a Gaussian noise on orientation or translation when the thumbnail images are generated. It can be seen that when the recall rate is around 40%, the proposed method has about 80% precision. Even with noise added, the same precision level can still be reached by slightly sacrificing the recall rate. By contrast, the precision of the RGB-NetVLAD/RGB-DBoW3 methods drops below 20%/10%, respectively. A 80% precision with around 40% recall rate is effective for ground and aerial collaborative mapping problem, since the candidate loops will be further narrowed down using the geometry and registration validation described in Section II-D. Once a loop is found, the cross-platform coordinate deviation can be eliminated by the pose graph optimization. Notice that the proposed method is more sensitive to the heading change, and thus the trajectories need to be carefully designed in practice to let different robots have similar orientations when their trajectories overlap.

(a) Scene Image




(b) Fused map with pose graph.

Fig. 15. The ground and aerial mapping result of real-world data.

TABLE II
EVALUATION OF OVERALL SYSTEM

| Stage | Odometry | | Loop Closure | | Global Refine | |
|---|---|---|---|---|---|---|
| Sequence | APE(m) | RPE(°) | APE | RPE | APE | RPE |
| Ground1 | 1.034 | 3.889 | 0.599 | 3.890 | **0.579** | **3.881** |
| Ground2 | **0.354** | **3.347** | 0.397 | 3.367 | 0.525 | 3.355 |
| Ground3 | **0.805** | 2.691 | 1.119 | **2.541** | 0.872 | 2.693 |
| Aerial1 | 1.219 | **4.362** | 1.052 | 4.363 | **0.885** | 4.367 |
| Aerial2 | 2.396 | **3.199** | 2.442 | 3.295 | **1.935** | 3.204 |
| Mean | 1.162 | 3.498 | 1.122 | **3.491** | **0.959** | 3.500 |

### C. Evaluation of the Complete System

After the assessment of the key modules, the proposed systematic method is validated in a practical ground-aerial collaborative mapping problem. In the experiments, five data sequences (Ground 1-3, Aerial 1-2) from Section III-A are used for heterogeneous collaborative mapping test. Fig. 15 shows the collaborative mapping result, where the map from different sessions are rendered in different colors. The ground and aerial maps can be merged successfully using the proposed collaborative mapping method, and the generated map is rich in details and is also consistent with the real-world scene. Quantity evaluation is performed by comparing the estimated sensor poses before and after optimization. As is shown in Table II although not all robots benefit from the optimization, the overall system error is lowered. This is as expected because when the pose graph of different sessions are connected, the sessions with smaller drift would absorb errors from other sessions, leading to an increase in session error. The time cost of the key operations is presented by Table III. Loop statistics among the five sessions are listed

TABLE III
TIME COST(S)

| Operation | Min | Max | Mean |
|---|---|---|---|
| Depth Completion | 0.01 | 0.05 | 0.01 |
| Pose Estimation | 0.01 | 0.22 | 0.09 |
| Scan to Map | 0.02 | 1.10 | 0.27 |
| Thumbnail Generation | 0.23 | 3.34 | 0.98 |
| NetVLAD | 0.01 | 0.20 | 0.08 |
| Submap Matching | 0.58 | 7.83 | 2.35 |
| PGO | 0.21 | 18.7 | 1.76 |

TABLE IV
LOOP STATISTICS

| Stage | Accept | Reject |
|---|---|---|
| Session 1 | 10 | 0 |
| Session 2 | 14 | 16 |
| Session 3 | 20 | 28 |
| Session 4 | 5 | 9 |
| Session 5 | 4 | 13 |
| Loop Detection | 53 | 66 |
| Global Refinement | 16 | - |

in Table IV. Details about the complete mapping process are shown in our supplementary video.

## IV. CONCLUSION

In this letter, a SLAM-based ground and aerial collaborative mapping system is presented. It comprises of three main parts: a visual-LiDAR odometry module provides robust and accurate pose estimation; a loop detection module deals with cross-platform and large-viewpoint-difference data association problem; a submap management module efficiently organizes the pose graph and point cloud segments together and provides the merged map of the environment. Real-world and synthetic experiments show that the proposed collaborative method is effective in the ground-aerial heterogeneous mapping problem and can provide high-quality mapping results.

In future work, the system will be further optimized by simplifying the map representation and reducing the bandwidth need for data sharing among multiple sessions. Also, IMU constraints will be considered in the odometry module to further improve the robustness and accuracy for pose estimation. The proposed loop detection method will also be improved to handle large heading deviation.

## REFERENCES


[1] C. Cadena*et al.*, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.

[2] X. Gao, S. Shen, Z. Hu, and Z. Wang, "Ground and aerial meta-data integration for localization and reconstruction: A review," *Pattern Recognition Lett.*, vol. 127, pp. 202–214, 2019, advances in Visual Correspondence: Models, Algorithms and Applications (AVC-MAA). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167865518303544

[3] P. Moulon, P. Monasse, R. Perrot, and R. Marlet, "Openmvg: Open multiple view geometry," in *Int. Workshop Reproducible Res. Pattern Recognition*. Springer, 2016, pp. 60–74.

[4] C. Wu, "Towards linear-time incremental structure from motion," in *Proc. 2013 Int. Conf. 3D Vis. - 3DV 2013*, 2013, pp. 127–134.

[5] Z. Yang, F. Gao, and S. Shen, "Real-time monocular dense mapping on aerial robots using visual-inertial fusion," in *Proc. Robot. Autom. (ICRA), IEEE Int. Conf.* IEEE, 2017, pp. 4552–4559.

[6] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, and A. W. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *10th IEEE Int. Symp. Mixed Augmented Reality, ISMAR 2011, Basel, Switzerland, Oct. 26-29, 2011*, 2011.

[7] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "Elasticfusion," vol. 35, no. 14, pp. 1697–1716, 2016.

[8] L. Zhou, S. Zhu, T. Shen, J. Wang, T. Fang, and L. Quan, "Progressive large scale-invariant image matching in scale space," in *ICCV*, 2017.

[9] Q. Shan, C. Wu, B. Curless, Y. Furukawa, C. Hernandez, and S. Seitz, "Accurate geo-registration by ground-to-aerial image matching," *Proc. Int. Conf. 3D Vis., 3DV 2014*, pp. 525–532, Feb. 2015.

[10] X. Gao, L. Hu, H. Cui, S. Shen, and Z. hu, "Accurate and efficient ground-to-aerial model alignment," *Pattern Recognition*, vol. 76, Nov. 2017.

[11] B. Yang, Y. Zang, Z. Dong, and R. Huang, "An automated method to register airborne and terrestrial laser scanning point clouds," *Isprs J. Photogrammetry Remote Sens.*, vol. 109, pp. 62–76, 2015.

[12] W. V. Hansen, H. Gross, and U. Thoennessen, "Line-based registration of terrestrial and airborne lidar data," Mccarthy, 2010.

[13] L. Cheng, L. Tong, M. Li, and Y. Liu, "Semi-automatic registration of airborne and terrestrial laser scanning data using building corner matching with boundaries as reliability check," *Remote Sens.*, vol. 5, no. 12, pp. 6260–6283, 2012.

[14] Z. Ji and S. Singh, "Aerial and ground-based collaborative mapping: An experimental study," in *Proc. The 11th Conf. Field Service Robot.*, 2017.

[15] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time," in *Proc. Robot.: Sci. Syst. Conf.*, 2014.

[16] H. Surmann, N. Berninger, and R. Worst, "3d mapping for multi hybrid robot cooperation," 09 2017, pp. 626–633.

[17] A. Gawel, R. Dube, H. Surmann, J. Nieto, R. Siegwart, and C. Cadena, "3d registration of aerial and ground robots for disaster response: An evaluation of features, descriptors, and transformation estimation," 10 2017, pp. 27–34.

[18] D. V. A. Iavarone, "Sensor fusion: Generating 3d by combining airborne and tripod-mounted lidar data," 2008.

[19] M. Fiocco, G. Bostrom, J. G. M. Goncalves, and V. Sequeira, "Multisensor fusion for volumetric reconstruction of large outdoor areas," in *3-D Digital Imag. and Model., 2005. 3DIM 2005. Fifth Int. Conf. on*, 2005.

[20] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Int. Joint Conf. Artificial Intell.*, 1981, pp. 674–679.

[21] J. Ku, A. Harakeh, and S. L. Waslander, "In defense of classical image processing: Fast depth completion on the cpu," in *2018 15th Conf. Comput. Robot Vis.*, 2018, pp. 16–22.

[22] Y. Chen, B. Yang, M. Liang, and R. Urtasun, "Learning joint 2d-3d representations for depth completion," in *The IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019.

[23] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," vol. 2018, 06 2018, pp. 2002–2011.

[24] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 4758–4765.

[25] P. J. Huber, "Robust estimation of a location parameter," *Ann. Math. Statist.*, vol. 35, no. 1, pp. 492–518, 1964.

[26] S. Agarwal and K. Mierle, "Ceres solver," http://ceres-solver.org

[27] D. Filliat, "A visual bag of words method for interactive qualitative localization and mapping," 05 2007, pp. 3921–3926.

[28] H. Jgou, M. Douze, C. Schmid, and P. Perez, "Aggregating local descriptors into a compact image representation," 07 2010, pp. 3304–3311.

[29] R. Arandjelovi, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1437–1451, Jun. 2018.

[30] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, p. 381395, Jun. 1981. [Online]. Available: https://doi.org/10.1145/358669.358692

[31] M. Magnusson, "The three-dimensional normal-distributions transform — an efficient representation for registration, surface analysis, and loop detection," Ph.D. dissertation, 12 2009.

[32] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp," in *Robot.: Science Systems*, vol. 2, no. 4. Seattle, WA, 2009, p. 435.

[33] M. Grupp, "evo: Python package for the evaluation of odometry and slam," https://github.com/MichaelGrupp/evo, 2017.

[34] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[35] G. Xiang and Z. Tao, "Loop closure detection for visual slam systems using deep neural networks," in *2015 34th Chin. Control Conf.*, 2015.

[36] K. L. Ho and P. Newman, "Detecting loop closure with scene sequences," *Int. J. Comput. Vis.*, vol. 74, no. 3, pp. 261–286, 2007.