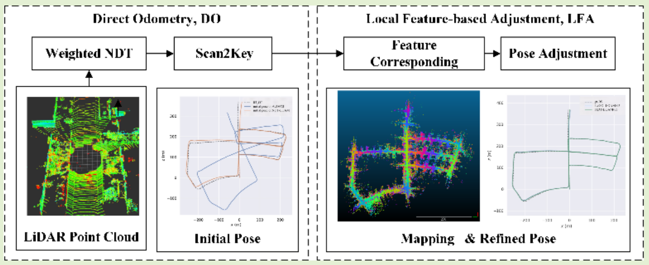
NDT-LOAM: A Real-Time Lidar Odometry and Mapping With Weighted NDT and LFA

Shoubin Chen, Hao Ma, Changhui Jiang, *Student Member, IEEE*, Baoding Zhou, Weixing Xue, Zhenzhong Xiao, and Qingquan Li

***Abstract*—The Lidar Simultaneous Localization and Mapping (Lidar-SLAM) processes the point cloud from the Lidar and accomplishes location and mapping. Lidar SLAM is usually divided to front-end odometry and back-end optimization, which can run parallelly to improve computation efficiency. The font-end odometry estimates the Lidar motion through processing the point clouds and the Normal Distributions Transform (NDT) algorithm is usually utilized in the point clouds registration. In this paper, with the aim to reduce the accumulated errors, we proposed a weighted NDT combined with a Local Feature Adjustment (LFA) to process the point clouds and improve the accuracy. Cells of the NDT are weighted according to the range's values and their surface characteristics, the new cost functions with weight are constructed. In the experiments, we tested NDT-LOAM on the KITTI odometry dataset and compared it with the state-of-the-art algorithm ALOAM/LOAM. NDT-LOAM had 0.899% average drift in translation, better than ALOAM and at the level of LOAM; moreover, NDT-LOAM can run at 10 Hz in real-time, while LOAM runs at 1 Hz. The results display that NDT-LOAM is a real-time and low-drift method with high accuracy. In addition, the source code is uploaded to GitHub and the download link is https://github.com/BurryChen/lv_slam.**



***Index Terms*—Simultaneous localization and mapping (SLAM), lidar odometry, normal distributions transform (NDT), real-time.**

Manuscript received November 8, 2021; accepted December 6, 2021. Date of publication December 28, 2021; date of current version February 11, 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 42101445 and in part by the China Postdoctoral Science Foundation under Grant 2021M702229. The associate editor coordinating the review of this article and approving it for publication was Dr. Abhishek K. Jha. *(Corresponding author: Qingquan Li.)*

Shoubin Chen is with the School of Architecture and Urban Planning, Shenzhen University, Shenzhen 518060, China, also with the Orbbec Research, Shenzhen 518052, China, also with the Institute of Urban Smart Transportation & Safety Maintenance, Shenzhen University, Shenzhen 518060, China, and also with the Key Laboratory for Resilient Infrastructures of Coastal Cities, Shenzhen University, Ministry of Education, Shenzhen 518060, China (e-mail: shoubin.chen@whu.edu.cn).

Hao Ma is with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China (e-mail: mahao_fido@szu.edu.cn).

Changhui Jiang is with the Department of Photogrammetry and Remote Sensing, Finnish Geospatial Research Institute (FGI), 02430 Masala, Finland (e-mail: changhui.jiang@nls.fi).

Baoding Zhou and Weixing Xue are with the Institute of Urban Smart Transportation & Safety Maintenance, Shenzhen University, Shenzhen 518060, China, and also with the Key Laboratory for Resilient Infrastructures of Coastal Cities, Shenzhen University, Ministry of Education, Shenzhen 518060, China (e-mail: bdzhou@szu.edu.cn; weixingxue@whu.edu.cn).

Zhenzhong Xiao is with Orbbec Research, Shenzhen 518052, China (e-mail: xiaozhenzhong@orbbec.com).

Qingquan Li is with the School of Architecture and Urban Planning, Shenzhen University, Shenzhen 518060, China (e-mail: liqq@szu.edu.cn).

Digital Object Identifier 10.1109/JSEN.2021.3135055

I. Introduction

SIMULTANEOUS Localization and Mapping (SLAM) issue has been widely studied in the robotics community, the key is how to construct or update a map of an unknown environment while simultaneously keeping track of an agent's location [1]–[3]. Typically, a SLAM system is divided into front-end and back-end parts; specifically, the front-end part is usually an odometry and the agent's motion is estimated through process the measurements from the SLAM sensors (Camera or Lidar); the back-end accomplishes the map construction, and reduces accumulated drift by loop closure detection and map optimization [4]. SLAM technologies have been extensively investigated and employed in self-driving cars, unmanned aerial vehicles, planetary rovers, newer domestic robots, indoor localization system and even inside the human body [5]–[8]. According to the sensors utilized in the SLAM, it can be mainly divided into visual SLAM/ odometry [9]–[12] and Lidar SLAM/ odometry [13]–[20].

Visual SLAM processes the images sequences and estimates the agent's motion and map the surrounded environment. The Visual SLAM approaches can be generally divided into two categories: indirect (feature-based) methods and direct methods [11]. Similarly, this concept can be extended to Lidar SLAM. Lidar SLAM processes the point clouds collected by Lidar and the motion is estimated thought the point clouds matching. Indirect Lidar SLAM methods previously compute correspondences of an intermediate representation from raw sensor measurements, and then estimate sensor motion by minimizing the re-projection error. The precomputation step of correspondences is typically solved by extracting and matching a sparse/dense set of key points (or feature points) and the parametric representations of other geometric primitives, such as line or curve segments. Direct methods skip this precomputation step and directly employ the sensor measurements (photometric measurements of cameras or geometric quantities of Lidars), jointly estimating motion and correspondences.

Compared with camera, Lidar can acquire huge-scale and high-accuracy measurements without being affected by the lighting conditions [21]. Lidar has become a vitally important sensor in SLAM applications recently. In this work we focus on Lidar Odometry, we present a real-time Lidar odometry (NDT-LOAM) using an improved weighted normal distributions transform (NDT) and a keyframe matching strategy. The cells in NDT are treated with different weights based on their ranges and surface features in the object function of NDT. A simple heuristic is further employed to select the ktyyframe. After compared with state-of-the-art method Lidar Odometry And Mapping (LOAM) [15], the proposed method shows a satisfactory performance on both efficiency and accuracy.

II. Related Work

Lidar has obvious advantages in environment sensing, and it has attracted wide attention in SLAM. Researchers has published extensive works on Lidar SLAM including improving its efficiency, reducing the accumulated errors and map optimization. Before the popularity of Graph-based SLAM, Bayesian filter techniques is a method of choice for model acquisition in the Lidar SLAM community [22]. Bayesian-based SLAM treats the SLAM problem as a pose state estimation problem at a mobile measurement platform, such as Hector SLAM [23], Gmapping [24], etc. While continuously merging new observation data, it uses Extended Kalman Filter (EKF), Particle filter (PF) or Information Filter (IF) to enhance and update the pose state. Graph-based SLAM is also named full SLAM [25], and representative open source algorithms are Karto SLAM [24], cartographer [14], etc. It constructs a graph whose nodes correspond to the platform poses at different points in time and whose edges represent constraints between the poses obtained from observations, and then the map can be computed by finding the spatial configuration of the nodes that is mostly consistent with the measurements modeled by the edges [26].

For the recent advances of 3D Lidar SLAM with six degrees of freedom (6-DOF), we present three state-of-the-art approaches: Cartographer, LOAM, and IMLS-SLAM. Google's Cartographer [14] combines scan-to-submap matching with loop closure detection and graph optimization. Local submap trajectories are created using an algorithm called correlative scan matching (CSM). In the back-end, all scans are matched to nearby submaps using pixel-accurate CSM to create loop closure constraints. The constraint graph of submap and scan poses is periodically optimized in the backend. Because CSM is based on the probability grid without the pre-computed correspondence, Cartographer is a direct method. In this approach, loop closure detection and graph optimization can correct the accumulated error and improve the mapping accuracy. But these calculations are too time-consuming for limited CPU to correct location error in real-time.

LOAM developed by Carnegie Mellon University (CMU) have been ranked high on the KITTI odometry benchmark for several years. It divides the SLAM problem into two modules. One module performs odometry at a high frequency but with low accuracy to estimate velocity of the laser scanner. Although not necessary, if an IMU is available, it can provide a motion prior and mitigate for gross, high-frequency motion. A second module runs at a frequency of an order of magnitude lower for fine matching and registration of the point cloud. Specifically, the correspondences are previously found and determined in both modules, which extract feature points located on sharp edges and planar surfaces, and match the feature points to edge line segments and planar surface patches, respectively. Obviously, LOAM belongs to the feature-based methods. Combination of the two modules allows the map creation in real-time manner. However, to reach the possible maximum accuracy on the KITTI odometry benchmark, the Lidar mapping runs actually at the same frequency as the Lidar odometry and processes each individual scan, running at 1 HZ, 10% of the sensor's real-time speed. Moreover, researchers have implemented some LOAM variants, including LeGO-LOAM [18] with ground-optimized process, LIO-SAM [19] with tightly-coupled Lidar inertial odometry, Loam_livox [18] with a solid-state Lidar Livox, R-LOAM [27] with point-to-mesh features, F-LOAM[1], A-LOAM[2] etc.

Recently, IMLS-SLAM [13] is an another low-drift SLAM algorithm based only on 3D Lidar data. It relies on a scan-to-model matching framework. This direct method first uses a specific sampling strategy based on the Lidar scans, but does not build an explicit correspondence relationship. It then defines the model as the previous localized Lidar sweeps and use the Implicit Moving Least Squares (IMLS) surface representation. This approach needs 1.3s to finish each individual scan and cannot run in real time.

In the concept "Simultaneous Localization and Mapping", the word "Simultaneous" specifically reveals the importance of the real-time processing for the odometry, the frontend of SLAM, while pursing the high localization accuracy. The above methods gain a higher odometry accuracy at the expense of more time consumption than the real-time processing. They cannot solve the contradiction between processing efficiency and localization accuracy. The NDT-LOAM in this paper aims to deal with the problems of accuracy and efficiency. It utilizes an improved weighted NDT and a keyframe matching strategy, and achieve a low-drift, real-time and state-of-the-art SLAM solution based on 3D Lidar data.

III. Methodology

A complete SLAM generally includes two parts: front-end odometry and back-end optimization. The paper focuses on the front-end, and presents an efficient Lidar odometry (NDT-LOAM). The system architecture is presented in Fig. 1, and it divides the SLAM front-end problem into

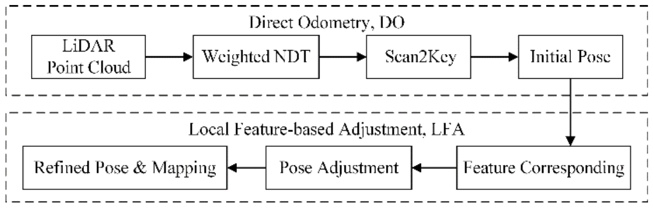


Fig. 1. System overview of NDT-LOAM.

two modules: direct odometry (DO) and local feature-based adjustment (LFA).

The DO module utilizes an improved weighted normal distributions transform (wNDT) and a keyframe matching strategy Scan2Key. The cells are distributed by different weights based on their ranges and dimensionality features in the object function of NDT. Further, a simple heuristic is employed to select the keyframe. In the original LOAM, feature points located on sharp edges and planar surfaces are extracted, and the feature points to edge line segments and planar surface patches are matched respectively. Obviously, original LOAM belongs to the feature-based methods. Differently, our odometry method utilizes the normal distributions transform (NDT) method instead of extracting feature points to the scans matching directly and efficiently in the odometry module, in other words, the NDT based odometry is named as direct odometry (DO) in Fig. 1.

The LFA module is similar to the mapping of the LOAM algorithm, and consists of two steps: feature corresponding, and pose adjustment. With the initial pose, the corner and surface features are processed and feature corresponding is established between the current frame and local point cloud map from the existing historical frames. Then, the pose adjustment refines the initial pose results from DO by optimize the cost functions from feature corresponding.

A. Classical NDT

NDT was firstly proposed by Biber and Straßer in 2003 [28] as a method for 2D scan registration, and then was described in detail for 3D point cloud by Martin Magnusson [29]. NDT maps and transforms a point cloud to a piecewise continuous function consisting of a set of local probability density functions (PDFs) of Gaussian normal distributions, each of which describes the surface shape in an occupied space cell (a square in the 2D case, or a cube in the 3D).

The Gaussian PDF in each cell can be interpreted as a generative process for the surface point p within a cell. In other words, it is originally assumed that the location of p is drawn from this distribution. The mean μ and covariance Σ of the distribution are computed as

$$\mu = \frac{1}{\mathrm{n}} \sum_{i=1}^{n} P_i \tag{1}$$

$$\Sigma = \frac{1}{n-1} \sum_{i=1}^{n} (P_i - \mu)(P_i - \mu)^T \tag{2}$$

where $p_{i=1,\ldots,n}$ are the positions of n point samples contained in a cell. The corresponding PDF is

$$f(x) = \frac{1}{(2\pi)^{D/2}\sqrt{|\Sigma|}} \exp\left(-\frac{(P_i-\mu)^T \Sigma^{-1} (P_i-\mu)}{2}\right) \tag{3}$$

which also is the likelihood of having measured x in a D-dimensional coordinate system.

NDT registration is to estimate the rigid transformation parameter T between a pair of scans that maximizes the likelihood that points of the source scan lie on the target scan surface. The object function of NDT is

$$\begin{aligned} F(\xi) &= \sum_{k=1}^{n} f(T(\xi, P_k))) \\ &= \sum_{k=1}^{n} -d_1 \cdot \exp\left(-\frac{d_2}{2}\left(T(\xi, P_k) - \mu_{P_k}\right)^T \right. \\ &\quad \left. \times\ \Sigma^{-1}\left(T(\xi, P_k) - \mu_{P_k}\right)\right) \end{aligned} \tag{4}$$

$$T(\xi, P_k) = \exp(\xi^\wedge) \cdot P_k = RP_k + t \tag{5}$$

where μ_{P_k} is the mean value of the cell that P_k belongs to. $T(\xi, P_k)$ is to transform the point P_k with the transformation matrix composed of the rotation matrix $\mathrm{R} \in \mathrm{SO}(3)$ and translation vector $t = [t_\mathrm{x}, t_\mathrm{y}, t_\mathrm{z}]$. In SE(3), the transformation matrix T composed of R and t can be encoded using the six-dimensional parameter vector $\xi = \left[\mathrm{t_x}, \mathrm{t_y}, \mathrm{t_z}, \varphi_\mathrm{x}, \varphi_\mathrm{y}, \varphi_\mathrm{z}\right]$, where $\varphi_\mathrm{x}, \varphi_\mathrm{y}, \varphi_\mathrm{z}$ are the Euler angles or a rotation vector corresponding to the rotation matrix R. The parameters d_1, d_2can be calculated by the outliers constants c_1, c_2, and they are expressed as [29]

$$\begin{aligned} d_3 &= -\log(c_2), \\ d_1 &= -\log(c_1 + c_2) - d_3, \\ d_2 &= -2\log((-\log(c_1\exp(-1/2) + c_2) - d_3)/d_1). \end{aligned} \tag{6}$$

B. Weighted NDT

In Classical NDT, all cells having enough points are equally weighted to build the registration objective function. In fact, the cells with different ranges and different dimensionality features have different effects on calculating the transformation parameters [15].

Firstly, we discuss the relationship between the long measurement range and the weight. For a long measurement range comparing one nearby the sensor center, the same rotation error can cause more significant distortions in the point cloud, which means that the longer the distance, the stronger the constraint. Therefore, we introduce a range weight $\mathrm{w_r}$

$$Wr_k = |x_k|. \tag{7}$$

where k is the index of the point.

Secondly, the NDT can also be described as a method for compactly representing a surface. The transformation maps a point cloud to a smooth surface representation, described as a set of local PDFs. Each PDF with the mean μ and covariance Σ describes the shape and dimensionality feature of a local surface in a cell.

The covariance Σ in each cell is a symmetric positive definite matrix, and it can be conducted an eigenvalue decomposition for the eigenvectors and eigenvalues. The eigenvalues are positive and ordered so that $\lambda_1 \geq \lambda_2 \geq \lambda_3 > 0$ Various geometrical parameters can be derived from the eigenvalues and eigenvectors of the covariance matrix Σ. Several indicators have already been proposed in [30], and the following indicators have been selected to express the linear, planar, and spherical indices within a cell [31]:

$$\begin{aligned} &\forall j \in [1,3], \quad \sigma_j = \sqrt{\lambda_j} \\ &a_{1D} = \frac{\sigma_1 - \sigma_2}{\sigma_1} \\ &a_{2D} = \frac{\sigma_2 - \sigma_3}{\sigma_1} \\ &a_{3D} = \frac{\sigma_3}{\sigma_1} \end{aligned} \tag{8}$$

and, the dimensionality index (1D, 2D or 3D) of a cell is defined by:

$$d^* = \arg\max_{d\in[1,3]}[a_{dD}] \tag{9}$$

The cells with different dimensionality indices have different impacts on NDT [13]. The more similar the local surface is to the 2D plane, the more accurate the local PDF reflects the true physical surface in point cloud. Therefore, the cells with higher planar index or 2D feature contribute more to NDT registration, which deserves a higher weight. Because of the sparse of the point cloud, it is possible that just one scan line is located in several cells. These cells with linear behavior or 1D feature cannot present the physical surface accurately and should be distributed less weight. The weight ratio of the 1D, 2D and 3D feature cells is obtained from experience, and the results achieve higher accuracy when the weights are set as follows:

$$WD = \begin{cases} 1.25(d^* = 2) \\ 1.0(d^* = 3) \\ 0.75(d^* = 1) \end{cases} \tag{10}$$

Therefore, the object function of weighted NDT is

$$\begin{aligned} F(\xi) &= \sum_{k=1}^{n} W_k \cdot f(T(\xi, P_k)) \\ &= -W_k d_1 \sum_{k=1}^{n} \exp\left(-\frac{d_2}{2}\left(T(\xi, P_k) - \mu_{P_k}\right)^T \right. \\ &\quad \left. \times\ \Sigma^{-1}\left(T(\xi, P_k) - \mu_{P_k}\right)\right) \end{aligned} \tag{11}$$

$$W_k = Wr_k \cdot WD_k \tag{12}$$

For brevity, let $P_k' \equiv T(\xi, P_k) - \mu_{P_k}$. In other words, P_k' is the different between the value of the point P_k after transformed by the current pose parameters and the center of the cell which the point belongs to. The entries of the gradient vector g is

$$\begin{aligned} g_i &= \frac{\delta F}{\delta \xi_i} \\ &= \sum_{k=1}^{n} W_k d_1 d_2 \cdot P_k'^T \Sigma_k^{-1} \frac{\delta P'}{\delta \xi_i} \exp\left(-\frac{d_2}{2} P_k'^T \Sigma_k^{-1} P_k'\right) \end{aligned} \tag{13}$$

The entries of the Hessian matrix H are

$$\begin{aligned} H_{ij} &= \frac{\delta^2 F}{\delta \xi_i \delta \xi_j} \\ &= \sum_{k=1}^{n} W_k d_1 d_2 \cdot \exp\left(-\frac{d_2}{2} P_k'^T \Sigma_k^{-1} P_k'\right) \\ &\quad \cdot (-d_2(P_k'^T \Sigma_k^{-1} \frac{\delta P_k'}{\delta \xi_i})(P_k'^T \Sigma_k^{-1} \frac{\delta P_k'}{\delta \xi_j}) \\ &\quad + P_k'^T \Sigma_k^{-1} \frac{\delta^2 P_k'}{\delta \xi_i \delta \xi_j} + \frac{\delta P_k'}{\delta \xi_j}^T \Sigma_k^{-1} \frac{\delta P_k'}{\delta \xi_i}) \end{aligned} \tag{14}$$

The parameter ξ can be solved through nonlinear iterations of the Newton method with line search [32] by maximizing the object function (8),

$$\xi \leftarrow \xi - \alpha H^{-1} g \tag{15}$$

where α is a factor determined by the line search method.

C. Keyframe Strategy

Data accumulation and temporal inference are important steps to mitigate the drift of the odometry. Especially for a methodology with sparse point clouds as presented here, the matching with several frames make it less susceptible to errors, compared with matching among consecutive frame. Therefore, we adopt a keyframe matching strategy for the Lidar odometry. If the current scan frame S_t in timestamp t is matched to the last scan frame S_{t-1} in timestamp *t-1* by NDT registration, the matching strategy of the consecutive frame is demoted as Scan2Scan. While the current scan frame $\mathrm{S_t}$ is matched to the recent keyframe in timestamp *t-n* (*n* $\geq$*1)*, we denote the keyframe matching strategy as Scan2Key. It's important to note that the Scan2Key strategy is just utilized in direct odometry rather than LFA, as shown in Fig. 1. In addition, we do not discuss the backend of SLAM in this paper, and the keyframes are independent of the loop closure and global optimization. For the keyframe selection, we employ a simple heuristic: if the translation relative to the last keyframe attains to a distance threshold ε_{dis}, or the rotation relative to the last keyframe attains to a threshold ε_{deg}, or the time interval from the last one is more than a time threshold ε_t, the frame is inserted as a keyframe. In the practical experiment, the distance threshold is usually tougher than the other two and it is the first rule of the keyframe selection.

D. Local Feature Adjustment (LFA)

The LFA module matches features in the current scan frame S_t to a surrounding point cloud map Q_{t-1} to further refine the pose transformation. LFA consists of two steps: feature corresponding and pose adjustment. We adopt the Lidar mapping module of LOAM to implement LFA, and the reader can obtain the detailed description from [15].

In the feature corresponding, the algorithm extracts point features from the raw scan points. Point features can be either corner points or surface points. The above odometry estimation yields a 6-DoF pose estimate, which is used as an initial pose estimate for the mapping module. Using initial pose to transforms S_t into the world coordinates, denoted as S_t^w. To find correspondences for the feature points, we store the feature points on the map Q_{t-1} in the cubic areas. The points in the cubes that intersect with the transformed point cloud S_t^w are extracted and stored in a 3D KD-tree. We find the correspondence points in Q_{t-1} within a certain region around the feature points of current scan. We denote corner correspondences as c^{C} and surface correspondences as c^{S} with $c = \{p_1, p_2\}$, where p_1 is a point of the current scan and p_2 is a point of the map, respectively.

Once the feature correspondences are found, residual blocks of pose adjustment can be formed using the corresponding cost functions for corner and surface features, f^{C} and f^{S}, respectively. The outputs of the cost functions are squared residuals, which are weighted by the loss function ρ. Residuals are the Euclidean distances between the points forming the correspondence. Residual blocks are the cost functions wrapped inside a loss function. The resulting residual blocks are added to the optimization function J. The pose adjustment problem is defined as

$$J(q,t) = \sum_{c^S \in c^S} \rho\left(\left\| f^S\left(c^S, q, t\right)\right\|^2\right) + \sum_{c^C \in c^C} \rho\left(\left\| f^C\left(c^C, q, t\right)\right\|^2\right) \quad (16)$$

where c^{C} and c^{S} are the sets of point feature correspondences for corner and surface features, respectively. s^{C} is defined as $s^C = \left\{c_1^C, c_2^C, \ldots, c_n^C\right\}$ and s^{S} is defined as $s^S = \left\{c_1^S, c_2^S, \ldots, c_m^S\right\}$ with n and m being the number of corner and surface feature correspondences, respectively. q denotes the quaternion and t denotes the translation vector, which transforms the feature points of the scan from the Lidar frame to the map frame during the optimization iterations. $||\bullet||^2$ denotes the squared Euclidean distance, which is used as residual.

With the optimized transformation, a second iteration of feature correspondence estimation with a subsequent optimization step is performed, which is the default in the conventional LOAM algorithm. After the two iterations, the feature points of the current scan are inserted into the map using the final optimized transform, which is illustrated by the map building module. Subsequently, the updated map can be used to estimate the 6-DoF pose for the next Lidar scan.

IV. Experiments

We tested the method on the public datasets, KITTI odometry benchmark [33], [34] (Fig. 2(a)) and Our Kylin backpack (Fig. 2(b)). KITTI has been recorded from an autonomous driving platform around Karlsruhe, Germany. As shown in Fig. 2, the platform is equipped with four stereo camera systems (grayscale and color), a Velodyne HDL-64E laser scanner and a high accuracy OXTS RT 3003 GPS/INS localization system. The cameras, laser scanner and localization system are calibrated and synchronized. The datasets contain 11 training sequences with the GPS/INS ground truth provided.

With the odometry development kit from KITTI, the odometry evaluation was carried out through computing average relative translational and rotational errors for all possible subsequences' trajectories of length (100, ..., 800) meters. The rotation error is measured in degrees per meter.

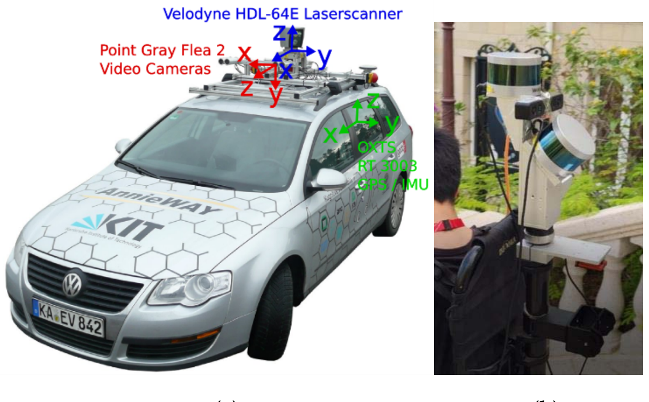

Velodyne HDL-64E Laserscanner
Point Gray Flea 2 Video Cameras
OXTS RT 3003 GPS / IMU
x
y
z
(a)
(b)

Fig. 2. (a) Recording platform used by the KITTI benchmark. The platform is equipped with four stereo camera systems (grayscale and color), a Velodyne HDL-64E laser scanner and a high accuracy OXTS RT 3003 localization system for the ground truth acquisition. The data from the Velodyne Lidar are employed in our method. (b) Our kylin backpack.

The major sensors of the Kylin backpack included: Velodyne VLP-16 Lidar, Mynak D1000-IR-120 color binocular camera, Xsens-300 IMU, and a power communi-cation module. An example of the mobile data collection is presented in Fig. 9. The average speed of the backpack was 1 m/s, and the data collection and algorithm running speed were both 10 Hz. There were two Lidars installed on the backpack. In this experiment, only the horizontal Lidar and the left camera of the binocular camera were used, and the images were 640 × 480, respectively. The backpack was not equipped with a GPS device, so there was no true value of the trajectory.

The experiment is implemented on a laptop with Intel i7-7700HQ CPU (2.8 GHz) and 8GB RAM, on top of the robot operating system (ROS) [35] in Ubuntu 16.04. Moreover, the laser data is logged at 10 Hz. The speed at 0.1 s per scan is the demand of the real-time processing.

A. Accuracy Evaluation of Weighted NDT and Keyframe Strategies

We have introduced two Lidar matching methods: classic NDT and weighted NDT (wNDT), and two key frame strategies: Scan2Scan and Scan2Key in detail. To fully illustrate the accuracy the wNDT and key frame strategy, we tested the KITTI data set. The grid size of NDT is set to 1m, and the key frame selection threshold of Scan2Key is (ε_{dis}, ε_{deg}, ε_t) = (10m, 10°, 1s). Fig. 3 presents the accuracy of the Lidar odometry under the combination of the two matching methods (classic NDT and wNDT) and the two key frame strategies (Scan2Scan and Scan2Key).

In the comparison experiment between classic NDT and weighted NDT, when using the same key frame strategy, except for Scan2Scan's sequence 6 and sequence 9, the offset error of remaining sequences has been significantly reduced after using the wNDT. For the same Scan2Scan, after using the wNDT, the average error of the 11 data set sequences is

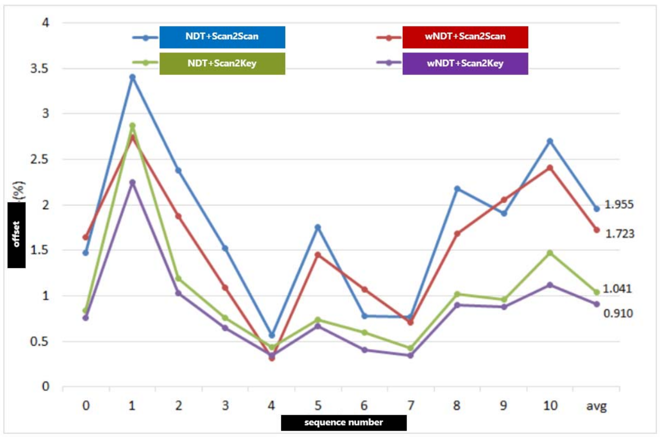


Fig. 3. Accuracy comparison under different matching methods and key frame strategies.

reduced from 1.955% to 1.723%; for the same Scan2key, the average errors of the 11 data set sequences were reduced from 1.041% to 0.910% after wNDT. Compared with the classic NDT, the wNDT reduced the error of DO by about 12%, which shows that the wNDT improves the accuracy of the odometry compared with traditional NDT.

In addition, for the experiments on the 11 data set sequence, the offset of Scan2key is significantly lower than that of Scan2Scan, and the average value of the former is only half of the latter under the same matching method. The experimental results fully demonstrate the effect of Scan2key strategy on the accuracy of the odometry. Therefore, in subsequent experiments, we will select the "wNDT+Scan2Key" combination as our Experimental parameter configuration.

B. Comparison of Initial Pose

Based on that the excellent performance of "wNDT+Scan2Key" configuration validated above, we set this configuration as the initial pose/odometry part of our NDT-LOAM. To compare the accuracy and efficiency of NDT-LOAM and ALOAM in initial pose estimation, we do experiments on the KITTI data set. In terms of feature points based odometry method, the LOAM algorithm proposed by Dr. Ji Zhang is an excellent representative, and it is constructed based on the matching of edge and plane feature points. LOAM consists of two parts: high-frequency odometry and low-frequency mapping. Here, we only adopt the positioning result of its high-frequency odometry as the initial pose. The open LOAM source code version is published very early and less accurate, and the author of the later version with higher accuracy does not open the source. The ALOAM, improved by the Hong Kong University of Science and Technology team is the most advanced version in Lidar-Only LOAM variants with open-source community essence that we can currently obtain. The ALOAM initial pose odometry is used as the representative of the feature points method odometry in the experiment. The accuracy comparison results of NDT-LOAM and ALOAM are shown in Fig. 4 and Tab. I.

TABLE I
Position Error Comparison of Initial Pose

sequence	**ALOAM（%）**	**NDT-LOAM（%）**
#00	4.12	0.76
#01	3.44	2.25
#02	7.35	1.03
#03	3.04	0.65
#04	0.69	0.34
#05	4.29	0.67
#06	0.99	0.41
#07	2.24	0.34
#08	4.89	0.90
#09	6.06	0.88
#10	3.63	1.12
Average	4.913	0.910

TABLE II
Position Error Comparison of Refined Pose

sequence	ALOAM（%）	**FLOAM[36]（%）**	LOAM（%）	NDT-LOAM（%）
#00	0.88	0.92	**0.78**	0.79
#01	2.09	2.80	**1.43**	1.46
#02	4.67	1.56	**0.92**	1.09
#03	0.67	1.09	0.86	**0.65**
#04	0.33	1.43	0.71	**0.31**
#05	0.53	0.79	0.57	**0.54**
#06	0.56	0.72	0.65	**0.56**
#07	0.32	0.54	0.63	**0.27**
#08	1.05	1.11	1.12	**1.04**
#09	0.80	1.28	0.77	**0.74**
#10	1.17	1.77	**0.79**	1.12
Average	1.809	1.27	-	0.899

Tab. I (only odometry part is considered for each algorithm) shows that the positioning errors of the 11 sequences of NDT-LOAM are significantly lower, and the average error of NDT-LOAM is only 1/5 of the feature points method odometry. Fig. 4 shows the trajectories of 11 sequences, the dotted line is the ground truth from GPS/INS, the blue solid line is the trajectory of ALOAM's feature points method odometry, and the orange solid line is the trajectory of NDT-LOAM. As illustrated in Fig. 4, the trajectory of NDT-LOAM is closer to and matches well with the ground truth for all 11 sequences, while that of ALOAM deviates largely from GT. So, it can be concluded that the positioning accuracy of NDT-LOAM is significantly higher than that of features point method odometry in estimating initial pose due to the effectiveness of the proposed wNDT.

C. Evaluation in Refined Pose

What's more, LFA in LOAM is adopted as the mapping part. We carry out experiments on the 11 sequences of KITTI using NDT-LOAM and record the relative errors. Fig. 5 presents the point cloud map from NDT-LOAM and pose trajectories from LeGO-LOAM/NDT-LOAM on three representative KITTI sequences of #00, #05 and #09. The loop-closure is closed in the LeGO-LOAM experiment. In the trajectory picture, the dashed line is GT value, and the green solid line is the positioning track of NDT-LOAM while the blue line is

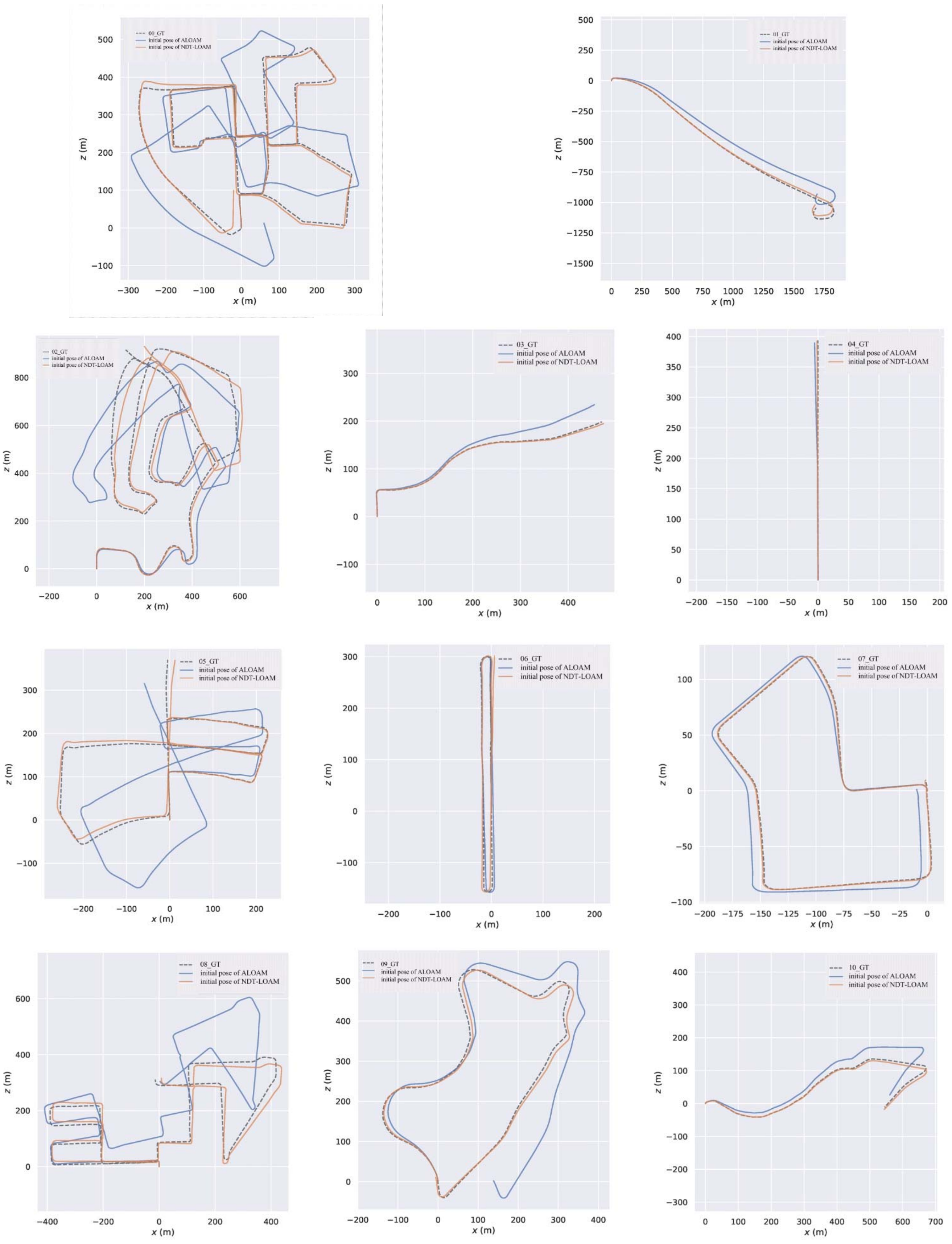


Fig. 4. The initial pose/ odometry trajectory comparison between ALOAM and NDT-LOAM.

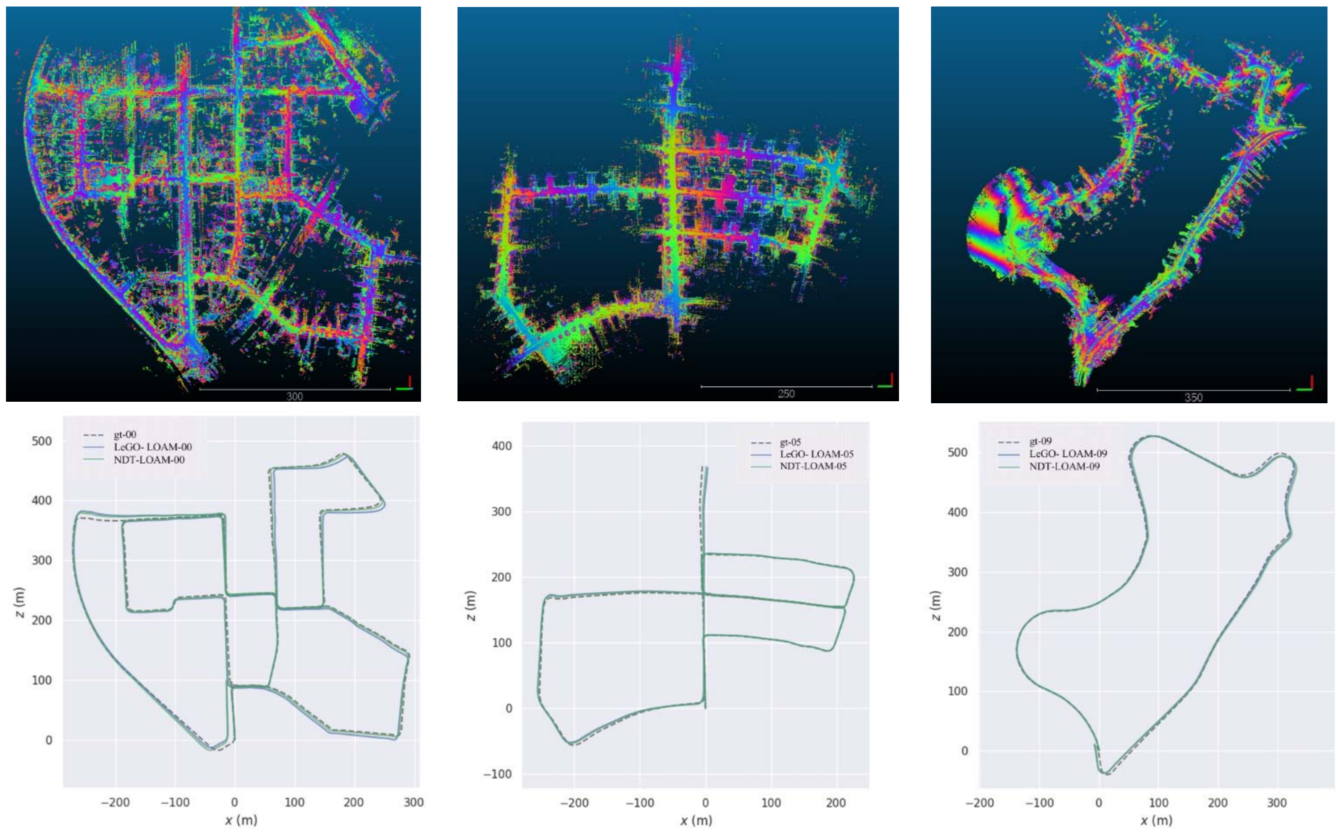


Fig. 5. Point cloud map from NDT-LOAM and pose trajectories from LeGO-LOAM/NDT-LOAM on #00, #05 and #09 KITTI sequences.

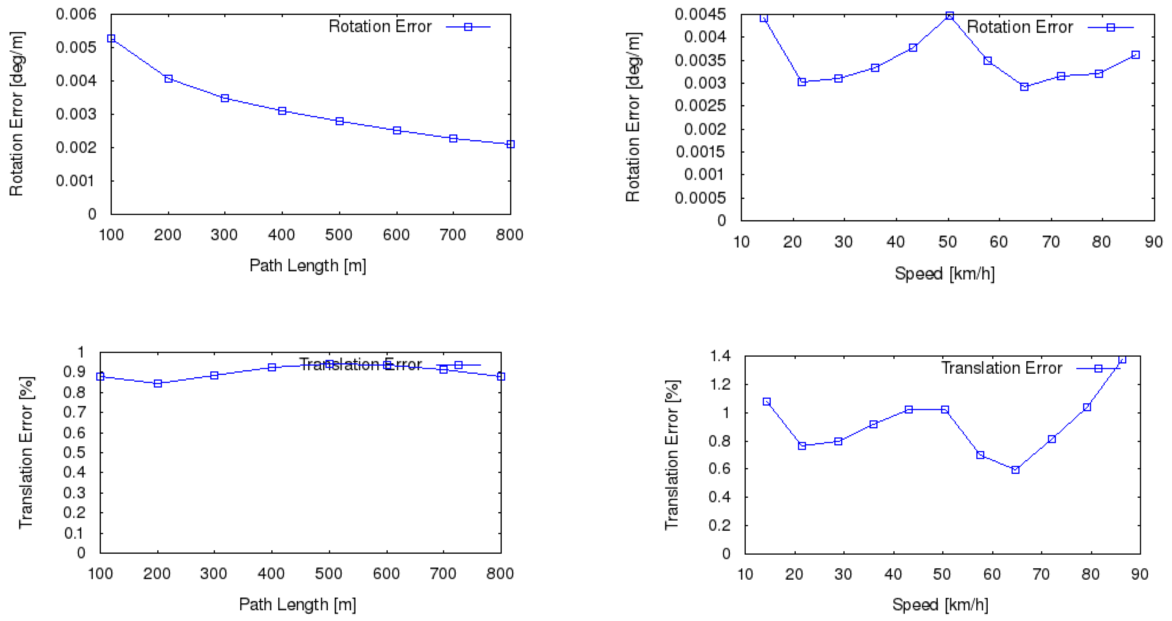
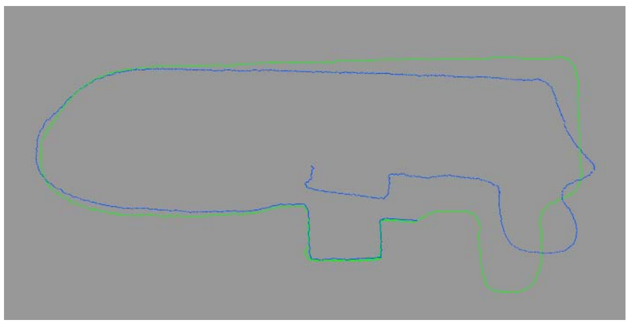
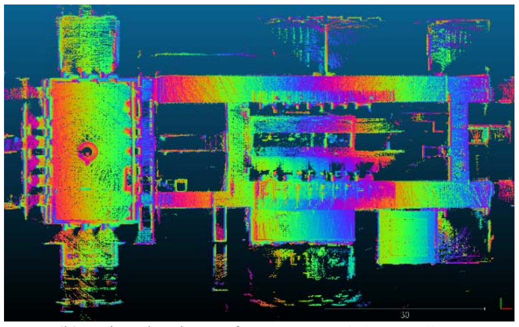


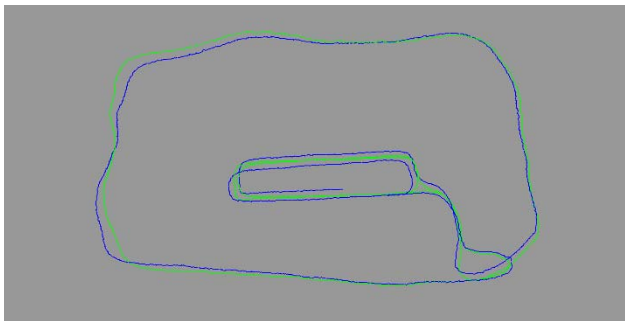
Fig. 6. Average error ratio of NDT-LOAM on 11 KITTI sequence.

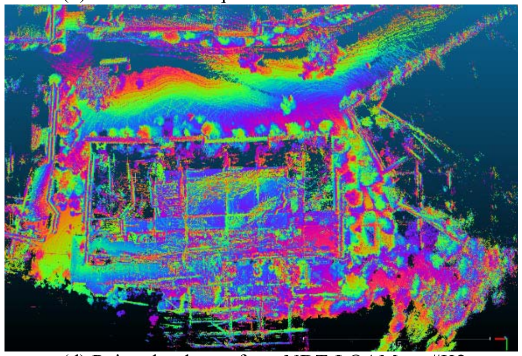
from LeGO-LOAM. The DNT-LOAM trace is closer to the GT of the motion than LeGO-LOAM, showing a relatively high level of accuracy, the resultant three-dimensional point cloud map is intuitive and accurate, and the movement is consistent. Tab. III lists several metrics to evaluate the performance of NDT-LOAM and LeGO-LOAM. Our NDT-LOAM outperforms LeGO-LOAM by almost 50%.

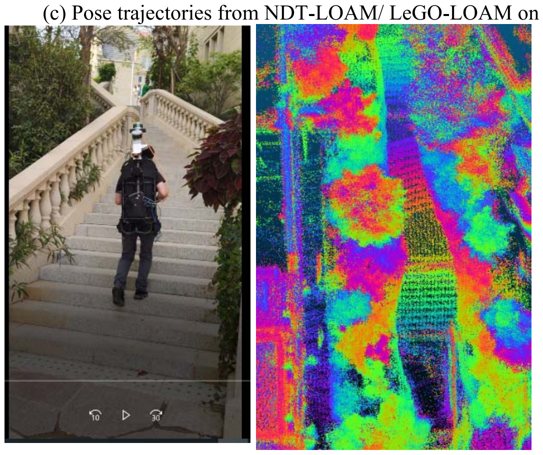
Fig. 6 shows the average angle error ratio and average distance error ratio of the 11 sequences for NDT-LOAM algorithm. It can be seen that the vehicle speeds of the 11 sequences are all between 15km/h and 85km/h. The average angle error ratio decreases as the length of the distance interval increases. It shows a certain wave with the change of the speed interval mobility, and the average angle error of each interval


(a) Pose trajectories from NDT-LOAM/ LeGO-LOAM on #K1


(b) Point cloud map from NDT-LOAM on #K1


(c) Pose trajectories from NDT-LOAM/ LeGO-LOAM on #K1


(d) Point cloud map from NDT-LOAM on #K2


(e) Corresponding point cloud in left ascending staircase of (d)

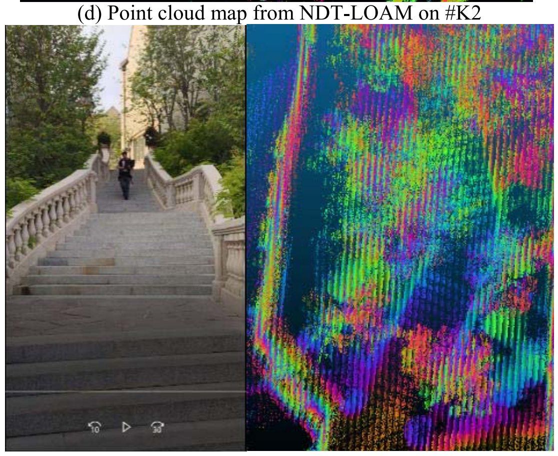

(f) Corresponding point cloud in right descending staircase of (d)

Fig. 7. Pose trajectories from NDT-LOAM/ LeGO-LOAM and point cloud map from NDT-LOAM and on Kylin backpack data.

TABLE III
ABSOLUTE POSE ERROR (m) OF NDT-LOAM AND LeGO-LOAM ON #00, #05 AND #09 KITTI SEQUENCES

sequence	LeGO-LOAM			NDT-LOAM		
	Mean	Std	RMSE	Mean	Std	RMSE
#00	5.16	2.87	5.98	3.38	1.97	3.98
#05	2.65	1.34	2.97	1.99	1.32	2.39
#09	2.00	1.12	2.30	1.38	0.78	1.58

TABLE IV
TRAJECTORY ERRORS OF NDT-LOAM/ LeGO-LOAM

sequence	scene	frame	distance(m)	trajectory error(m)	
				NDT-LOAM	LeGO-LOAM
#K1	indoor	2007	254.58	0.20	7.99
#K2	indoor & outdoor	3877	432.75	0.25	2.60

is between [0.0025 degrees/m, 0.0045 degrees/m]. Average position error ratio remains stable as the length of the distance interval increases and the average position error of each interval is between [0.8%, 0.95%] (the total average position error is 0.899%), indicating that the overall positioning accuracy is relatively stable. The average position error ratio presents a certain degree of volatility, and the average position error of each interval is between [0.6%, 1.4%].

The average position errors of NDT-LOAM and ALOAM/ FLOAM/LOAM are shown in Tab. II. Both ALOAM and NDT-LOAM in this article run at a rate of 10 Hz in real-time level. The results of LOAM are from the original paper, the author sacrifices efficiency for accuracy, and the corresponding speed is 1 Hz. The results of FLOAM are from the paper [36]. From the position error data in Tab. II, we can see the position accuracy of NDT-LOAM on

TABLE V
RUNTIME OF MODULES FOR PROCESSING ONE SCAN (ms)

sequence	LeGO-LOAM(ms)				NDT-LOAM(ms)		
	Segmentation	**Extraction**	**Odometry**	**Mapping**	**DLO**	**LFA-Extraction**	**LFA-Mapping**
#04	21.5	2.2	8.2	106.4	48.2	42.9	86.9
#06	22.7	2.5	8.9	147.4	46.8	42.9	107.8
#07	21.6	2.4	7.4	76.4	38.0	37.1	89.8
#09	55.6	2.0	7.2	96.6	44.5	40.6	90.2
K1	6.2	2.5	4.7	81.3	9.9	12.1	103.7

sequence #00, #03, #04, #05, #06, #07, #09 is slightly higher within 0.8% position errors. While the positioning accuracy of #01, #02, #08, #10 is poor, and the position error is higher than 1.0%. #01 is the highway environment, with few reference targets on both sides of the road, and vehicle movement speed is high (the average speed of #01 is 20m/s, and the value of other data is about 10m/s), which heavily affects the positioning accuracy. The acquisition platform sensors of #00, #01, #02 have poor external calibration accuracy, and this is a key factor that results in big error on sequence #01, #02. The real motion value of #08 has an obvious error in the starting position, reaching to 5 m, which influences the accuracy assessment of #08. Through the observation on the continuous playback of #10 camera pictures, we find that there is a section of data captured on rugged roads where have a large vibration at the level, which may be which may be an important factor affecting the error. Under the same real-time efficiency conditions, the position accuracy of NDT-LOAM is much higher than ALOAM and FLOAM. The position error of the 7 sequence data #03-#09 of NDT-LOAM is slightly lower than LOAM, and the position error of the 4 sequences #01, #02, #10 is slightly higher than LOAM. The overall position of the NDT-LOAM and the original LOAM is basically at the same level. The evo tool is a popular Python package for the evaluation of odometry and SLAM. We also use the tool evaluation the absolute pose error of NDT-LOAM and LeGO-LOAM on #00, #05 and #09 KITTI sequences, and Tab. III lists the result. Our NDT-LOAM outperforms LeGO-LOAM by almost 50%.

D. Kylin Backpack Experiment

With our Kylin backpack, we collected two sequences #K1 and #K2 to assess the proposed method. The #K1 data is from indoor space, while #K2 data includes indoor and outdoor scenes. Due to the lack of high-precision GPS location as the pose ground truth, we consciously walk out of a relatively regular area at the beginning and end of the path, and use the position offset of the beginning and the end as the pose trajectory error of the NDT-LOAM/ LeGO-LOAM method. Some important threshold parameters used in the NDT-LOAM experiment were set as follows: the distance threshold and angle threshold of the key frame selection were 2m and 10°, respectively.

Quantitative results of the NDT-LOAM/ LeGO-LOAM experiment are shown in Tab. IV. The trajectory errors of our method are 0.2 m and 0.25 m for #K1 and #K2 respectively, while LeGO-LOAM has almost 8 m trajectory error for sequence #K1 and 2.6 m error for sequence #K2. More qualitative results are shown in Fig. 7. Blue lines in Fig. 7(a) and 7(c) are the trajectory of LeGO-LOAM, and the green lines are our NDT-LOAM. Our method has lower trajectory error than LeGO-LOAM. Fig. 7(b) and 7(d) show the point cloud map from NDT-LOAM, while Fig. 7(e) and 7(f) show the collection scenes and corresponding point clouds of ascending and descending stairs of #K2. These results indicate that our algorithm has high accuracy and strong robustness.

E. Evaluation of Runtime Performance

We evaluate the time consumption of our method and the baseline LeGO-LOAM. It is concluded that LeGO-LOAM outperforms LOAM in real-time performance [18], so we ignore the comparison with LOAM in this part, readers can reference to the paper [18] for more details. As shown in Tab. V, LeGO-LOAM consists of segmentation, extraction, odometry and mapping modules, and our NDT-LOAM includes DLO, LFA-extraction and LFA-Mapping. Mapping is the most time-consuming module. Since our initial pose is close to the optimized one, it doesn't cost much time to find the refined pose and can run in real-time. In some cases (such as #4, #6, and #9), mapping module takes smaller time than that of LeGO-LOAM. Overall, we can see that the NDT-LOAM algorithm achieved a high level of accuracy and resolved the conflict between accuracy and efficiency.

V. CONCLUSION

In the concept "Simultaneous Localization and Mapping", the high localization accuracy and the real-time processing are equally important. Some previous methods gain a higher odometry accuracy at the expense of more time consumption than the real-time processing. In this paper we propose an efficient Lidar odometry method using an improved wNDT and a keyframe matching strategy. In the experiments, we tested NDT-LOAM on the KITTI odometry dataset and compare it with the state-of-the-art algorithm LOAM. The results illustrate that NDT-LOAM is an efficient odometry method with low drift.

Since the current method does not involve loop closure and graph optimization in the backend of SLAM, our future work is to improve the approach to fix motion estimation drift by closing the loop. Also, we will develop the multi-sensor (Visual/Lidar/IMU) integration technology for further reducing the motion estimation drift.

References

[1] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): Part II," *IEEE Robot. Autom. Mag.*, vol. 13, no. 3, pp. 108–117, Sep. 2006.

[2] 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.

[3] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, "Visual simultaneous localization and mapping: A survey," *Artif. Intell. Rev.*, vol. 43, no. 1, pp. 55–81, 2015.

[4] X. Gao, R. Wang, N. Demmel, and D. Cremers, "LDSO: Direct sparse odometry with loop closure," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 2198–2204.

[5] P. Mountney, D. Stoyanov, A. Davison, and G.-Z. Yang, "Simultaneous stereoscope localization and soft-tissue mapping for minimal invasive surgery," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.*, 2006, pp. 347–354.

[6] B. Zhou *et al.*, "Crowdsourcing-based indoor mapping using smartphones: A survey," *ISPRS J. Photogramm. Remote Sens.*, vol. 177, pp. 131–146, Jul. 2021.

[7] X. Liu *et al.*, "Kalman filter-based data fusion of Wi-Fi RTT and PDR for indoor localization," *IEEE Sensors J.*, vol. 21, no. 6, pp. 8479–8490, Mar. 2021.

[8] J. Chen *et al.*, "A data-driven inertial navigation/Bluetooth fusion algorithm for indoor localization," *IEEE Sensors J.*, early access, Jun. 15, 2021, doi: 10.1109/JSEN.2021.3089516.

[9] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1052–1067, Jun. 2007.

[10] 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.

[11] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 611–625, Mar. 2018.

[12] J. Li *et al.*, "Attention-SLAM: A visual monocular SLAM learning from human gaze," *IEEE Sensors J.*, vol. 21, no. 5, pp. 6408–6420, Mar. 2020.

[13] J.-E. Deschaud, "IMLS-SLAM: Scan-to-model matching based on 3D data," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 2480–2485.

[14] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LIDAR SLAM," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2016, pp. 1271–1278.

[15] J. Zhang and S. Singh, "Low-drift and real-time lidar odometry and mapping," *Auton. Robots*, vol. 41, no. 2, pp. 401–416, Feb. 2017.

[16] X. Chen, A. Milioto, E. Palazzolo, P. Giguere, J. Behley, and C. Stachniss, "SuMa++: Efficient LiDAR-based semantic SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 4530–4537.

[17] J. Lin and F. Zhang, "Loam livox: A fast, robust, high-precision LiDAR odometry and mapping package for LiDARs of small FoV," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, 2020, pp. 3126–3131.

[18] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2018, pp. 4758–4765.

[19] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2020, pp. 5135–5142.

[20] Y. Pan, P. Xiao, Y. He, Z. Shao, and Z. Li, "MULLS: Versatile LiDAR SLAM via multi-metric linear least square," 2021, *arXiv:2102.03771*.

[21] S. Chen *et al.*, "Extrinsic calibration of 2D laser rangefinders based on a mobile sphere," *Remote Sens.*, vol. 10, no. 8, p. 1176, Jul. 2018.

[22] S. Thrun and M. Montemerlo, "The graph SLAM algorithm with applications to large-scale mapping of urban structures," *Int. J. Robot. Res.*, vol. 25, nos. 5–6, pp. 403–429, 2006.

[23] S. Kohlbrecher, O. von Stryk, J. Meyer, and U. Klingauf, "A flexible and scalable SLAM system with full 3D motion estimation," in *Proc. IEEE Int. Symp. Saf., Secur., Rescue Robot.*, Nov. 2011, pp. 155–160.

[24] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE Trans. Robot.*, vol. 23, no. 1, p. 34, Feb. 2007.

[25] R. Vincent, B. Limketkai, and M. Eriksen, "Comparison of indoor robot localization techniques in the absence of GPS," *Proc. SPIE*, vol. 7664, 2010.

[26] K. Konolige, G. Grisetti, R. Kümmerle, W. Burgard, B. Limketkai, and R. Vincent, "Efficient sparse pose adjustment for 2D mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 22–29.

[27] M. Oelsch, M. Karimi, and E. Steinbach, "R-LOAM: Improving LiDAR odometry and mapping with point-to-mesh features of a known 3D reference object," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 2068–2075, Apr. 2021.

[28] P. Biber and W. Strasser, "The normal distributions transform: A new approach to laser scan matching," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Jun. 2003, pp. 2743–2748.

[29] M. Magnusson, "The three-dimensional normal-distributions transform: an efficient representation for registration surface analysis and loop detection," Ph.D. dissertation, Örebro Univ., Örebro, Sweden, 2009.

[30] K. F. West, B. N. Webb, J. R. Lersch, S. Pothier, J. M. Triscari, and A. E. Iverson, "Context-driven automated target detection in 3D data," in *Proc. 14th Autom. Target Recognit.*, 2004, pp. 133–144.

[31] J. Demantké, C. Mallet, N. David, and B. Vallet, "Dimensionality based scale selection in 3D lidar point clouds," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vols. 38-5/W12, pp. 97–102, Sep. 2012.

[32] K. Madsen, H. Nielsen, and O. Tingleff, *Methods for Non-Linear Least Squares Problems*, 2nd ed. 2004.

[33] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, pp. 1231–1237, Sep. 2013.

[34] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361.

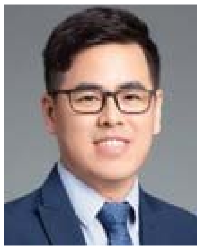
[35] M. Quigley *et al.*, "ROS: An open-source robot operating system," in *Proc. ICRA Workshop Open Source Softw.*, 2009, p. 5.

[36] L. Li *et al.*, "SA-LOAM: Semantic-aided LiDAR SLAM with loop closure," 2021, *arXiv:2106.11516*.

Shoubin Chen received the Ph.D. degree in photogrammetry and remote sensing from Wuhan University, Wuhan, China, in 2020. He is currently a Postdoctoral Researcher with the School of Architecture and Urban Planning, Shenzhen University, Shenzhen, China. His research interests include robot navigation, LiDAR/Visual SLAM, and multi-sensor fusion.

Hao Ma received the M.S. degree from Central China Normal University, China, in 2014, and the Ph.D. degree from Wuhan University, China, in 2018. She is now doing research with the College of Computer Science and Software Engineering, Shenzhen University, as a Postdoctoral Fellow. Her research interests include binocular stereo matching, multi-source data registration, and StlyeGAN-based image generation.

Changhui Jiang (Student Member, IEEE) was born in Xinghua, China, in 1992. He received the B.S. degree from Soochow University, Suzhou, China, in 2014, and the Ph.D. degree from the Nanjing University of Science and Technology, Nanjing, China, in 2019. From December 2017 to November 2018, he studied as a Visiting Ph.D. student sponsored by the China Scholarship Council, Finnish Geo-spatial Research Institute (FGI). He has authored or coauthored more than 20 journal or conference papers in LiDAR, GNSS, and inertial navigation system. He holds five patents in GNSS and inertial navigation system. His research interests include positioning location and navigation for autonomous driving vehicles and personal smartphone in urban or super-urban signals challenging environments.



Baoding Zhou received the Ph.D. degree in photogrammetry and remote sensing from Wuhan University, Wuhan, China, in 2015. He is currently an Assistant Professor with the College of Civil and Transportation Engineering, Shenzhen University, China. His research interests include indoor localization, indoor mapping, intelligent sensing, and intelligent transportation systems.

Weixing Xue was born in Lu Yi, Henan, China, in 1990. He received the Ph.D. degree in geodesy and survey engineering from the School of Geodesy & Geomatics, Wuhan University, in 2019. He is currently a Postdoctoral Researcher with the Shenzhen Key Laboratory of Spatial Smart Sensing and Service, College of Civil Engineering, Shenzhen University, Shenzhen, China. His research interests include seamless positioning and navigation, multi sensor information fusion and data processing theory, and precision engineering measurement.

Zhenzhong Xiao was born in Shandong, China, in 1980. He received the Ph.D. degree from Xi'an Jiaotong University, Xi'an, China. He is currently one of the founders and the Chief Technology Officer of Orbbec Inc. His research interests include 3D camera research and development, 3D perception, artificial intelligence, and computer vision.

Qingquan Li received the Ph.D. degree in photogrammetry and remote sensing from the Wuhan Technical University of Surveying and Mapping. He is the President of Shenzhen University. For over 20 years, Dr. Li has been engaged in the developments of surveying engineering and their innovation applications, contributing especially to the principles and practices of dynamic precision engineering survey. His research interests include the integration of GNSS, RS and GIS, precision engineering survey, and spatiotemporal big data analytics. His work has been recognized with a variety of distinctions. He is an Academician of IEAS, a member of Committee at Science and Technology Commission, Ministry of Education, China, and the Chief-Editor of the *Journal of Geodesy and Geoinformation Science*. His research was awarded National Scientific and Technological Progress Second Prize (China) and MoE Scientific and Technological Progress First Prize (China).