Ph.D. DISSERTATION

Registration Method of 3D Probabilistic Normal Distributions Transform for Odometry and Mapping

주행계 및 지도 작성을 위한
3차원 확률적 정규분포변환의 정합 방법

BY

HONG HYUN-KI

FEBRUARY 2019

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Ph.D. DISSERTATION

Registration Method of 3D Probabilistic Normal Distributions Transform for Odometry and Mapping

주행계 및 지도 작성을 위한
3차원 확률적 정규분포변환의 정합 방법

BY

HONG HYUN-KI

FEBRUARY 2019

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Registration Method of 3D Probabilistic Normal Distributions Transform for Odometry and Mapping

주행계 및 지도 작성을 위한
3차원 확률적 정규분포변환의 정합 방법

지도교수 이 범 희

이 논문을 공학박사 학위논문으로 제출함

2018년 10월

서울대학교 대학원

전기 컴퓨터 공학부

홍 현 기

홍현기의 공학박사 학위 논문을 인준함

2018년 12월

Abstract

The robot is a self-operating device using its intelligence, and autonomous navigation is a critical form of intelligence for a robot. This dissertation focuses on localization and mapping using a 3D range sensor for autonomous navigation. The robot can collect spatial information from the environment using a range sensor. This information can be used to reconstruct the environment. Additionally, the robot can estimate pose variations by registering the source point set with the model. Given that the point set collected by the sensor is expanded in three dimensions and becomes dense, registration using the normal distribution transform (NDT) has emerged as an alternative to the most commonly used iterative closest point (ICP) method. NDT is a compact representation which describes using a set of GCs (GC) converted from a point set. Because the number of GCs is much smaller than the number of points, with regard to the computation time, NDT outperforms ICP. However, the NDT has issues to be resolved, such as the discretization of the point set and the objective function.

This dissertation is divided into two parts: representation and registration. For the representation part, first we present the probabilistic NDT (PNDT) to deal with the destruction and degeneration problems caused by the small cell size and the sparse point set. PNDT assigns an uncertainty to each point sample to convert a point set with fewer than four points into a distribution. As a result, PNDT allows for more precise registration using small cells. Second, we present lattice adjustment and cell insertion methods to overlap cells to overcome the discreteness problem of the NDT. In the lattice adjustment method, a lattice is expressed as the distance between the cells and the side length of each cell. In the cell insertion method, simple, face-centered-cubic, and body-centered-cubic lattices are compared. Third, we present a means of regenerating the NDT for the target lattice. A single robot updates its poses using simultaneous localization and mapping (SLAM) and fuses the NDT at each pose to update its NDT map. Moreover, multiple robots share NDT maps built with inconsistent lattices and fuse the maps. Because the simple fusion of the NDT maps can change the centers, shapes, and normal vectors of GCs, the regeneration method subdivides the NDT into truncated GCs using the target lattice and regenerates the NDT.

For the registration part, first we present a hue-assisted NDT registration if the robot acquires color information corresponding to each point sample from a vision sensor. Each GC of the NDT has a distribution of the hue and uses the similarity of the hue distributions as the weight in the objective function. Second, we present a key-layered NDT registration (KL-NDT) method. The multi-layered NDT registration (ML-NDT) registers points to the NDT in multiple resolutions of lattices. However, the initial cell size and the number of layers are difficult to determine. KL-NDT determines the key layers in which the registration is performed based on the change of the number of activated points. Third, we present a method involving dynamic scaling factors of the covariance. This method scales the source NDT at zero initially to avoid a negative correlation between the likelihood and rotational alignment. It also scales the target NDT from the maximum scale to the minimum scale. Finally, we present a method of incremental registration of PNDTs which outperforms the state-of-the-art lidar odometry and mapping method.

keywords: normal distributions transform, registration, odometry, mapping

student number: 2013-20911

Contents

List of Tables

List of Figures

Nomenclature

$\mathcal{D}$	NDT, a set of Gaussian components
$\mathcal{D}^{\mathcal{P}}$	point set as a GMM
d	Gaussian component
d_c	distance between centroids of the cells
d_{corr}	corresponding Gaussian component
e_r	rotational error
e_t	translational error
h	hue value
L	layer
L_c	coarse layer
L_f	fine layer
l	side length of the cell (regular hexahedra)
$\mathcal{M}$	multi-layered NDT set

M	map
$n_{\mathcal{D}}$	the number of Gaussian components in $\mathcal{D}$
$n_{\mathcal{P}}$	the number of point samples in $\mathcal{P}$
$\hat{\mathbf{p}}$	measured point sample coordinate
$\mathbf{p}$	point sample coordinate
$\mathcal{P}$	a set of point samples
$\mathbf{R}$	rotation matrix
$\mathbf{t}$	translation vector
r_1, r_2	regularizing factors
s	scaling factor of the covariance matrix
$\mathbf{T}_H$	transformation in the homogeneous form
$\mathcal{V}$	lattice
v	cell
w	weight
$\Delta\boldsymbol{\Theta}$	robot pose variation
γ	step size
$\boldsymbol{\Theta}$	robot pose
λ_i	the ith eigenvalue ($\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$)

μ	mean vector
μ_h	circular mean of the hue
ρ	correlation coefficient
$\mathbf{\Sigma}$	variance-covariance matrix
σ	standard deviation
σ_h	circular variance of the hue

Chapter 1

Introduction

1.1 Background

A device that operates by itself by means of artificial intelligence is called a robot. Examples of successful robotic systems include mobile platforms for planetary exploration, industrial robotics arms on assembly lines, self-driving cars, and manipulators that assist surgeons [1]. The robot must be capable of autonomous navigation to follow a planned path to complete its given tasks, such as cleaning floors, farming, exploring unknown environments, and transporting goods and passengers.

Autonomous navigation consists of four components: localization, mapping, path planning, and control. First, localization estimates the pose, including the position and orientation. Second, mapping integrates the information obtained from the environment. Third, path planning plans the feasible path in response to the situation. Fourth, the control component actuates the moving platform to navigate along the path. These four components are closely related, as shown in Fig. 1.1. Given a goal pose, the path planning component considers the current and goal poses on the map to plan a path,

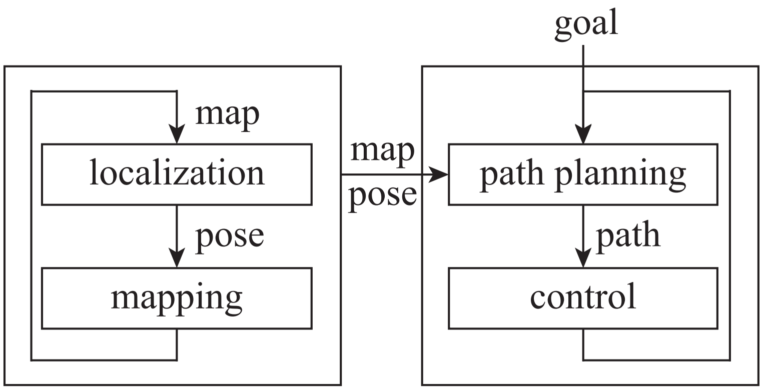


Figure 1.1: Process of autonomous navigation.

and the control component controls the actuators to follow the planned path to reach the goal. Here, to plan and follow the path, an accurate map and accurate poses are required. Given an accurate map, the localization component can estimate the robot pose accurately. Moreover, given an accurate pose, the mapping component can update the map accurately. Because the localization and mapping components are dependent, an approach known as simultaneous localization and mapping (SLAM) has been developed to estimate the pose and build the map simultaneously.

The SLAM problem can be expressed as the graphical model shown in Fig. 1.2 [1]. Given the control $u_{1:t}$, measurement $z_{1:t}$, and constraint $l_{1:k}$, estimating a posterior over pose x_t at time t along with map M

$$p(x_t, M|z_{1:t}, u_{1:t}) \tag{1.1}$$

is known as online SLAM, and estimating the posterior over $x_{1:t}$ along with M

$$p(x_{1:t}, M|z_{1:t}, u_{1:t}) \tag{1.2}$$

is known as full SLAM [1]. Additionally, SLAM can be categorized into filter-based SLAM and graph-based SLAM. Filter-based SLAM recursively estimates x_t and M

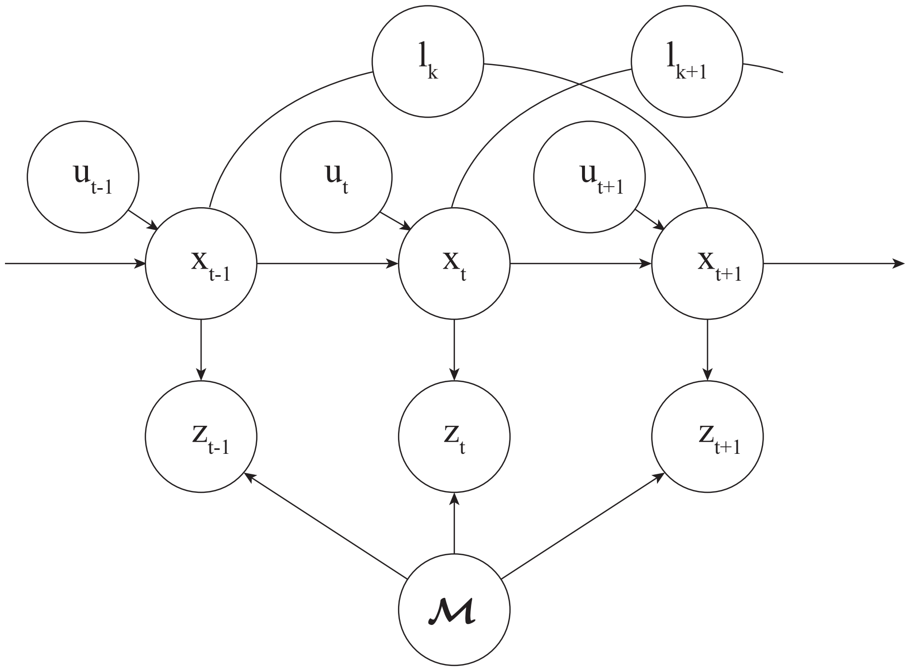


Figure 1.2: Graphical model of SLAM [1, 2].

[3], such as EKF-SLAM using an extended Kalman filter (EKF) [4] and RBPF-SLAM using the Rao-Blackwellized particle filter (RBPF) [5]. Graph-based SLAM can be divided into two processes: front-end and back-end [6]. The front-end forms a graph using sensors, and the back-end optimizes the graph. Graph optimization is time-consuming, but the processing time can be shortened by such methods as incremental smoothing and mapping (iSAM) [7] and general graph optimizer (g2o) [8].

To perform SLAM, the robot can mount devices, as shown in Fig. 1.3. It can estimate its pose using global navigation satellite systems (GNSS), such as GPS or GLONASS [9]. It can also estimate its pose using indoor positioning system (IPS) based on signals such as Wi-Fi [10], RFID [11, 12], and Bluetooth [13]. Moreover, without external references, it can estimate its pose using an inertial measurement unit

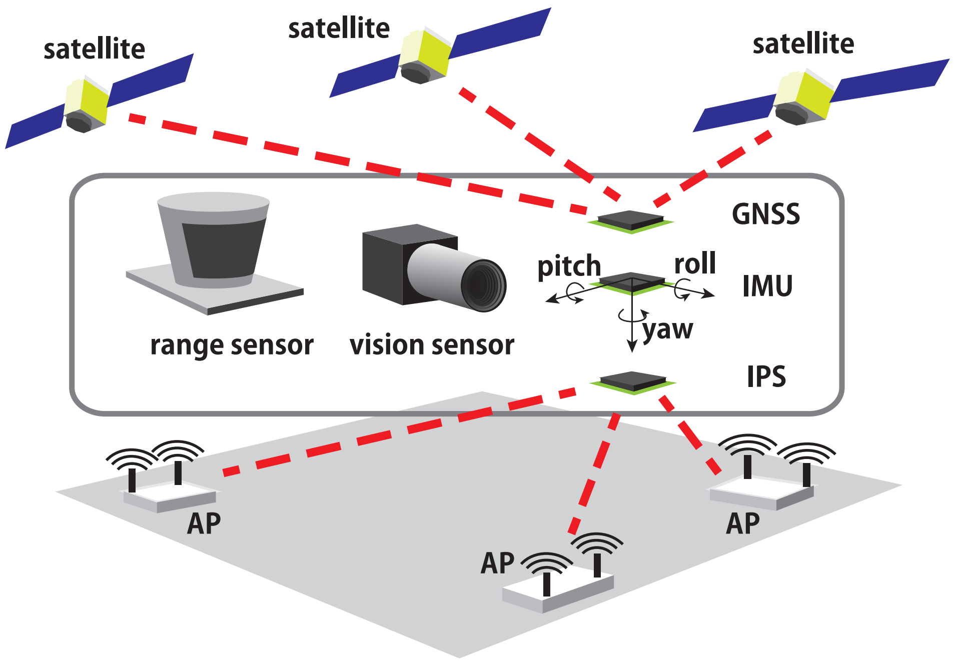


Figure 1.3: Illustration of sensors for SLAM.

(IMU) fused with sensors, such as accelerometers, gyroscopes, and magnetometers [14, 15]. On the other hand, to estimate the pose while also building a map of the environment, the robot can be equipped with vision sensors such as a monocular camera [16], stereo camera [17], or omnidirectional camera [18], or range sensors such as a laser rangefinder (LRF) [19, 20], radar [21], sonar [22], time-of-flight (TOF) depth sensor [23], or structured light (SL) depth sensor [24, 25].

The vision-based odometry estimation method, also known as visual odometry (VO), can be extended to visual SLAM (vSLAM) with global map optimization [26]. VO can be categorized into three approaches: the feature-based method, the direct method, and the learning-based method. The first VO is a feature-based VO which estimates the pose variation based on matching features in a pair of images [27]. Later, parallel tracking and mapping (PTAM) accelerated this method using a parallel processing framework and optimized using via the bundle adjustment (BA) approach [28]. Additionally, ORB-SLAM improved the processing rate using the oriented fast and rotated BRIEF (ORB) feature and improved the localization accuracy by means of graph optimization [29]. The second approach, the direct method, traces the input image without abstraction. After dense tracking and mapping (DTAM) had been proposed to track the pose using the intensity directly [30], large-scale direct SLAM (LSD-SLAM) was proposed to track the pose using semi-dense VO [31] and correct the poses using graph optimization [32]. Moreover, the convolutional neural network (CNN) SLAM was proposed to overcome the difficulty of estimating the scale using the depth learned from the image [33]. Direct sparse odometry (DSO) was also proposed to improve the processing rate by selecting high-intensity points as candidates for reconstruction [34]. Recently, due to the wave of research in deep learning, the third approach, learning-based VO, was proposed. An end-to-end VO with a deep recurrent CNN (RCNN), abbreviated to DeepVO, demonstrated the possibility of learning-based VO [35]. Later, UnDeepVO, which relies on learning through an unsupervised deep learning technique, was proposed to learn the depth map from stereo images and predict the depth map from monocular images [36].

Odometry estimation using a range sensor is based on point set registration given that transformation to match the source point set to the target model can be regarded as the estimated pose variation of the robot. Point cloud registration can be categorized according to the representation, such as the point-to-point [37, 38], point-to-line [39], plane-to-plane [40], point-to-distribution [41, 42], and distribution-to-distribution [43, 44, 45]. Most point set registrations improve the accuracy of transformation estimation in the framework of the iterative closest point (ICP) method [37]. This dissertation also deals with point set registration to estimate the transformation.

It is recommended to use heterogeneous sensors on a robot due to the limitations of each sensor, as follows. It is challenging for the robot to estimate the exact pose from the GNSS device under certain conditions, such as indoors, underwater, underground, in tunnels, in urban canyons, in forests, in caves, during cloudy weather, and during GNSS jamming. In indoor and urban canyons, the robot can estimate the pose using IPS, but IPS is unavailable if the robot is in a suburban location without any access points or during a power outage. In GNSS- and IPS-denied situations, the robot can estimate the pose using other devices. First, the robot can estimate the acceleration and angular velocity using IMU. However, because the pose and the variation are obtained by integrating the acceleration and angular velocity, any errors in the acceleration and angular velocity will accumulate in the pose. In addition to the IMU, the pose can be estimated using vision sensors and range sensors. Although night vision can be learned from day vision [46], it remains challenging to utilize a vision sensor in a dark environment. Thus, using a range sensor instead is recommended. Moreover, to deal with weather conditions such as rain, fog, and snow as well as transparent materials such as glass and acrylic, radar or sonar can be used instead. However, with radar and sonar, interference can arise due to entities which operate on the same frequency. With regard to interference, the robot can utilize other devices to estimate its pose. Because there may be environmental conditions in which one of the sensors operates alone, it is necessary to improve the accuracy of pose estimation using each type of sensor.

Among these sensors, this dissertation deals with lidar-based pose estimation and studies registration methods in an effort to improve the accuracy of the estimated rigid-body transformation.

1.2 Problem Statement

In this dissertation, we describe the robot pose as a parameter vector $\mathbf{\Theta}$ of the homogeneous transformation matrix T_H. Consider that a robot explores an environment, as shown in Fig. 1.4(a), the robot has the map M_{k-1}, as shown in Fig. 1.4(f), and it obtains a set $\mathcal{P}_k$ of point samples at the current pose $\mathbf{\Theta}_k$, as shown in Fig. 1.4(d). After the robot estimates $\mathbf{\Theta}_k$, it transforms $\mathcal{P}_k$ to fuse it with the map M_{k-1} to obtain an updated map M_k, as shown in Fig. 1.4(g).

We define the frame-to-frame registration as a problem of finding an optimal $\Delta\mathbf{\Theta}_k^*$ which transforms the source point set $\mathcal{P}_k$ to register the target point set $\mathcal{P}_{k-1}$. The pose variation $\Delta\mathbf{\Theta}_k$ can be obtained by maximizing the likelihood as

$$\underset{\Delta\mathbf{\Theta}_k}{\arg\max}\, \mathcal{L}(\Delta\mathbf{\Theta}_k; \mathcal{P}_k, \mathcal{P}_{k-1}). \tag{1.3}$$

Similarly, we define the frame-to-map registration as a problem of finding the optimal pose $\mathbf{\Theta}_k^*$ which transforms the source point set $\mathcal{P}_k$ to register the target map model M_{k-1}. The pose can be obtained by maximizing the likelihood as

$$\underset{\mathbf{\Theta}_k}{\arg\max}\, \mathcal{L}(\mathbf{\Theta}_k; \mathcal{P}_k, M_{k-1}). \tag{1.4}$$

The conventional objective functions of NDT registrations are introduced in Chapter 2. Also, the modified objective functions are presented in the rest of chapters.

1.3 Literature Review

1.3.1 Point Set Registration

Point set registration has a history of over 30 years of research. After Haralick et al. proposed a least-squares singular value decomposition (SVD) registration method esti-

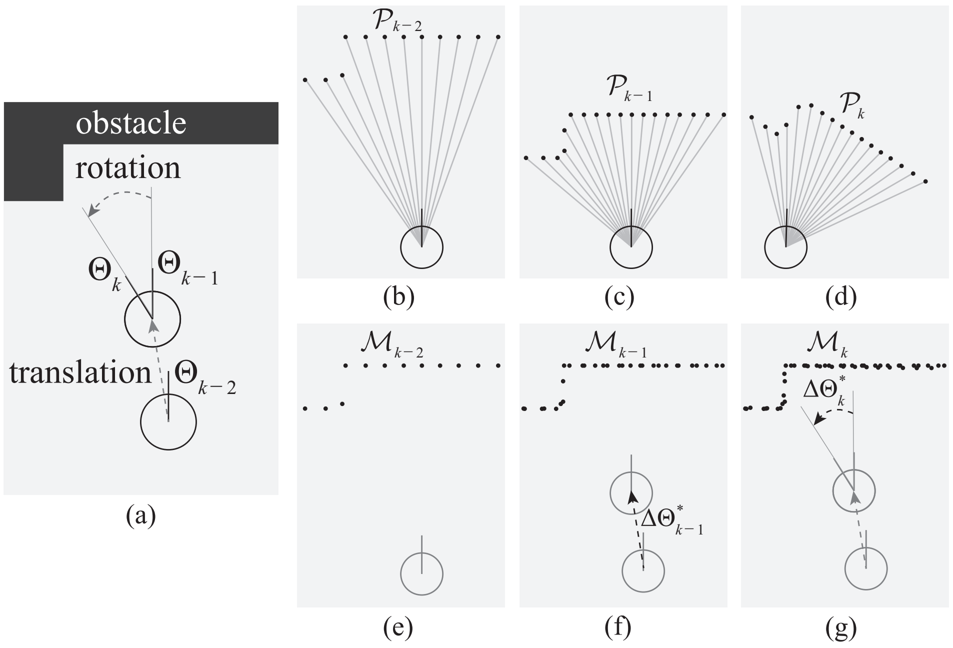


Figure 1.4: Illustration of point sets collected with different poses and maps: robot moving from $\mathbf{\Theta}_{k-2}$ to $\mathbf{\Theta}_k$ in (a); point set $\mathcal{P}_{k-2}$ collected with pose $\mathbf{\Theta}_{k-2}$ in (b); $\mathcal{P}_{k-1}$ collected with pose $\mathbf{\Theta}_{k-1}$ translationally moved from $\mathbf{\Theta}_{k-2}$ in (c); $\mathcal{P}_k$ collected with pose $\mathbf{\Theta}_k$ rotated from $\mathbf{\Theta}_{k-1}$ in (d); the map M_{k-2} updated at $k-2$ in (e);M_{k-1} at $k-1$ in (f);M_k at k in (g).

mating the robot pose [47], Besl and Mckay proposed a method called iterative closest point (ICP) as a shape registration algorithm for point sets, parametric curves, parametric surfaces, implicit curves, implicit surfaces, polylines, and triangle sets [37]. Since then, numerous improved methods have been proposed. Also, numerous approaches of registration have been proposed, such as Gaussian mixture model (GMM) [44], normal distributions transform (NDT) [41], polar scan matching (PSM) [48], and Hough scan matching (HSM) [49]. The evolution of the point set registration can be depicted as Fig. 1.5.

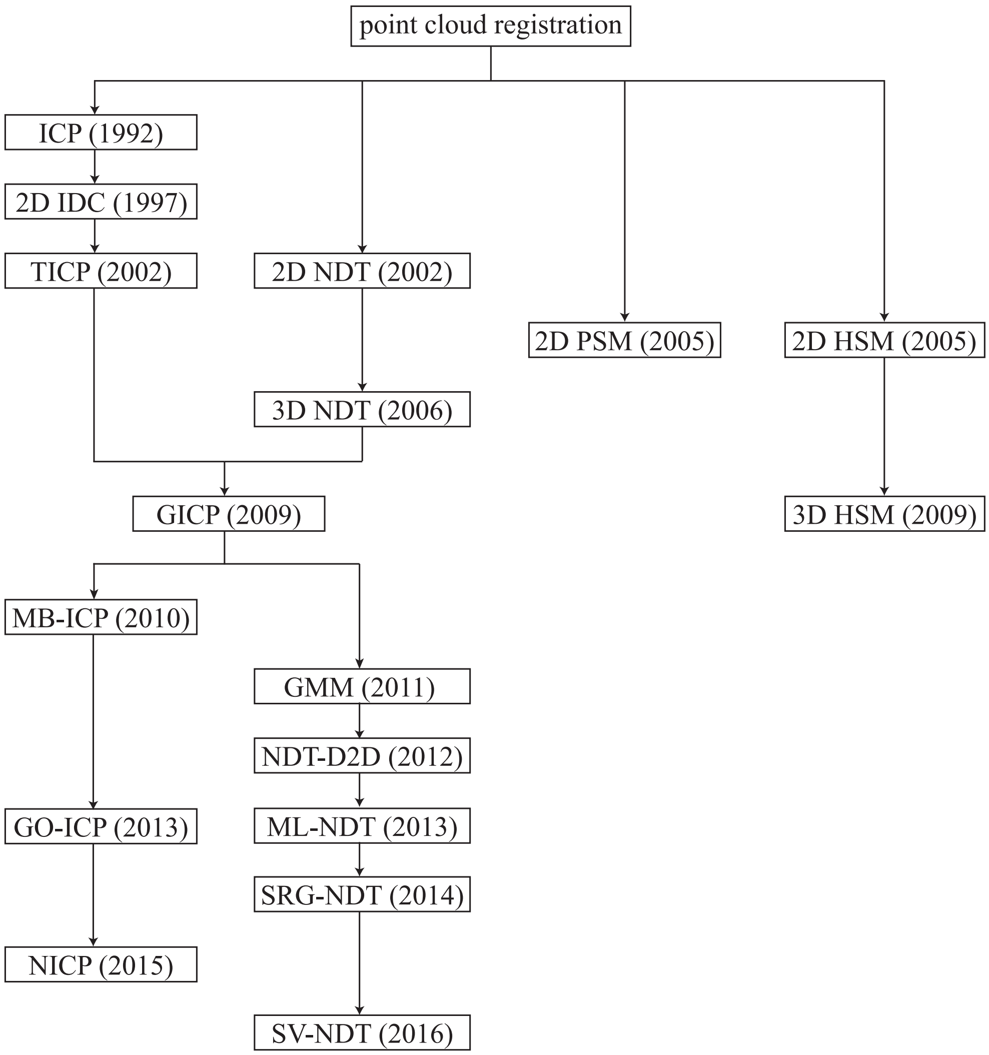


Figure 1.5: Evolution of the point set registration.

ICP

The basic process of ICP can be divided into three components: applying the updated transformation, searching for the correspondence, and optimizing the objective function, such as the point-to-point case:

$$f(\mathcal{P}_s, \mathcal{P}_t, \mathbf{\Theta}) = \frac{1}{n_{\mathcal{P}_s}} \sum_{i=1}^{n_{\mathcal{P}_s}} \|T(\hat{\mathbf{p}}_{s,i}, \mathbf{\Theta}) - \hat{\mathbf{p}}_{c,i}\|^2, \tag{1.5}$$

where $\mathcal{P}_s = \{\hat{\mathbf{p}}_{s,i}\}_{i=1}^{n_{\mathcal{P}_s}}$ is the source point set, $\mathcal{P}_t = \{\hat{\mathbf{p}}_{t,i}\}_{i=1}^{n_{\mathcal{P}_t}}$ is the target point set, $\hat{\mathbf{p}}_c$ is the correspondence of $\hat{\mathbf{p}}_s$, and $n_{\mathcal{P}_s}$ is the number of points in $\mathcal{P}_s$.

In the past 27 years, to improve the performance of the ICP, Numerous methods have been proposed. One of issues is the computational complexity, especially the process of searching for correspondences since the computational complexity of the naive search is $O(n_{\mathcal{P}_s} n_{\mathcal{P}_t})$. A method of using kd-tree effectively was proposed, and it reduced the complexity to $O(n_{\mathcal{P}_s} log(n_{\mathcal{P}_t}))$ [50]. Another issue is outliers of the least values which lead to inaccurate transformation. A trimmed ICP (TrICP) was proposed to deal with the issue [51]. It defines the objective function as the sum of the top n_{trim} least values selected from $n_{\mathcal{P}_s}$ least values. Since $n_{trim} \leq n_{\mathcal{P}_s}$, it can also accelerate the registration. Moreover, the performance of rotational alignment was another issue of ICP. To improve the rotational alignment, an iterative dual correspondence (IDC), which combines ICP with iterative matching-range-point (IMRP), was proposed [52]. Later, a 2D metric-based ICP (MB-ICP) defines the objective function, using a new metric consist of the position and orientation between two points, as

$$f(\mathcal{P}_s, \mathcal{P}_t, \mathbf{\Theta}) = \frac{1}{n_{\mathcal{P}_s}} \sum_{i=1}^{n_{\mathcal{P}_s}} \left(\|\mathbf{e}_i\|^2 - \frac{(e_{x,i}\hat{p}_{s,y,i} - e_{y,i}\hat{p}_{s,x,i})^2}{\|\hat{\mathbf{p}}_{s,i}\|^2 + L^2} \right), \tag{1.6}$$

where L is a positive real number homogeneous to a length, $\hat{\mathbf{p}}_s = (\hat{p}_{s,x}, \hat{p}_{s,y})^T$, $\mathbf{e} =$

$(e_x, e_y)^T$ is computed as

$$\mathbf{e} = T(\hat{\mathbf{p}}_s, \mathbf{\Theta}) - \hat{\mathbf{p}}_c, \tag{1.7}$$

and $T(\cdot, \cdot)$ is an operator which transforms $\hat{\mathbf{p}}_s$ with $\mathbf{\Theta}$ [53]. Also, MB-ICP was extended to the 3D case [54]. The appearance of the generalized-ICP (g-ICP) can be seen as the beginning of the registration of Gaussian mixture models (GMM). g-ICP considers the shape uncertainty of points and uses the sum of Mahalanobis distances as objective function, which can be expressed as

$$f(\mathcal{P}_s, \mathcal{P}_t, \mathbf{\Theta}) = \sum_{i=1}^{n_{\mathcal{P}_s}} \mathbf{e}_i^T \left(C_{c,i} + RC_{s,i}R^T\right)^{-1} \mathbf{e}_i, \tag{1.8}$$

where R is the rotation matrix, C_s and C_c are the variance-covariance matrices of $\mathbf{p}_s$ and $\mathbf{p}_c$, respectively [43]. Later, normal ICP (NICP) was proposed to improve g-ICP [55]. The objective function of NICP is the sum of the Mahalanobis distance of the point pair and their normal vectors. Furthermore, to overcome the problem of local minima, a globally optimal ICP (Go-ICP) which registers point sets with a branch-and-bound (BnB) scheme was proposed [56].

So far, ICP has been modularized, as shown in Fig. 1.6 [57]. First, *data filter* can sample the point samples or add information for the point samples, such as normal, uncertainty, and curvature. Second, *transform* transforms the filtered source point set. Third, *match* links the transformed source points to the target points. Fourth, *outlier filter* can exclude outliers or set weights according to the statistics of distances between the transformed source points and the correspondences. Fifth, *metric minimize* computes the transformation which minimizes the objective function. Sixth, *transformation check* checks whether the termination condition of iteration is satisfied. Usually, the match module is regarded as the major disadvantage of ICP. Although the computational complexity is reduced from $O(n_a n_b)$ to $O(n_b log(n_a))$ using the k-d tree, it

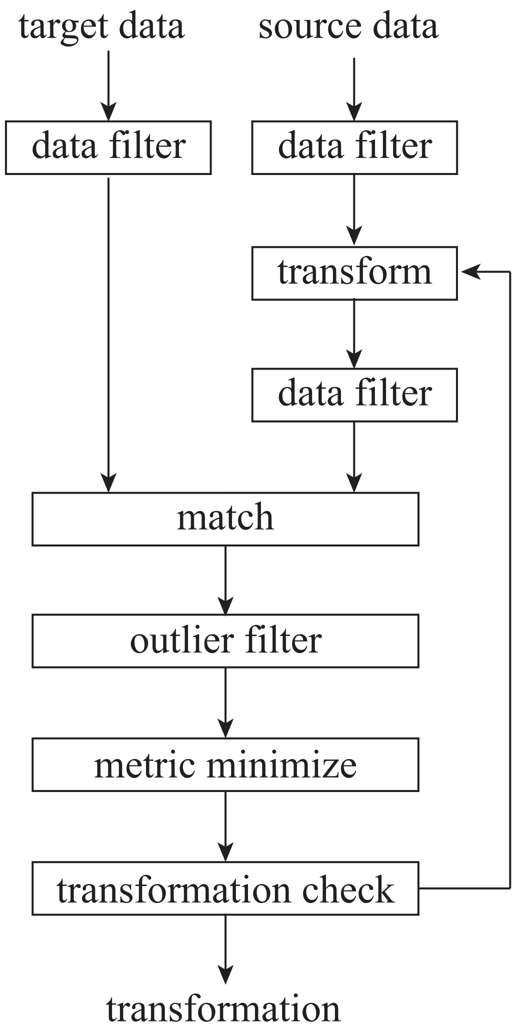


Figure 1.6: Block diagram of ICP.

is still difficult to process over 100k points collected from a 3D range sensor in real-time. Thus, the source and target point set must be sampled, respectively, to reduce the number of points to 5k or less.

NDT and GMM

In 2002, Biber and Straßer proposed 2D normal distributions transform (NDT) which is a compact spatial representation describing the shape in the form of normal distributions [41]. NDT sets regular cubic cells to subdivide the point set $\mathcal{P}$, and points in each cell are converted into a normal distribution. As Stoyanov et al. introduced, NDT is a special case of the Gaussian mixture model (GMM) with uniform weights and largely disjoint components [45]. In [41], Biber and Straßer also proposed a 2D point-to-distribution NDT registration (NDT-P2D) which aligns the source point set $\mathcal{P}_s$ to an NDT converted from the target point set $\mathcal{P}_t$. The corresponding distribution of each source point is the distribution in the cell where the source point is located. As a result, NDT can reduce the time complexity to $O(n_b)$. Also, Takeuchi and Tsubouchi extended the NDT-P2D to 3D case [42]. The authors also proposed a primitive coarse-to-fine NDT-P2D using two-resolution NDT. Later, Magnusson et al. formalized the 3D NDT-P2D and presented the advanced applications [58].

The advantages of the NDT are the number of distributions which is usually much smaller than the number of target points and the geometric representation which is more accurate than the sampled points. NDT-P2D using the NDT was evaluated by Magnusson et al. and compared to ICP. As a result, the NDT-P2D performs more robust and faster than ICP [59]. The processing rate of the NDT-based registration can be further accelerated by registering the source NDT $\mathcal{D}_s$ to the target NDT $\mathcal{D}_t$. Based on this intuition, a distirbution-to-distirbution NDT registration (NDT-D2D) was proposed [45]. NDT-D2D estimates the transformation which registers $\mathcal{D}_s$ to $\mathcal{D}_t$ by minimizing L_2 distance between two NDTs [44]. Since the number of source distributions is also much smaller than the number of source points, its computational complexity $O(n_{D,s})$ is much lower than the computational complexity of NDT-P2D.

The cell size is the most critical when using the NDT representation. High resolution lattice leads to the fast registration but low accuracy, while the low resolution lattice leads to the high accuracy but slow registration. The multi-resolution NDT-P2D, also named as multi-layered NDT (ML-NDT) [60], can be a solution. The ML-NDT roughly registers with the NDT generated by large cells, and then the cell size is reduced to register precisely. Segmented region growing NDT (SRG-NDT) [61] and supervoxel NDT (SV-NDT) [62] are alternative solutions which generate NDT using irregular cells. SRG-NDT converts the points, except for the ground points, into NDT by the region growing method. The processing rate of SRG-NDT is breakneck, and the accuracy is high due to removal of the ground points. However, the processing rate can be changed according to the proportion of the ground points to the total points. On the other hand, SV-NDT clusters the point set using supervoxel method and converts it into NDT, and it shows a high success rate and accuracy of registration.

Smoothing NDT is another issue of NDT. So far, various approaches have been presented to improve the performance of NDT registration, and we found a common property which is smoothing objective function to reduce the local minima problem. This property can be seen in following approaches. NDT generated by the overlapped cells can reduce the discreteness of the NDT [41]. Later, trilinear 3D NDT was proposed in [41, 72]. Even though the trilinear NDT-D2D shows high accuracy, its computation time is eight or more times longer than the conventional NDT-D2D. The hierarchical approach is another good example. It registers P_s to coarse-to-fine NDTs

[42, 58, 60]. Also, a method of choosing resolution by comparing the number of utilized points was presented to accelerate the NDT-P2D [73]. Recently, an uncertainty approach for GMM is presented [74]. According to the paper, the GCs can be expanded by the surface and normal uncertainties. Also, PNDT in Chapter 3 can expand GCs by the expected mean and covariance considering pdf based on the sensor model.

GMM and NDT converts the point set into a set of Gaussian components (GC). NDT, which is generated by regular cells, is a special case of GMM representation. However, the objective functions are different. While the GMM registration uses the Mahalanobis distance, the NDT registration uses the likelihood.

PSM

PSM is a point-to-point registration method. It registers the 2D point set in the polar coordinate and shows more accurate transformation than ICP [63]. It was improved by a scan restoration method to deal with dynamic environments [64]. However, as far as we know, it has not been extended into 3D.

HSM

HSM is a global registration method which computes the 3-degree-of-freedom (DOF) motion to globally align the source point set to the target point set. The feature of HSM is transforming point samples into Hough domain to generate Hough spectrum. It computes multiple rigid-body transformation candidates which can align the source Hough spectrum to the target Hough spectrum. Since it aligns two point sets globally, HSM was utilized for map merging [65, 66, 67]. Later, HSM was extended into 3D to estimate the 6-DOF motion and abbreviated as HSM3D [68]. However, if we naively implement the algorithm according to the theory, it consumes a lot of resources.

1.3.2 Incremental Registration for Odometry Estimation

The incremental registration was proposed in [69]. It can be regarded as a scan-to-map approach that the aligned scans are integrated on the map, and the source scan is registered with the map.

The scan-to-map approach for NDT was originally proposed in [70]. The authors proposed the simultaneous mapping and tracking method based on NDT-D2D and NDT occupancy map (NDT-OM) to deal with the dynamic objects. The method extracts the local map from the global map and estimates the pose by registering the source NDT to the local map. Also, the source point set is transformed and integrated into the global map. Although the authors considered the discreteness problem of NDT for updating NDT-OM, they did not consider the problem for extracting the local map. Due to the discreteness of the NDT and the one-to-one correspondence strategy, the ground truth transformation is not guaranteed to be the optimal point.

Lidar odometry and mapping (LOAM) is also based on the incremental registration [20]. One thread of LOAM extracts the plane and edge points and registers the feature sets to estimate the pose in high frequency roughly. Another thread of LOAM maintains a global map and registers the source plane and edge points to the plane converted from the plane and edge points in the map respectively in low frequency. LOAM and its improved method visual-lidar odometry and mapping (V-LOAM) [71] rank at the top two places of KITTI chart.

1.4 Contributions

The main contribution of this dissertation is in how it improves NDT registration. To this end, we present improved methods.

The manner in which regular cells which spatially subdivide a point set are dealt with is critical when using the NDT. This dissertation tackles issues pertaining to cells and presents the solutions described below.

In Chapter 3, we present a probabilistic NDT (PNDT) representation. To avoid the destruction of the NDT scene and the degeneration of GCs caused by a high-resolution lattice or sparse point samples, the representation assigns an uncertainty value to each point sample to convert all of the point samples into NDT. PNDT also leads to more accurate registration than the conventional NDT method.

In Chapter 4, we present a lattice adjustment and a cell insertion to avoid the discreteness of GCs. Lattice adjustment generates overlapped cells by decreasing the distance between cells or increasing the side lengths of cells. On the other hand, cell insertion inserts cells to generate simple, face-centered-cubic, or body-centered-cubic structured lattice. In addition, we demonstrate how the overlapped GCs lead to more accurate registration than the conventional NDT.

In Chapter 5, we present the regeneration of NDT to avoid the distortion of GCs after the simple fusion of NDTs. The method subdivides the source GCs into truncated GCs using a target lattice and fuses the truncated GCs in the same cell. Thus, it regenerates the NDT so that it fits the target lattice and reduces the distortion of the fused NDT. Using this method, a robot can build a more accurate NDT map after the robot poses are updated by simultaneous localization and mapping (SLAM). Moreover, the fusion of NDT maps can be more accurate than the simple fusion method.

This dissertation also presents improved NDT registration methods, as follows.

In Chapter 6, we present a hue-assisted NDT registration approach. The hue mostly retains its value even if the brightness of the environment changes. This method additionally subdivides points in a cell according to their hue values and generates multiple

GCs in the cell. Each GC has a distribution of hues, and the presented method improves the NDT registration by weighting the likelihood of correspondence. This facilitates the NDT registration of point sets which are scanned in non-structured environments.

In Chapter 7, we present a key-layered NDT registration (KL-NDT) scheme. Compared to the conventional heuristic multi-layered registration technique, the presented method determines a key layer to register and conducts registration until the likelihood value is converged. Thus, it skips other layers and accelerates the registration process.

In Chapter 8, we present a scaled NDT representation and a multi-scale NDT registration. The overall method decreases the scale of the target NDT from the maximum value to the minimum value to smooth the objective function, and it increases the scale of the source NDT from zero to overcome the negative correlation between the likelihood and the rotational alignment.

In Chapter 9, we present the scan-to-map incremental registration of NDTs. The accuracy of odometry estimation is improved by the following process. First, the source point set is transformed by the initial guess to be converted into an NDT. Second, the submap is extracted according to the robot pose and the source NDT. Third, the source NDT is registered to the target submap in a coarse-to-fine manner. We show that the presented incremental NDT registration outperforms the state-of-the-art odometry estimation method, LIDAR odometry and mapping (LOAM), with the KITTI benchmark dataset.

The issues of NDT and presented methods are summarized in Table 1.1.

Table 1.1: Issues of NDT registration and contributions of the dissertation.

issues of NDT registration	presented method	chapter
degeneration	probabilistic NDT	3
destruction		
interpolation of NDT	lattice adjustment	4
	cell insertion	
	scaled NDT	8
negative correlation between rotation and L_2 likelihood	dynamic scaling factors	
distortion of simple fusion	regeneration of NDT	5
vision-aided registration	hue-assisted NDT	6
heuristic multi-layered NDT	key-layered NDT	7
inefficient scan-to-map registration	incremental NDT registration	9

1.5 Organization

This dissertation is divided into ten chapters. In Chapter 2, we introduce the mathematical preliminaries of NDT representation, NDT registration, and NDT map. We also introduce the implementation of the presented methods and the benchmark dataset used in this dissertation. In Chapter 3-5, the improved NDT representations are presented. Following these, in Chapter 6-9, the improved NDT registration methods are presented. Finally, Chapter 10 draws conclusions and presents topics for future work. The chapters are summarized as follows:

Chapter 1:	background, problem statement, literature review, contributions, and organization.
Chapter 2:	preliminaries of NDT representation, registration, mapping, cell, lattice, and optimization.
Chapter 3:	PNDT using the sensor uncertainty.
Chapter 4:	lattice adjustment and insertion methods.
Chapter 5:	regeneration of NDT for target lattice.
Chapter 6:	registration improved by the hue.
Chapter 7:	KL-NDT determining the key-layer to register.
Chapter 8:	scaled NDT and multi-scale registration.
Chapter 9:	scan-to-map incremental registration.
Chapter 10:	conclusions.

Chapter 2

Preliminaries

2.1 NDT Representation

NDT representation is a special case of Gaussian mixture model (GMM) [45]. Instead of performing expectation-maximization (EM) algorithm, NDT method sets regular cells to rapidly subdivide point samples and computes the parameters of probability density function (pdf) in a cell by maximum likelihood estimation (MLE) [41, 58]. For the set of point samples $\mathcal{P} = \{\mathbf{p}_i = (x_i, y_i, z_i)\}_{i=1}^{n_\mathcal{P}}$ in a cell, it can be assumed that the point samples are driven from a normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Thus, the likelihood of $\mathbf{p}$ can be expressed as

$$\mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\Sigma}; \mathbf{p}) = \frac{1}{\sqrt{|2\pi\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{p}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{p}-\boldsymbol{\mu})\right). \tag{2.1}$$

and the likelihood function of $\mathcal{P}$ is

$$\mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\Sigma}; \mathcal{P}) = \prod_{i=1}^{n_\mathcal{P}} \mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\Sigma}; \mathbf{p}_i). \tag{2.2}$$

By MLE, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ can be obtained as follows:

$$\boldsymbol{\mu} = \frac{1}{n_\mathcal{P}} \sum_{i=1}^{n_\mathcal{P}} \mathbf{p}_i, \tag{2.3}$$

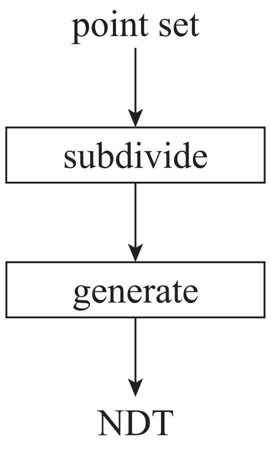


Figure 2.1: Process of converting point set into NDT.

$$\mathbf{\Sigma} = \frac{1}{n_{\mathcal{P}} - 1} \sum_{i=1}^{n_{\mathcal{P}}} (\mathbf{p}_i - \boldsymbol{\mu})(\mathbf{p}_i - \boldsymbol{\mu})^T. \tag{2.4}$$

This process is performed in $n_{\mathcal{D}}$ cells which have four or more point samples, and the NDT model $\mathcal{D} = \{\mathcal{N}(\boldsymbol{\mu}_j, \mathbf{\Sigma}_j)\}_{j=1}^{n_{\mathcal{D}}}$ can be obtained. Given this NDT model as the reference model, the likelihood of observing a point $\mathbf{p}$ generated from the model is a weighted sum of distributions:

$$\mathcal{L}(\mathbf{p}|\mathcal{D}) = \sum_{j=1}^{n_{\mathcal{D}}} w_j \mathcal{N}(\mathbf{p}|\boldsymbol{\mu}_j, \mathbf{\Sigma}_j), \tag{2.5}$$

where w_j is the weight of the likelihood corresponding to $\mathcal{N}(\boldsymbol{\mu}_j, \mathbf{\Sigma}_j)$ [58].

The process of generating NDT can be summarized as Fig. 2.1. After subdividing point samples with regular cells in *subdivide*, the statistical parameters $\boldsymbol{\mu}$ and $\mathbf{\Sigma}$ are computed in *generate*.

2.2 NDT Registration

In the history of NDT registration, the first wave was point-to-distribution registration based on maximum likelihood estimate (MLE) [41, 58], and the second wave was distribution-to-distribution registration based on minimizing L_2 distance between pdf [45]. Although two algorithms are based on the NDT representation, the objective functions are derived from different conceptions.

Assume that the scenes described by a target point set $\mathcal{P}_t$, taken at the state $\mathbf{T}_{H,t}$, and source point set $\mathcal{P}_s$, taken at the state $\mathbf{T}_{H,s}$, are partially overlapped, NDT-P2D converts $\mathcal{P}_t$ into NDT $\mathcal{D}_t$ and finds the optimal $\mathbf{\Theta}$ by minimizing the objective function:

$$f_{P2D}(\mathcal{P}_t, \mathcal{D}_s, \mathbf{\Theta}) = -\sum_{k=1}^{|\mathcal{P}_s|} r_1 \exp\left(-\frac{r_2}{2} g_{P2D}(\mathbf{p}_k, d_{corr,k}, \mathbf{\Theta})\right), \tag{2.6}$$

$$g_{P2D}(\mathbf{p}_k, d_{corr,k}, \mathbf{\Theta}) = \left(T(\mathbf{p}_k, \mathbf{\Theta}) - \boldsymbol{\mu}_{c,k}\right)^T \mathbf{\Sigma}_{c,k}^{-1} \left(T(\mathbf{p}_k, \mathbf{\Theta}) - \boldsymbol{\mu}_{c,k}\right), \tag{2.7}$$

where $\mathcal{N}(\boldsymbol{\mu}_{c,k}, \mathbf{\Sigma}_{c,k})$ is the corresponding distribution to the kth transformed point $T(\mathbf{p}_k, \mathbf{\Theta})$, r_1 and r_2 are the regularizing factors, respectively. According to [58], (2.6) is an approximation of the following negative log-likelihood given the NDT $\mathcal{D}_t$:

$$l(\mathcal{P}_s|\mathcal{D}_t) = -\log \prod_{k=1}^{|\mathcal{P}_s|} \mathcal{L}(T(\mathbf{p}_k, \mathbf{\Theta})|\mathcal{D}_t). \tag{2.8}$$

Since finding $\mathbf{\Theta}^*$ minimizing (2.6) can be seen as finding $\mathbf{\Theta}$ maximizing the likelihood, NDT-P2D is originally in the context of maximum likelihood estimation (MLE).

The process can be depicted as Fig. 2.2. The target point set is converted into NDT by *NDT converter*, and source point set is filtered by *data filter*. The robot pose is estimated by *metric minimize*, and the estimated pose is applied in *transform* to transform the source point set. The iteration is terminated if the pose is converged.

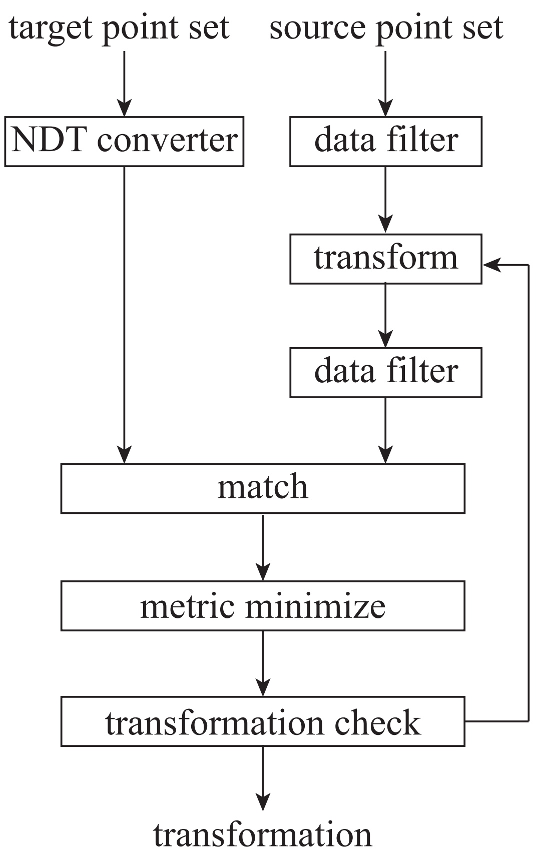


Figure 2.2: Process of NDT-P2D.

On the other hand, NDT-D2D computes $\mathbf{\Theta}$ which minimizes L_2 distance between the target NDT $\mathcal{D}_t$ and source NDT $\mathcal{D}_s$. In [45], the L_2 distance is defined as

$$D_{L_2}(\mathcal{P}_t, \mathcal{D}_s, \mathbf{\Theta}) = \int (p(\mathbf{x}|\mathcal{D}_t) - p(\mathbf{x}|T(\mathcal{D}_s, \mathbf{\Theta}))^2 d\mathbf{x}, \tag{2.9}$$

where $T(\mathcal{D}_s, \mathbf{\Theta})$ transforms $\mathcal{D}_s$ with $\mathbf{\Theta}$ as follows:

$$\boldsymbol{\mu}_{ij} = T(\boldsymbol{\mu}_i, \mathbf{\Theta}) - \boldsymbol{\mu}_j, \tag{2.10}$$

$$\mathbf{\Sigma}_{ij} = T(\mathbf{\Sigma}_i, \mathbf{\Theta}) + \mathbf{\Sigma}_j, \tag{2.11}$$

where $T(\boldsymbol{\Sigma}_i, \boldsymbol{\Theta})$ transforms $\boldsymbol{\Sigma}_i$ with $\boldsymbol{\Theta}$. (2.9) can be simplified as

$$D_{L_2}(\mathcal{P}_t, \mathcal{P}_s, \boldsymbol{\Theta}) \sim \sum_{i=1}^{|\mathcal{D}_s|} \sum_{j=1}^{|\mathcal{D}_t|} \mathcal{N}(0|\boldsymbol{\mu}_{ij}, \boldsymbol{\Sigma}_{ij}). \tag{2.12}$$

Consequently, the objective function of NDT-D2D is defined as

$$f_{D2D}(\mathcal{D}_t, \mathcal{D}_s, \boldsymbol{\Theta}) = -\sum_{i=1}^{|\mathcal{D}_s|} \sum_{j=1}^{|\mathcal{D}_t|} r_1 \exp\left(-\frac{r_2}{2} g_{D2D}(d_i, d_j, \boldsymbol{\Theta})\right), \tag{2.13}$$

$$g_{D2D}(d_i, d_j, \boldsymbol{\Theta}) = \boldsymbol{\mu}_{ij}^T \boldsymbol{\Sigma}_{ij}^{-1} \boldsymbol{\mu}_{ij}, \tag{2.14}$$

where r_1 and r_2 are regularizing factors similar to those in (2.6).

The process of NDT-D2D can be depicted as Fig. 2.3. The target point set and source point set are converted into NDT by *NDT converter*. The robot pose is estimated by *metric minimize*, and the estimated pose is applied in *transform* to transform the source NDT. The iteration is terminated if the pose is converged.

Since NDT-P2D compresses $\mathcal{P}_t$ into $\mathcal{D}_t$, whose number of distributions is significantly smaller than n_t, the time complexity of NDT-P2D is lower than ICP variants. Also, in the same context, since NDT-D2D additionally compresses $\mathcal{P}_s$ into $\mathcal{D}_s$, where $|\mathcal{D}_s|$ is much smaller than $|\mathcal{P}_s|$, the computation complexity of NDT-D2D is much lower than NDT-P2D.

2.3 NDT Mapping

NDT mapping is one of applications using NDT. The NDT occupancy map (NDT-OM) [75], a typical NDT mapping method, formalized the processes of NDT mapping. Later, generic NDT mapping demonstrated the possibility of applying NDT to lifelong graph SLAM [76]. Since the NDT can be built in the multiple resolutions [42, 58, 60],

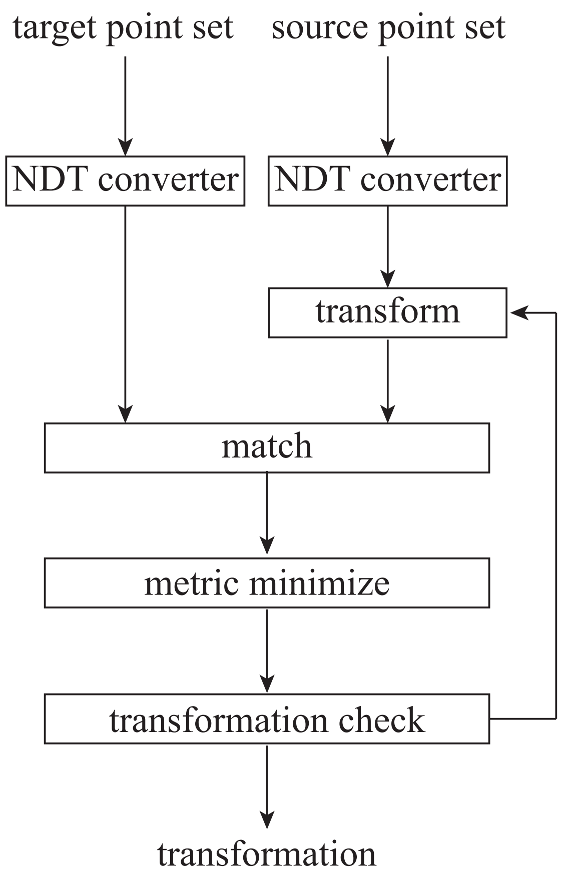


Figure 2.3: Process of NDT-D2D.

NDT-OM and generic NDT mapping also presented the methods of multi-resolution NDT mapping [75, 76]. So far, the hierarchical GMM [74] and the Gaussian mixture map whose cell holds a one-dimensional GMM [77] were proposed. On the other hand, an interest descriptor for robust NDT map matching (IRON) was proposed to estimate the map transformation between a pair of NDT maps for map merging [78].

NDT occupancy map (NDT-OM) is a map represented in the form of an NDT model [75]. In [75], the cell c_j is defined as

$$c_j = \{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j, n_j, p(m_j|\mathbf{x}_{1:t})\}, \tag{2.15}$$

where n_j is the number of points in c_j, and $p(m_j|\mathbf{x}_{1:t})$ is the probability of c_j being occupied [75]. Recursive sample covariance (RSC) update of NDT-OM is performed by following equations. Given an aligned NDT model D_t, c_j can be updated by

$$n_{1:t} = n_{1:t-1} + n_t, \tag{2.16}$$

$$\boldsymbol{\mu}_{1:t} = \mathbf{T}_{1:t}/n_{1:t}, \tag{2.17}$$

$$\boldsymbol{\Sigma}_{1:t} = \mathbf{S}_{1:t}/(n_{1:t} - 1), \tag{2.18}$$

where $\mathbf{T}$ and $\mathbf{S}$ are computed as

$$\mathbf{T} = \sum_{k=1}^{n} \mathbf{x}_k, \tag{2.19}$$

$$\mathbf{S} = \sum_{k=1}^{n} (\mathbf{x}_k - \boldsymbol{\mu})(\mathbf{x}_k - \boldsymbol{\mu})^T, \tag{2.20}$$

and $\mathbf{T}_{1:t}$ and $\mathbf{S}_{1:t}$ are updated by

$$\mathbf{T}_{1:t} = \mathbf{T}_{1:t-1} + \mathbf{T}_t, \tag{2.21}$$

$$\mathbf{S}_{1:t} = \mathbf{S}_{1:t-1} + \mathbf{S}_t + \frac{n_{1:t-1} n_t}{n_{1:t}} \left(\boldsymbol{\mu}_{1:t-1} - \boldsymbol{\mu}_t\right) \left(\boldsymbol{\mu}_{1:t-1} - \boldsymbol{\mu}_t\right)^T. \tag{2.22}$$

2.4 Transformation Matrix and The Parameter Vector

The robot pose can be expressed as a homogeneous transformation matrix:

$$\mathbf{T}_H = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix}, \tag{2.23}$$

where $\mathbf{R}$ is the rotation matrix, $\mathbf{t} = (t_x, t_y, t_z)^T$ is the translation vector. In this dissertation, Tait-Bryan angle vector $\mathbf{r} = (r_x, r_y, r_z)^T$ determines $\mathbf{R}$, and the parameter vector is deinfed as $\mathbf{\Theta} = (t_x, t_y, t_z, r_x, r_y, r_z)^T \in \mathbb{R}^6$. Also, we define that $T(\mathbf{v}, \mathbf{\Theta})$ transforms the vector $\mathbf{v}$ with $\mathbf{\Theta}$ as follows:

$$T(\mathbf{v}, \mathbf{\Theta}) = \mathbf{R}\mathbf{v} + \mathbf{t}, \tag{2.24}$$

and $T(\mathbf{M}, \mathbf{\Theta})$ transforms the matrix $\mathbf{M}$ with $\mathbf{\Theta}$ as follows:

$$T(\mathbf{M}, \mathbf{\Theta}) = \mathbf{R}\mathbf{M}\mathbf{R}^T. \tag{2.25}$$

2.5 Cubic Cell and Lattice

In this dissertation, a regular cell v is defined as a unit volume for the 3D case, and we describe v using the side length l. A lattice $\mathcal{V} = \{v_i\}_{i=1}^n$ is defined as a repetitive arrangement of the cells. To perform the coarse-to-fine NDT registration, the NDTs generated in multiple resolutions are required. To manage the multi-resolution NDTs efficiently, we choose octree data structure to implement the NDT. Octree is a hierarchical data structure for the spatial subdivision in 3D [79]. It sets a cube covering the current interest region and recursively subdivides the cube into 8 son-cubes, as shown in Fig. 2.4. Here, the layer is used to distinguish the NDT generated by cubes in each level of octree. The side length l_0 in the layer 0 can be obtained by

$$l_0 = 2^{L_f} l_f, \tag{2.26}$$

where L_f is the fine layer and l_f is the length of the cells in the layer L_f. Also, the length l_k of cells in the kth layer can be computed as

$$l_k = 2^{-k} l_0. \tag{2.27}$$

2.6 Optimization

In this dissertation, we minimize the objective function using Newton method. Given an objective function $f(\mathbf{\Theta})$, we compute the gradient $\nabla f(\mathbf{\Theta})$ and Hessian $\mathbf{H}f(\mathbf{\Theta})$ to update $\mathbf{\Theta}$ as follows:

$$\mathbf{\Theta}_{k+1} = \mathbf{\Theta}_k + \Delta\mathbf{\Theta}, \tag{2.28}$$

$$\Delta\mathbf{\Theta} = -\gamma \left(\mathbf{H}f(\mathbf{\Theta})\right)^{-1} \nabla f(\mathbf{\Theta}), \tag{2.29}$$

where γ is the step size which satisfies Armijo rule:

$$f(\mathbf{\Theta}_{k+1}) \leq f(\mathbf{\Theta}_k) + c_1 \Delta\mathbf{\Theta}^T \nabla f(\mathbf{\Theta}), \tag{2.30}$$

where c_1 is a constant. In this dissertation, we set c_1 to 10^{-4}. The optimization terminates if the step size $|\Delta\mathbf{\Theta}|$ is smaller than threshold $\tau = 10^{-6}$.

For NDT-P2D, the gradient of (2.6) can be derived as

$$\frac{\partial f_{P2D}}{\partial \theta_a}(\mathcal{P}_t, \mathcal{D}_s, \mathbf{\Theta}) = \frac{r_2}{2} \sum_{k=1}^{|\mathcal{P}_s|} r_1 \exp\left(-\frac{r_2}{2} g_{P2D}(\mathbf{p}_k, d_{corr,k}, \mathbf{\Theta})\right) \frac{\partial g_{P2D}}{\partial \theta_a}(\mathbf{p}_k, d_{corr,k}, \mathbf{\Theta}), \tag{2.31}$$

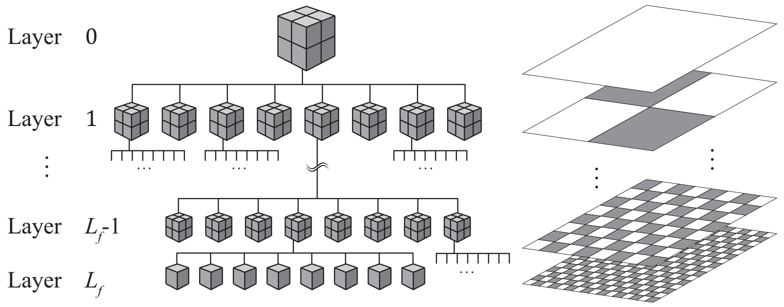


Figure 2.4: Illustration of the octree structure.

$$\frac{\partial g_{P2D}}{\partial \theta_a}(\mathbf{p}_k, d_{corr,k}, \mathbf{\Theta}) = 2\left(T(\mathbf{p}_k, \mathbf{\Theta}) - \boldsymbol{\mu}_{c,k}\right)^T \mathbf{\Sigma}_{c,k}^{-1} \frac{\partial T}{\partial \theta_a}(\mathbf{p}_k, \mathbf{\Theta}), \tag{2.32}$$

and the Hessian of (2.6) can be derived as

$$\frac{\partial^2 f_{P2D}}{\partial \theta_a \partial \theta_b}(\mathcal{P}_t, \mathcal{D}_s, \mathbf{\Theta}) = \frac{r_2}{2}\sum_{k=1}^{|\mathcal{P}_s|} r_1 \exp\left(-\frac{r_2}{2} g_{P2D}(\mathbf{p}_k, d_{corr,k}, \mathbf{\Theta})\right) \cdot \left(\frac{\partial^2 g_{P2D}}{\partial \theta_a \partial \theta_b}(\mathbf{p}_k, d_{corr,k}, \mathbf{\Theta}) - \frac{r_2}{2}\frac{\partial g_{P2D}}{\partial \theta_a}(\mathbf{p}_k, d_{corr,k}, \mathbf{\Theta})\frac{\partial g_{P2D}}{\partial \theta_b}(\mathbf{p}_k, d_{corr,k}, \mathbf{\Theta})\right), \tag{2.33}$$

$$\frac{\partial^2 g_{P2D}}{\partial \theta_a \partial \theta_b}(\mathbf{p}_k, d_{corr,k}, \mathbf{\Theta}) = 2\left(T(\mathbf{p}_k, \mathbf{\Theta}) - \boldsymbol{\mu}_{c,k}\right)^T \mathbf{\Sigma}_{c,k}^{-1} \frac{\partial^2 T}{\partial \theta_a \partial \theta_b}(\mathbf{p}_k, \mathbf{\Theta}) + 2\left(\frac{\partial T}{\partial \theta_a}(\mathbf{p}_k, \mathbf{\Theta})\right)^T \mathbf{\Sigma}_{c,k}^{-1} \frac{\partial T}{\partial \theta_b}(\mathbf{p}_k, \mathbf{\Theta}). \tag{2.34}$$

For NDT-D2D, the gradient of (2.13) can be derived as

$$\frac{\partial f_{D2D}}{\partial \theta_a}(\mathcal{D}_t, \mathcal{D}_s, \mathbf{\Theta}) = \frac{r_2}{2}\sum_{i=1}^{|\mathcal{D}_s|}\sum_{j=1}^{|\mathcal{D}_t|} r_1 \exp\left(-\frac{r_2}{2} g_{D2D}(d_i, d_j, \mathbf{\Theta})\right)\frac{\partial g_{D2D}}{\partial \theta_a}(d_i, d_j, \mathbf{\Theta}), \tag{2.35}$$

$$\frac{\partial g_{D2D}}{\partial \theta_a}(d_i, d_j, \mathbf{\Theta}) = 2\boldsymbol{\mu}_{ij}^T \mathbf{\Sigma}_{ij}^{-1} \frac{\partial \boldsymbol{\mu}_{ij}}{\partial \theta_a} + \boldsymbol{\mu}_{ij}^T \frac{\partial \mathbf{\Sigma}_{ij}^{-1}}{\partial \theta_a}\boldsymbol{\mu}_{ij}, \tag{2.36}$$

$$\frac{\partial \boldsymbol{\mu}_{ij}}{\partial \theta_a} = \frac{\partial T}{\partial \theta_a}(\boldsymbol{\mu}_i, \mathbf{\Theta}), \tag{2.37}$$

$$\frac{\partial \mathbf{\Sigma}_{ij}^{-1}}{\partial \theta_a} = -\mathbf{\Sigma}_{ij}^{-1}\frac{\partial T}{\partial \theta_a}(\mathbf{\Sigma}_i, \mathbf{\Theta})\mathbf{\Sigma}_{ij}^{-1}, \tag{2.38}$$

and the Hessian of (2.13) can be derived as

$$\frac{\partial^2 f_{D2D}}{\partial \theta_a \partial \theta_b}(\mathcal{D}_t, \mathcal{D}_s, \mathbf{\Theta}) = \frac{r_2}{2}\sum_{i=1}^{|\mathcal{D}_s|}\sum_{j=1}^{|\mathcal{D}_t|} r_1 \exp\left(-\frac{r_2}{2} g_{D2D}(d_i, d_j, \mathbf{\Theta})\right) \cdot \left(\frac{\partial^2 g_{D2D}}{\partial \theta_a \partial \theta_b}(d_i, d_j, \mathbf{\Theta}) - \frac{r_2}{2}\frac{\partial g_{D2D}}{\partial \theta_a}(d_i, d_j, \mathbf{\Theta})\frac{\partial g_{D2D}}{\partial \theta_b}(d_i, d_j, \mathbf{\Theta})\right), \tag{2.39}$$

$$\frac{\partial^2 g_{D2D}}{\partial\theta_a\partial\theta_b}(d_i,d_j,\boldsymbol{\Theta}) = 2\boldsymbol{\mu}_{ij}^T\boldsymbol{\Sigma}_{ij}^{-1}\frac{\partial^2\boldsymbol{\mu}_{ij}}{\partial\theta_a\partial\theta_b} + 2\boldsymbol{\mu}_{ij}^T\frac{\partial\boldsymbol{\Sigma}_{ij}^{-1}}{\partial\theta_b}\frac{\partial\boldsymbol{\mu}_{ij}}{\partial\theta_a} + 2\boldsymbol{\mu}_{ij}^T\frac{\partial\boldsymbol{\Sigma}_{ij}^{-1}}{\partial\theta_a}\frac{\partial\boldsymbol{\mu}_{ij}}{\partial\theta_b} + 2\left(\frac{\partial\boldsymbol{\mu}_{ij}}{\partial\theta_a}\right)^T\boldsymbol{\Sigma}_{ij}^{-1}\frac{\partial\boldsymbol{\mu}_{ij}}{\partial\theta_b} + \boldsymbol{\mu}_{ij}^T\frac{\partial^2\boldsymbol{\Sigma}_{ij}^{-1}}{\partial\theta_a\partial\theta_b}\boldsymbol{\mu}_{ij}, \quad (2.40)$$

$$\frac{\partial^2\boldsymbol{\mu}_{ij}}{\partial\theta_a\partial\theta_b} = \frac{\partial^2 T}{\partial\theta_a\partial\theta_b}(\boldsymbol{\mu}_i,\boldsymbol{\Theta}), \quad (2.41)$$

$$\frac{\partial^2\boldsymbol{\Sigma}_{ij}^{-1}}{\partial\theta_a\partial\theta_b} = -\boldsymbol{\Sigma}_{ij}^{-1}\frac{\partial^2 T}{\partial\theta_a\partial\theta_b}(\boldsymbol{\Sigma}_i,\boldsymbol{\Theta})\boldsymbol{\Sigma}_{ij}^{-1} + \boldsymbol{\Sigma}_{ij}^{-1}\frac{\partial T}{\partial\theta_a}(\boldsymbol{\Sigma}_i,\boldsymbol{\Theta})\left(\boldsymbol{\Sigma}_{ij}^{-1}\frac{\partial T}{\partial\theta_b}(\boldsymbol{\Sigma}_i,\boldsymbol{\Theta})\boldsymbol{\Sigma}_{ij}^{-1}\right) + \boldsymbol{\Sigma}_{ij}^{-1}\frac{\partial T}{\partial\theta_b}(\boldsymbol{\Sigma}_i,\boldsymbol{\Theta})\boldsymbol{\Sigma}_{ij}^{-1}\frac{\partial T}{\partial\theta_a}(\boldsymbol{\Sigma}_i,\boldsymbol{\Theta})\boldsymbol{\Sigma}_{ij}^{-1}. \quad (2.42)$$

2.7 Implementation

The presented methods are implemented in C/C++ language with Eigen library for linear algebra [80], Boost library [81] for multi-thread processing, and OpenGL and FreeGLUT [82] for visualization. The implementations are processed on Intel i7 7700 3.60GHz.

2.8 Evaluation of Registration

The presented methods in this dissertation evaluates the accuracy of the registration to show the improvement. For the estimated transformation $\hat{\mathbf{T}}_H$ and the ground truth transformation $\mathbf{T}_H$, we compute the error matrix $\mathbf{E}_k$ as follows:

$$\mathbf{E}_k = \hat{\mathbf{T}}_H^{-1}\mathbf{T}_H = \begin{bmatrix} \mathbf{E}_R & \mathbf{E}_t \\ 0 & 1 \end{bmatrix}. \quad (2.43)$$

Next, the translation and rotation errors $e_{t,k}$ and $e_{r,k}$ can be obtained by $e_t = \|\mathbf{E}_t\|_2$ and $e_r = \angle[\mathbf{E}_R]$, where $\angle[\cdot]$ is the function which computes the rotation angle θ as

$$\theta = \arccos\left(\frac{Tr(\mathbf{R})-1}{2}\right). \quad (2.44)$$

Table 2.1: Configuration of KITTI benchmark data set [20].

seq. no.	configuration	
	distance(m)	environment
0	3714	urban
1	4268	highway
2	5075	urban+country
3	563	country
4	397	country
5	2223	urban
6	1239	urban
7	695	urban
8	3225	urban+country
9	1717	urban+country
10	919	urban+country

2.9 Benchmark Dataset

This dissertation uses KITTI benchmark data set collected by 64 channel lidar Velodyne mounted on a car since it provides the reliable ground truths to evaluate odometry estimated by the point set registration [83]. The maximum range was set to 100m, and the number of points per frame is about 0.1M. KITTI benchmark data set is consisted of 11 sequences, as shown in Table 2.1. It provides reliable ground truths, as shown in Fig. 2.5. KITTI uses the criteria which are the averages of relative translational and rotational errors e_t and e_r of (100, 200, 300, ..., 800)m intervals [83]. To be specific,

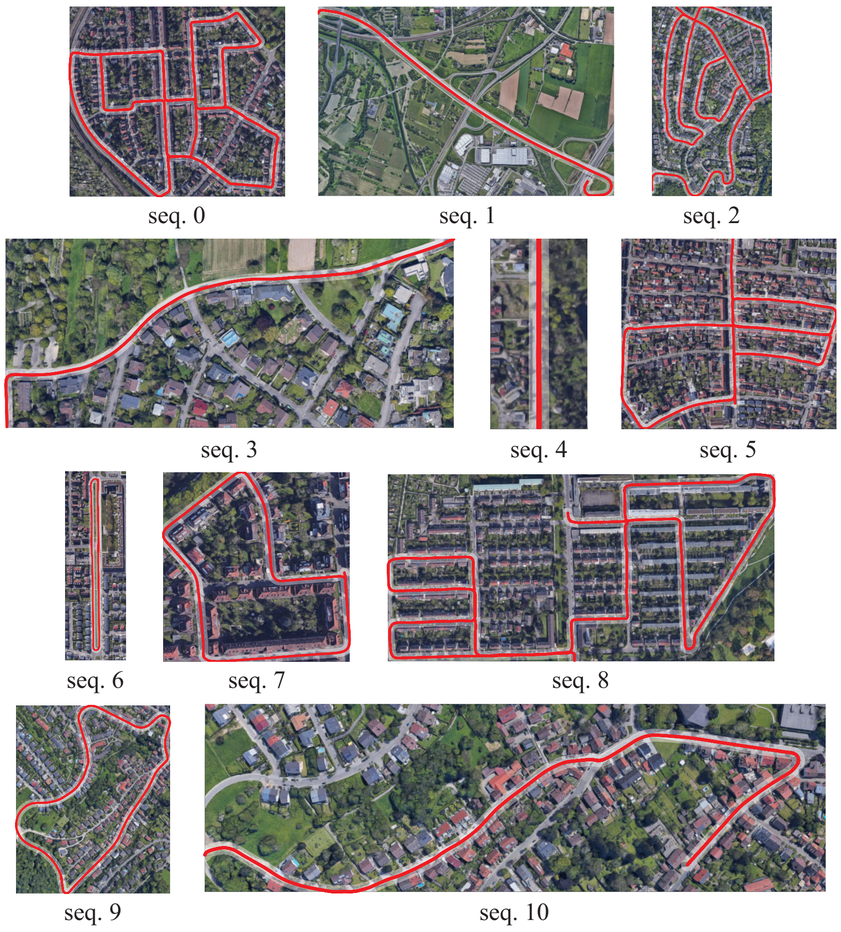


Figure 2.5: Ground truth odometry of KITTI data set on the satellite map.

e_t and e_r are computed as follows:

$$e_t = \frac{|\mathbf{E}_t|}{\Delta\mathbf{t}} \times 100\%, \tag{2.45}$$

$$e_r = \frac{1}{\Delta\mathbf{t}} \arccos\left(\frac{Tr(\mathbf{E}_r) - 1}{2}\right), \tag{2.46}$$

which are different from the errors in Section 2.8.

Chapter 3

Probabilistic NDT Representation

3.1 Introduction

This chapter presents a probabilistic NDT (PNDT) representation to overcome two issues of NDT representation: degeneration and destruction. Degeneration is a problem that the covariance matrix $\mathbf{\Sigma}$ is singular. The conventional method is to adjust the eigenvalues manually. Destruction is a problem that NDT does not convert point samples which are fewer than 4. Due to the destruction, some regions are disappeared in the NDT representation, as shown in Fig. 3.1(b-d). There are two major factors that determine the number of points in a cell: density of point samples and resolution of the lattice. For example, the density of points decreases as the range or incidental angle increases. Also, the number of points in the cell decreases as the cell size decreases. If NDTs are destructed due to the high-resolution cells, the registration of the NDTs may lead to an inaccurate transformation.

PNDT is a representation which considers the sensor uncertainty as the inherent covariance for each point sample and converts all point samples into NDT regardless

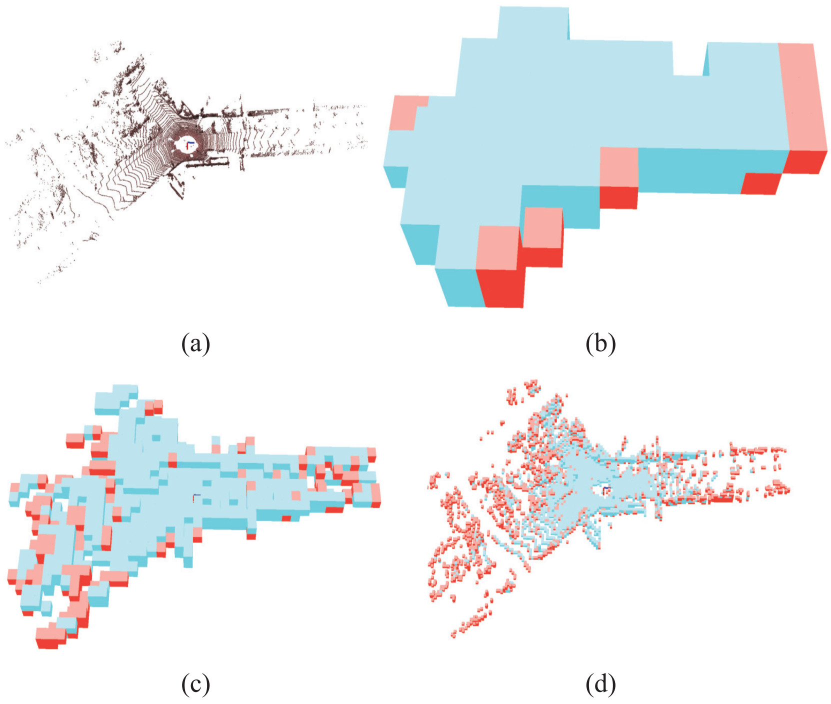


Figure 3.1: Examples of the destructed NDT. From (b) to (d) are the NDTs converted from a point set in (a) using 8m, 2m, and 0.5m cells. The cells which generate the GCs are in blue; otherwise, cells are in red.

of the cell size. The details of the PNDT are presented in the following sections. In Section 3.2, we introduce the sensor uncertainty modeled in the spherical coordinate system. In Section 3.3, we present the PNDT representation and derive the modified covariance. In Section 3.4, we discuss the generalization of NDT registration. In Section 3.5, we compare the generation rate of NDT and PNDT. We also compare the registration accuracy of NDT and PNDT.

3.2 Uncertainty of Point Based on Sensor Model

This dissertation considers a range finder collecting point samples in spherical coordinate system as

$$\hat{\mathbf{p}}_s = (r, \theta, \varphi), \tag{3.1}$$

where r is radius, θ is horizontal angle, and φ is vertical angle, as shown in Fig. 3.2. Also, $\mathbf{\Sigma}_s$, the covariance matrix of $\hat{\mathbf{p}}_s$, can be expressed as

$$\Sigma_s = \begin{bmatrix} \sigma_r^2 & 0 & 0 \\ 0 & \sigma_\theta^2 & 0 \\ 0 & 0 & \sigma_\varphi^2 \end{bmatrix}, \tag{3.2}$$

where σ_r^2, σ_θ^2, and σ_φ^2 are variances of r, θ, and φ, respectively. We use the measurement $\hat{\mathbf{p}}$ and covariance $\mathbf{\Sigma}_p$ in Cartesian coordinate system transformed from $\hat{\mathbf{p}}_s$ and $\mathbf{\Sigma}_s$ to define the pdf of the point as a normal distribution:

$$p\left(\mathbf{p}|\hat{\mathbf{p}}, \mathbf{\Sigma}_p\right) = \mathcal{N}\left(\hat{\mathbf{p}}, \mathbf{\Sigma}_p\right) = \frac{1}{|2\pi\Sigma_p|^{1/2}} \exp\left(-\frac{1}{2}\left(\mathbf{p} - \hat{\mathbf{p}}\right) \mathbf{\Sigma}_p^{-1} \left(\mathbf{p} - \hat{\mathbf{p}}\right)\right). \tag{3.3}$$

The point sample $\hat{\mathbf{p}}$ is transformed from $\hat{\mathbf{p}}_s$ by

$$f_{s2c}(r, \theta, \varphi) = \begin{bmatrix} r \sin\varphi \cos\theta \\ r \cos\varphi \\ r \sin\varphi \sin\theta \end{bmatrix}, \tag{3.4}$$

and $\mathbf{\Sigma}_p$ can be approximately transformed from $\mathbf{\Sigma}_s$ by

$$\mathbf{\Sigma}_p = \mathbf{J}\mathbf{\Sigma}_s\mathbf{J}^T, \tag{3.5}$$

where $\mathbf{J}$ is a Jacobian matrix computed as

$$\mathbf{J} = \frac{\partial f_{s2c}\left(r, \theta, \varphi\right)}{\partial\left(r, \theta, \varphi\right)} = \begin{bmatrix} \sin\varphi\cos\theta & -r\sin\varphi\sin\theta & r\cos\varphi\cos\theta \\ \cos\varphi & 0 & -r\sin\varphi \\ \sin\varphi\sin\theta & r\sin\varphi\cos\theta & r\cos\varphi\sin\theta \end{bmatrix}. \tag{3.6}$$

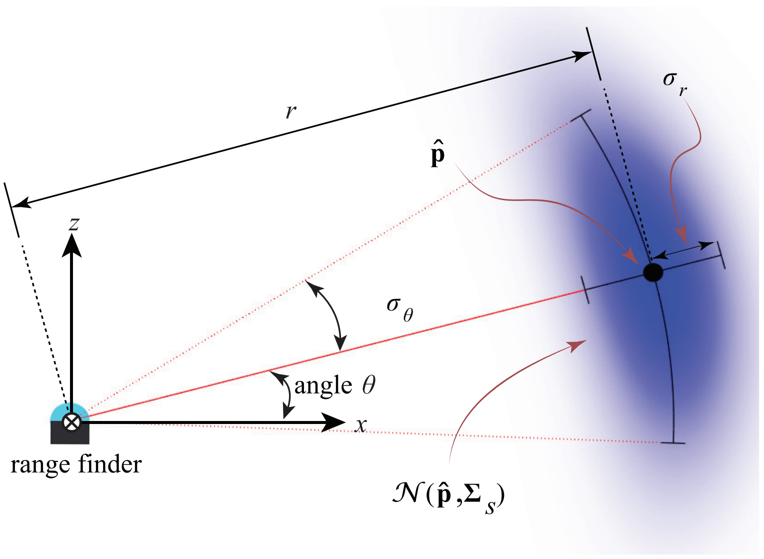


Figure 3.2: Top view of a sensor model in spherical coordinate system and the normal distribution of a point $\mathbf{p}$.

Since the eigenvalues $(\lambda_1, \lambda_2, \lambda_3)$ of $\mathbf{\Sigma}_p$ is equal to $\left(\sigma_r^2, (rsin\varphi\sigma_\theta)^2, (r\sigma_r)^2\right)$, λ_2 and λ_3 increase as r increases. Using the conversion, a point set $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^{n_\mathcal{P}}$ can be converted into an NDT as

$$\mathcal{D}^P = \{(\mathcal{N}(\hat{\mathbf{p}}_i, \Sigma_{U,i}), 1)\}, i = 1, 2, 3, ..., n_\mathcal{P}, \tag{3.7}$$

where $\Sigma_{U,i}, i = 1, 2, 3, ..., n_\mathcal{P}$ is the uncertainty covariance of $\hat{\mathbf{p}}_i$.

3.3 Probabilistic NDT

NDT model is a set of GCs which are converted from a set of point samples. The most critical parameters to generate and update a GC are the mean vector μ, covariance Σ, and the number n of points. Therefore,in this dissertation, to include the parameters,

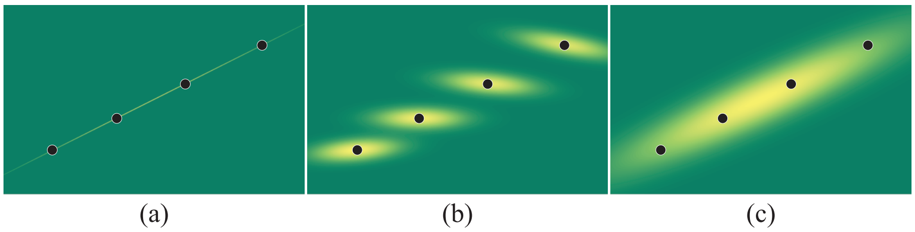


Figure 3.3: Example of PNDT representation. Given four points on a line, the GC of the points is degenerated, as shown in (a). Given the uncertainty of each point, as shown in (b), the GC can avoid degeneration, as shown in (c).

we define a GC d as

$$d = (\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), n). \tag{3.8}$$

Also, we define the NDT model $\mathcal{D} = \{d_i\}_{i=1}^{n_\mathcal{D}}$.

PNDT is the representation considering the uncertainty for the measured point sample. For the point sample set $\mathcal{P} = \{\mathbf{p}_j\}_{j=1}^{n_\mathcal{P}}$, as shown in Fig. 3.3(a), the probability observing a point coordinate $\mathbf{p}$ can be defined as GMM:

$$p(\mathbf{p}) = \frac{1}{n_\mathcal{P}} \sum_{j=1}^{n_\mathcal{P}} \frac{1}{|2\pi \boldsymbol{\Sigma}_{u,j}|} \exp\Big(-(\mathbf{p} - \hat{\mathbf{p}}_j)\boldsymbol{\Sigma}_{u,j}^{-1}(\mathbf{p} - \hat{\mathbf{p}}_j)\Big), \tag{3.9}$$

as in Fig. 3.3(b). The key idea of the PNDT is to use the mean and covariance of $\mathbf{p}$. Given a lattice $\mathcal{V} = \{v_i = (\mathbf{c}_i, \boldsymbol{l}_i)\}_{i=1}^{n_\mathcal{V}}$, where $n_\mathcal{V}$ is the number of cells, the point set in the cell v_i can be represented as $\mathcal{D}_i^P = \{d_j = (\mathcal{N}(\mu_j = \hat{\mathbf{p}}_j, \boldsymbol{\Sigma}_j = \boldsymbol{\Sigma}_{U,j}), n_j = 1)\}_{j=1}^{n_{P,i}}$. The mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ of the PNDT can be computed as follows:

$$\boldsymbol{\mu} = E[\mathbf{p}] = \int \mathbf{p} p(\mathbf{p}) d\mathbf{p}, \tag{3.10}$$

$$\boldsymbol{\Sigma} = Cov[\mathbf{p}] = \int (\mathbf{p} - E[\mathbf{p}])(\mathbf{p} - E[\mathbf{p}])^T p(\mathbf{p}) d\mathbf{p}, \tag{3.11}$$

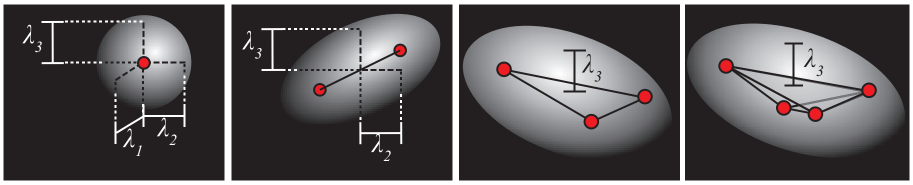


Figure 3.4: Illustration of the degeneration cases and the one-sigma ellipsoids of PNDT: point case in (a) (one point sample), line case in (b) (two point samples), plane case in (c) (three point samples), and tetrahedron case in (d) (four point samples).

and these can be derived as

$$\boldsymbol{\mu} = \frac{1}{n_{\mathcal{P}}} \sum_{j=1}^{n_{\mathcal{P}}} \hat{\mathbf{p}}_j, \tag{3.12}$$

$$\boldsymbol{\Sigma} = \frac{1}{n_{\mathcal{P}}} \sum_{j=1}^{n_{\mathcal{P}}} \left((\hat{\mathbf{p}}_j - E[\mathbf{p}])(\hat{\mathbf{p}}_j - E[\mathbf{p}])^T + \boldsymbol{\Sigma}_{u,j} \right). \tag{3.13}$$

The GC converted from the uncertainty GMM in Fig. 3.3(b) is not degenerated, as shown in Fig. 3.3(c).

As shown in (3.12), $\boldsymbol{\mu}$ is equal to the conventional mean computed by (2.3). It means the center of GC is not changed by the modification. On the other hand, $\boldsymbol{\Sigma}$ is the sum of the conventional covariance and the mean of uncertainties $\{\boldsymbol{\Sigma}_{u,j}\}_{j=1}^{n_{\mathcal{P}}}$. This modification provides two advantages. First, it can provide a covariance for one-, two-, and three-point cases, as shown in Fig. 3.4(a-c). Second, it reduces the degeneration effect of the GC, as shown in Fig 3.4(d). The conventional method adjusts each eigenvalue to $\rho\lambda_1$, where ρ is a threshold ratio, if the eigenvalue is smaller than $\rho\lambda_1$. However, using (3.13), some distributions are expected to ignore the adjustment.

3.4 Generalization of NDT Registration Based on PNDT

This section is to show that NDT-P2D is a special case of NDT-D2D. According to the PNDT representation, we can model a raw point set $\mathcal{P}_s$ as a set of point samples whose uncertainties are equal to

$$\mathbf{\Sigma}_{u,\epsilon} = \begin{bmatrix} \epsilon & 0 & 0 \\ 0 & \epsilon & 0 \\ 0 & 0 & \epsilon \end{bmatrix}, \tag{3.14}$$

where $\epsilon \to 0$. Assume that we set a lattice $\mathcal{V}_s$ whose resolution is so high that each cell has at most one point sample. The PNDT $\mathcal{D}_s$ converted from $\mathcal{P}_s$ using $\mathcal{V}_s$ is a set of GCs whose means μ_i and covariances $\mathbf{\Sigma}_i$ are equal to point coordinates $\mathbf{p}_i$ and the uncertainty $\mathbf{\Sigma}_{u,\epsilon}$, respectively. Consider that NDT-D2D registers $\mathcal{D}_s$ to $\mathcal{D}_t$, we can substitute the mean and covariance into (2.10) and (2.11) and obtain

$$\mu_{ij} = T(\mathbf{p}_i, \mathbf{\Theta}) - \mu_j, \tag{3.15}$$

$$\mathbf{\Sigma}_{ij} = T(\mathbf{\Sigma}_{u,\epsilon}, \mathbf{\Theta}) + \mathbf{\Sigma}_j. \tag{3.16}$$

Given that $\epsilon \to 0$, we can approximate (3.16) and obtain $\mathbf{\Sigma}_{ij} = \mathbf{\Sigma}_j$. Thus, the equation (2.40) can be simplified as

$$g_{D2D}(d_i, d_j, \mathbf{\Theta}) = (T(\mathbf{p}_i, \mathbf{\Theta}) - \mu_j)^T \mathbf{\Sigma}_j^{-1} (T(\mathbf{p}_i, \mathbf{\Theta}) - \mu_j), \tag{3.17}$$

which is the same as (2.7). Next, assume that the likelihood of the source GC d_i and target GC d_j is approximately 0 if they are not in the same cell, the objective function of NDT-D2D as (2.13) becomes the objective function of NDT-P2D as (2.6).

3.5 Experiments

3.5.1 Overview

This chapter used KITTI benchmark data set. Since the maximum range of Velodyne is 120m, the cell size corresponding to the root octree node is set to 128m. To evaluate NDT and PNDT in the same conditions, the cell sizes are set to l, and the ranges searching for correspondence is set to the 1.2 times l. On the other hand, the generation process calculates means and covariances of point sets in cells to transform the point set into NDT. The NDT sets the threshold number n_{th} to reject conversion of points fewer than n_{th}, whereas PNDT does not set n_{th}. The PNDT additionally computes the uncertainties of points using the variances of r, θ, and φ analyzed in [19]. Since mapping is not considered in this chapter, each point set is deleted after the process generates the corresponding NDT.

In this chapter, we apply the initial guess $\hat{\boldsymbol{\Theta}}_0$ equal to the preceding transformation $\hat{\boldsymbol{\Theta}}[k-1]$ to the registration, as shown in Fig. 3.5. After it returns the estimated transformation $\hat{\boldsymbol{\Theta}}$, the target NDT $\mathcal{D}_t$ is deleted, and the source NDT $\mathcal{D}_s$ becomes the target NDT.

3.5.2 Evaluation of Representation

To evaluate the performance of representation, all scans in the 0th dataset were transformed into conventional NDTs and PNDTs by cells in layers from 1 to 12. We collected two data sets: the generation rate and the number of values. The generation rate is the percentage of cells having distributions among the occupied cells, and the number of values is for evaluating the memory usage per NDT. Since a distribution can be reconstructed by nine values, the number is equal to nine times the number of cells

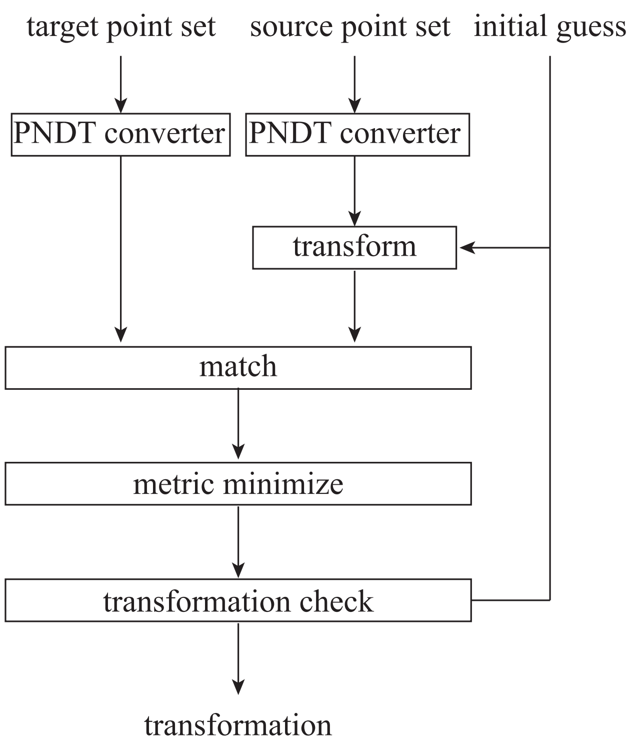
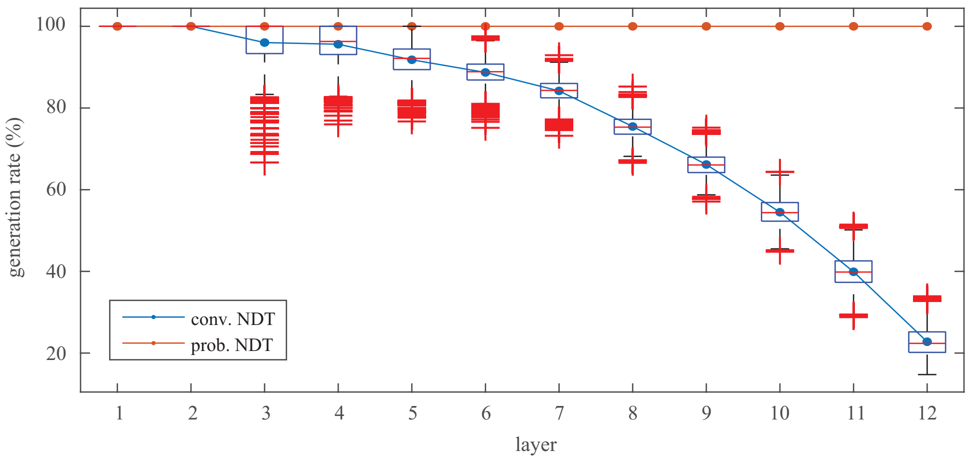


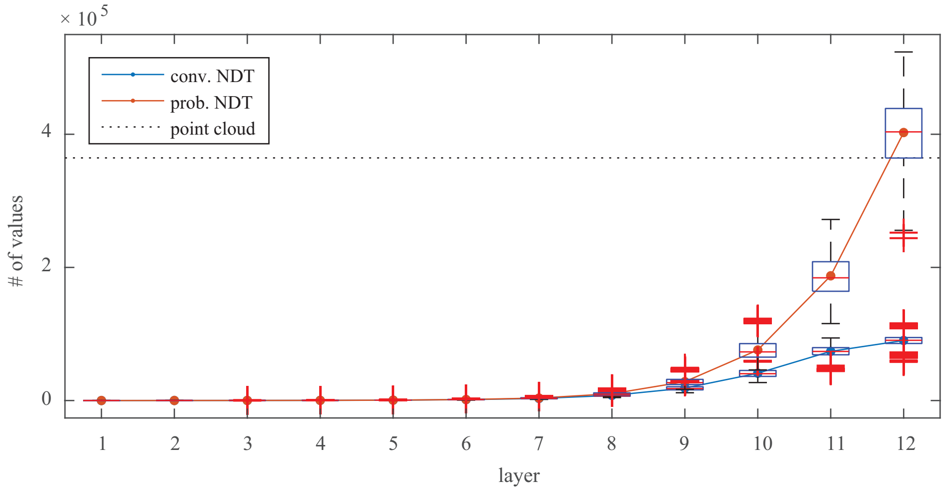
Figure 3.5: Block diagram of PNDT-D2D with an initial guess.

having distributions. We obtained results of these as shown in Fig. 3.6. The box plots against layer in Fig. 3.6 show the statistical results obtained by 4541 scans in the 0th dataset.

The result in Fig. 3.6(a) shows that the generation rate of the conventional NDT decreases with increasing layer, while that of the PNDT does not change with layer and remain 100%. The conventional NDT can represent the environment in the lower layer as shown in Fig. 3.7(a), but the end parts of the scan are disappeared in the higher layer as shown in Fig. 3.7(d). On the other hand, the PNDT generates the distributions by using the parameter $\sum_{i=1}^{n_P} \mathbf{\Sigma}_{U,i}/n_P$ as shown in Fig. 3.7(b) and (e). It can generate distributions in all of the occupied cells without NDT parameters as shown in Fig.



(a)



(b)

Figure 3.6: Evaluation of representation performance. (a) generation rate against layer. (b) the number of values against layer.

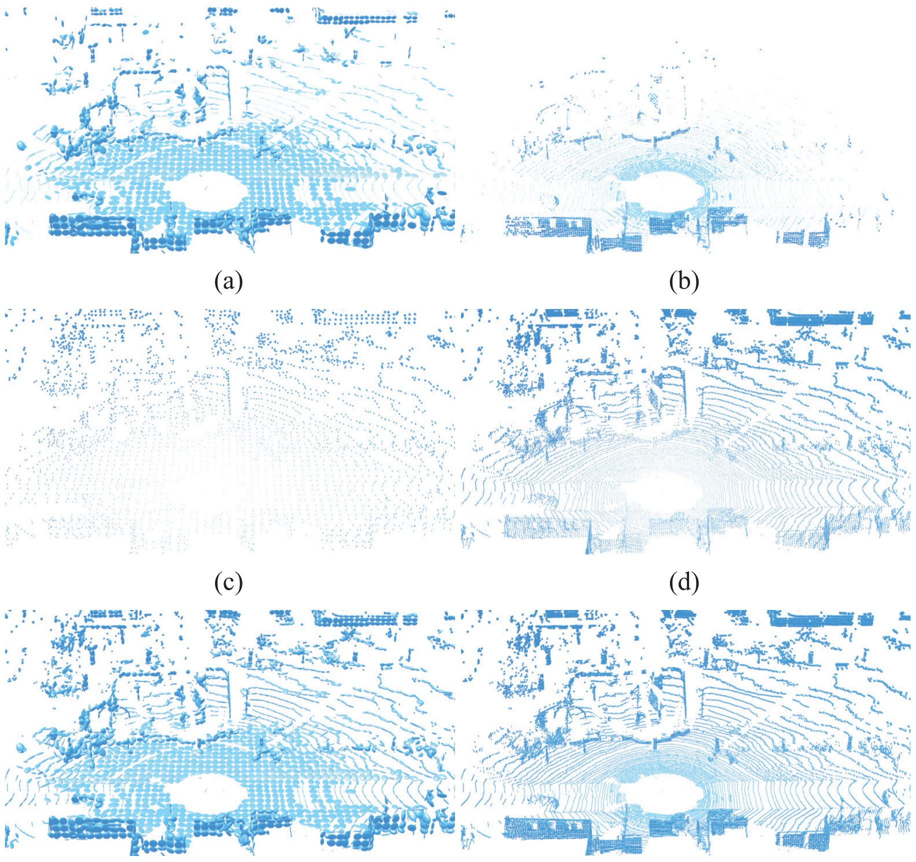

(a) (b) (c) (d) (e) (f)

Figure 3.7: Comparison of $\mathbf{\Sigma}$, $\sum_{i=1}^{n_P} \mathbf{\Sigma}_{u,i}/n_P$, and $\mathbf{\Sigma}_{PNDT}$ in the shape of 1σ Ellipsoid. Left column shows distributions generated by 0.5m cells, and the right column shows distributions generated by 0.125m. (a) and (b) are 1σ ellipsoids of $\mathbf{\Sigma}$, (c) and (d) are 1σ ellipsoids of $\sum_{i=1}^{n_P} \mathbf{\Sigma}_{u,i}/n_P$, and (e) and (f) are 1σ ellipsoids of $\mathbf{\Sigma}_{PNDT}$.

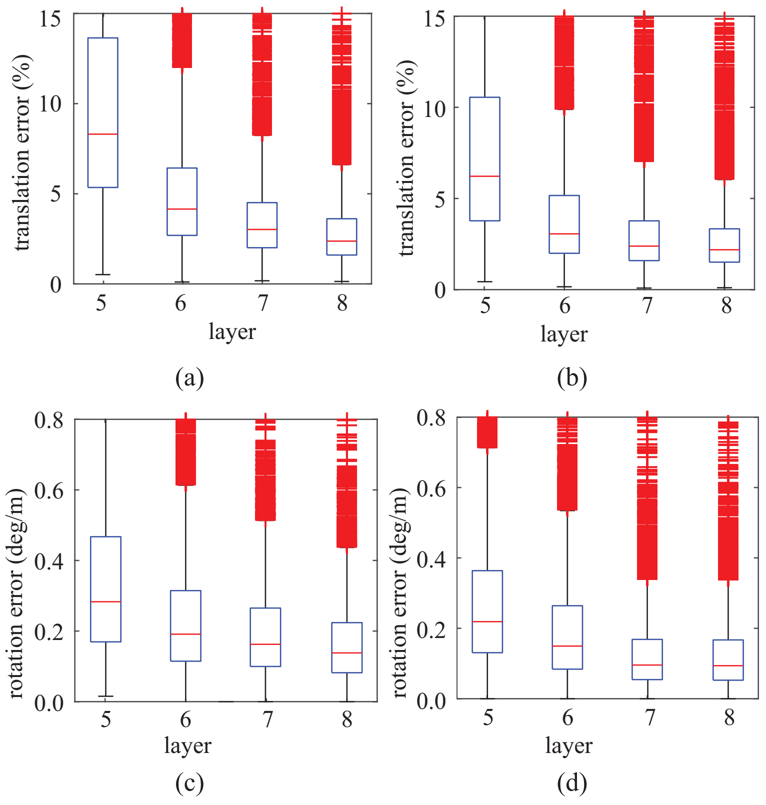


Figure 3.8: Box plots of errors against layers. (a) and (c) are the results of conventional NDT, while (b) and (d) are the results of PNDT.

3.7(d). PNDT computes the modified covariance $\mathbf{\Sigma}^{'}$ and generates distributions as shown in Fig. 3.7(c) and (f).

Due to the high generation rate, the number of values used by PNDT exponentially increases with increasing layer as shown in Fig. 3.6(b). Even, it uses more values than the corresponding point set in layer 12 since a distribution transformed from two or less points still requires nine values. Therefore, it is important to note that PNDT can generate distributions at high-resolution cells, but this can result in low memory efficiency.

3.5.3 Evaluation of Registration

To evaluate the performance of registration, the registrations using conventional NDT and PNDT are conducted in different layers from 5 to 7. We compared the estimated transformation vector $\hat{\mathbf{\Theta}}_k$ to the ground truth Θ_k. Also, the first initial guess $\hat{\Theta}_{1,0}$ is set to $\mathbf{0}$, and the kth initial geuss $\hat{\Theta}_{k,0}$ is set to the preceding transformation vector $\hat{\Theta}_{k-1}$.

As a result, the box plots of translation and rotation errors are shown in Fig. 3.8. Fig. 3.8 (a) and (b) show that PNDT improves the translational registration. The median value in each layer is decreased by applying the presented method. Also, the interquartile range (IQR) of each layer is shorter in Fig. 3.8 (b) than that in Fig. 3.8 (a). Similarly, the improvement of rotational registration by applying the presented method can be seen as shown in Fig. 3.8(c) and (d).

The means of rotation errors, translation errors are shown in Table 3.1. The presented method decreased the translation error by about 20% in layer 5, 6 and about 30% in layer 7, 8. Also, it decreased the rotation error by about 20-25% in layer 5, 6, 7 and about 8% in layer 8. The estimated odometry are shown in Fig. 3.9 and Fig. 3.10. It can be seen that the presented PNDT improves the odometry accuracy.

3.6 Summary

The modification considering pdf based on the sensor model to compute the mean and covariance for NDT is presented. The mean is not changed, but the covariance is changed as the sum of conventional covariance and the average of covariances of point samples. Since each point sample has its covariance, the presented PNDT can generate distributions in all of the occupied cells. Thus, the number threshold of points samples is no more necessary. Also, the generalization between objective functions of NDT-P2D and NDT-D2D is shown. To show the improvement of the presented method, two experiments are conducted. The results of PNDT representation show that all of the occupied cells have GCs regardless of the resolution. On the other hand, the results of registration show that PNDT improves the accuracy of the NDT registration.

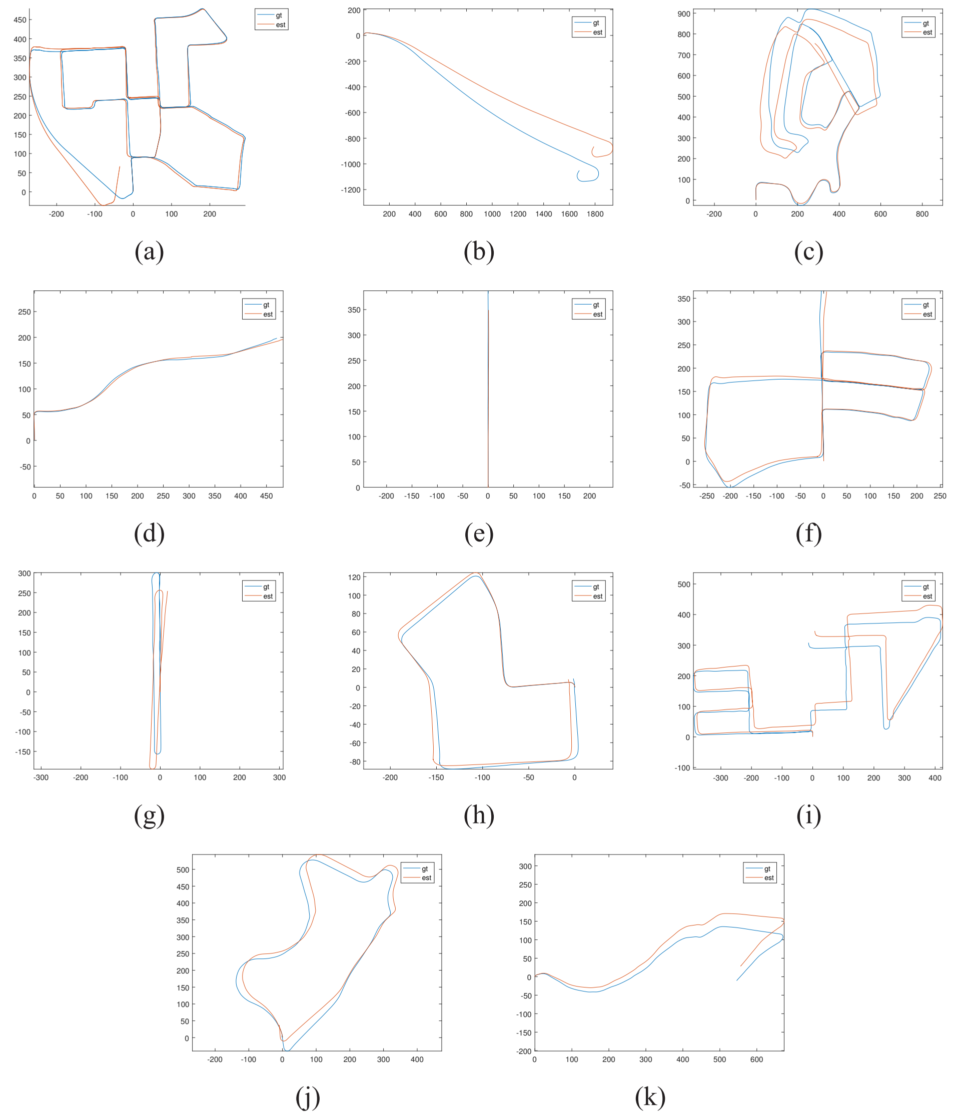


Figure 3.9: Odometry estimated by NDT-D2D. The results of 0th to 10th sequences are in (a) to (k) respectively (cell size=1m).

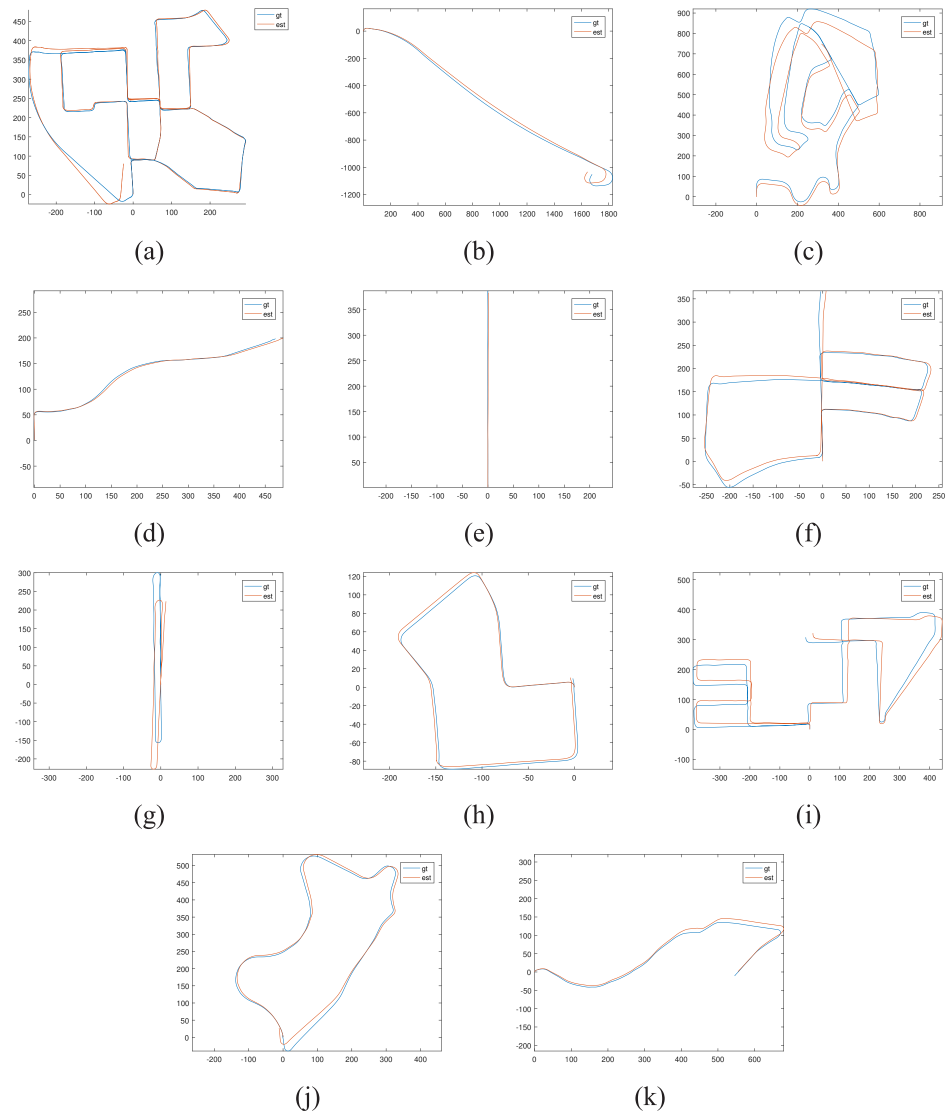


Figure 3.10: Odometry estimated by PNDT-D2D. The results of 0th to 10th sequences are in (a) to (k) respectively (cell size=1m).

Table 3.1: Comparison of odometry accuracy between NDT and PNDT.

cell size		1m		2m		4m	
		NDT	PNDT	NDT	PNDT	NDT	PNDT
Seq 0	T(%)	2.000	1.526	1.83	1.689	2.393	2.076
	R(deg/m)	0.008	0.006	0.007	0.007	0.011	0.008
Seq 1	T(%)	6.906	5.948	13.04	5.535	11.633	6.468
	R(deg/m)	0.012	0.014	0.012	0.012	0.015	0.011
Seq 2	T(%)	3.974	2.929	5.49	3.192	8.014	6.109
	R(deg/m)	0.011	0.008	0.013	0.008	0.02	0.016
Seq 3	T(%)	4.783	3.434	10.817	2.642	8.75	3.8
	R(deg/m)	0.015	0.009	0.018	0.011	0.017	0.012
Seq 4	T(%)	9.112	2.89	14.881	4.095	33.344	5.791
	R(deg/m)	0.009	0.008	0.01	0.008	0.007	0.009
Seq 5	T(%)	1.453	1.482	1.729	1.634	2.285	1.951
	R(deg/m)	0.006	0.006	0.007	0.006	0.008	0.007
Seq 6	T(%)	5.181	4.159	1.746	1.525	2.309	1.653
	R(deg/m)	0.012	0.011	0.006	0.006	0.01	0.008
Seq 7	T(%)	1.22	0.906	2.038	1.121	1.852	1.424
	R(deg/m)	0.005	0.006	0.01	0.007	0.014	0.012
Seq 8	T(%)	2.349	1.526	2.278	1.581	3.781	2.061
	R(deg/m)	0.008	0.006	0.009	0.007	0.015	0.008
Seq 9	T(%)	2.946	1.699	3.424	1.956	4.982	3.357
	R(deg/m)	0.012	0.009	0.014	0.01	0.015	0.014
Seq 10	T(%)	3.557	2.902	3.185	3.089	5.181	4.159
	R(deg/m)	0.01	0.007	0.01	0.01	0.012	0.011

Chapter 4

Interpolation for NDT Using Overlapped Regular Cells

4.1 Introduction

This chapter defines the cell which subdivides the point set $\mathcal{P}$. Since NDT uses the regular cells, it can rapidly search for the correspondences for the source point samples or distributions. However, given a target NDT, as shown in Fig. 4.1(c), the likelihood of observing a point would dramatically decrease due to the discreteness of the target NDT. To overcome this problem, the interpolation method using overlapped regular cells has been proposed, and it can generate an interpolated NDT, as shown in Fig. 4.1(d) [41]. Given the interpolated target NDT $\mathcal{D}_t$, the source point sample $\mathbf{p}_s$ has eight corresponding cells which are overlapped 50% of lengths with their neighbor cells, as shown in Fig. 4.1(b). In [41], the author suggested to sum up the likelihoods obtained by each corresponding cell as the objective function of NDT-P2D as follows:

$$f(\mathcal{P}_s, \mathcal{D}_t, \mathbf{\Theta}) = -\sum_{j=1}^{n_{\mathcal{P}_s}} \sum_{k=1}^{8} r_1 \exp\Big(-\frac{r_2}{2} g_{P2D}(\mathbf{p}_j, d_{corr,jk}, \mathbf{\Theta})\Big), \tag{4.1}$$

where $d_{corr,jk}$ is the kth corresponding GC of $\mathbf{p}_j$. Although the interpolation method improves the registration accuracy, the algorithm computes the likelihoods at most 8

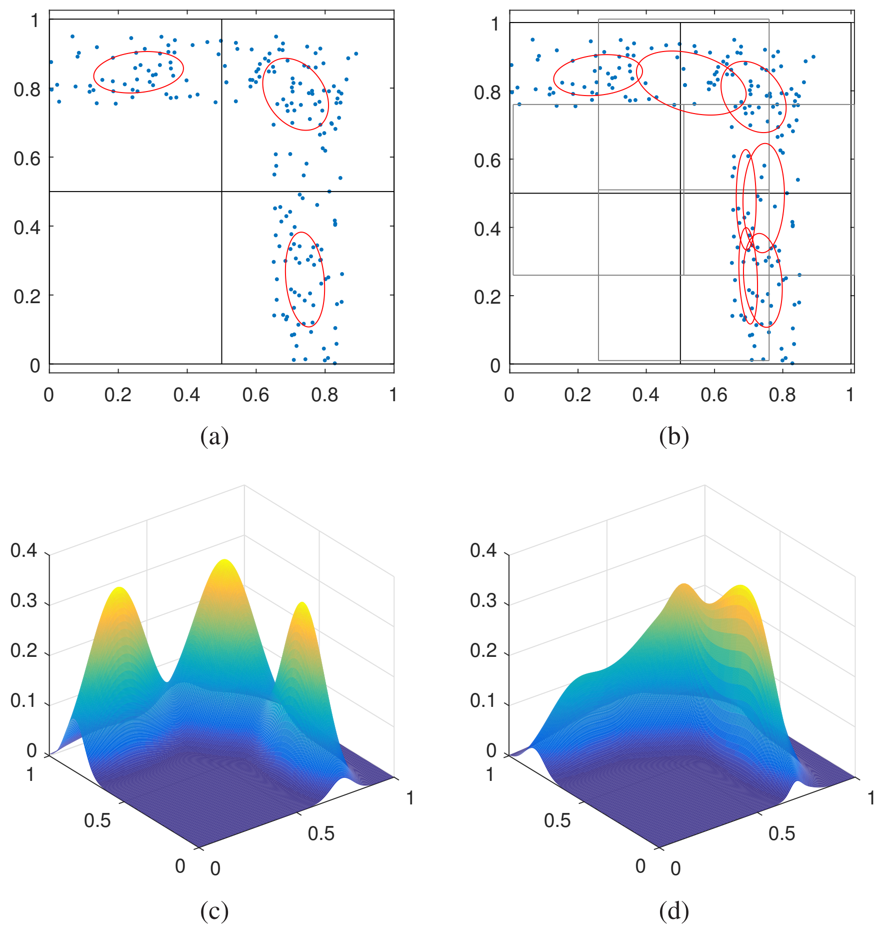


Figure 4.1: Comparison between conventional and interpolated NDTs. (a) and (b) show the 1σ ellipsoids of the conventional and interpolated NDTs, respectively. The likelihoods of two NDTs are demonstrated in (c) and (d), respectively.

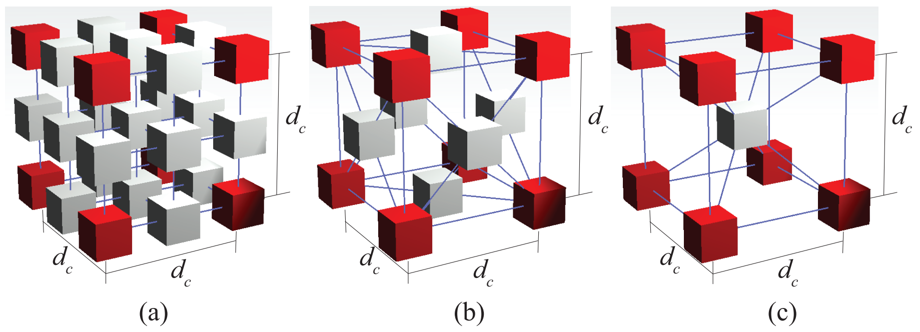


Figure 4.2: Crystalline-structured cells which generate GCs: (a) simple cubic structure, (b) face-centered cubic structure, (c) body-centered cubic structure. Red cubes are conventional cells, and white cubes are inserted cells.

times for each source point and costs about 8 times elapsed time compared to the conventional NDT. To balance the computation and the accuracy, this chapter presents a method of adjusting the distance between cell centroids or the cell length.

Also, inspired by the conventional overlap method, we propose a crystalline NDT generated by cells in the face-centered cubic (FCC) and body-centered cubic (BCC) structures as shown in Fig. 4.2(b) and (c), respectively. Moreover, to reduce the computation caused by multiple correspondences, we present the correspondence region to match corresponding GCs rapidly.

4.2 Lattice Adjustment

In this section, we introduce the interpolation method for NDT-D2D using overlapped regular cells based on octree instead of constructing eight individual octrees [79].

The interpolation using overlapped regular cells is to overcome the discontinuity of NDT. Usually, the registration NDT-D2D inputs a source GC d_s into the octree of the target NDT to search for the correspondence. When the pose $\boldsymbol{\Theta}$ is updated, d_s

may moves to the middle of two target GCs, where the L_2 likelihood can decrease drastically due to the discreteness of the target NDT.

First, we introduce the cells. Given the orthogonal interval $d_{c,f}$ between cells and the length l_f of the cells in the fine layer L_f, as shown in Fig. 4.3, we can obtain the length l_0 of root octree node by

$$l_0 = 2^{L_f} d_{c,f} + (l_f - d_{c,f}), \tag{4.2}$$

and the interval $d_{c,k}$ and length $l_{s,k}$ in kth layer is defined as

$$d_{c,k} = 2^{L_f - k} d_{c,f}, \tag{4.3}$$

$$l_k = d_{c,k} + (l_f - d_{c,f}). \tag{4.4}$$

On each axis, we define $b_{1,i}$ and $b_{2,i}$ on ith axis to subdivide the octree box into three intervals: $[b_{2,i} - l, b_{1,i})$, $[b_{1,i}, b_{2,i})$, and $[b_{2,i}, b_{1,i} + l]$, where $b_{1,i} = c_i - (l - d_c)$, $b_{2,i} = c_i + (l - d_c)$, and c_i is the center coordinate on the ith axis. The point samples are subdivided into 27 subsets (9 subsets for the 2D case), and the subdivision is executed recursively until the length l_k of the cell is equal to l_f.

4.3 Crystalline NDT

In this chapter, we introduce a crystalline NDT whose GCs are generated by the FCC or BCC structured lattice and compare with the conventional NDT interpolated in a simple cubic (SC) structure. Crystalline NDT aims to increase the correspondence candidates for the source GCs. Therefore, we insert GCs into the target NDT $\mathcal{D}_t$ in a crystalline structure. To construct crystalline NDT, two processes of NDT registration

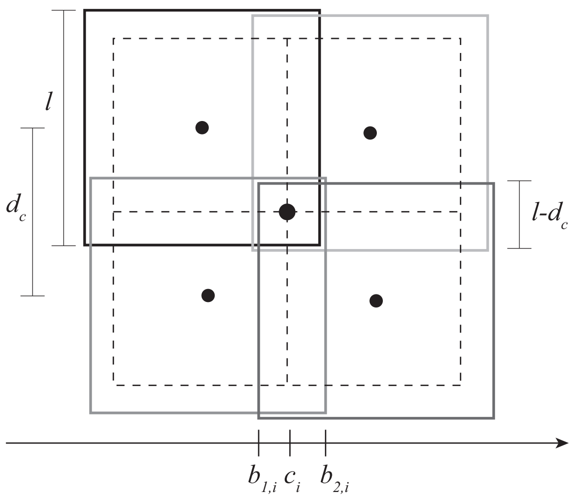


Figure 4.3: Illustration of overlapped octree.

are modified: setting cells of generating NDT and matching the correspondence of registering NDTs.

In the process of setting cells, first, centroids are set with the edge d_c. Second, the cells with the side length l equal to d_c are set. Third, additional cells are inserted. For the insertion in the SC structure, as shown in Fig. 4.2(a), we insert seven lattices translated by $(d_c/2, 0, 0)$, $(0, d_c/2, 0)$, $(0, 0, d_c/2)$, $(d_c/2, d_c/2, 0)$, $(d_c/2, 0, d_c/2)$, $(0, d_c/2, d_c/2)$, and $(d_c/2, d_c/2, d_c/2)$, respectively. For the FCC case, as shown in Fig. 4.2(b), we insert three lattices translated by $(d_c/2, d_c/2, 0)$, $(d_c/2, 0, d_c/2)$, and $(0, d_c/2, d_c/2)$, respectively. For the BCC case, as shown in Fig. 4.2(c), we insert a lattice translated by $(d_c/2, d_c/2, d_c/2)$. Fourth, we compute $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$ of the point set $\mathcal{P}_j = \{\mathbf{p} | d_\infty(\mathbf{p}, \mathbf{c}_j) \leq l/2\}$, where d_∞ is the l_∞-metric, and $\mathbf{c}_j$ is the center of the jth cell.

In the process of matching the correspondence, the source GC has the target GC in the same cell as its correspondence. Since the cells are overlapped, the source GC can have multiple correspondences so that it needs more computation than the conventional NDT registration. To accelerate the registration, we define a correspondence region (CR) in each cell, as shown in Fig. 4.4, to tessellate the space so that each source GC can be matched with a single correspondence. For crystalline NDT in the SC structure (SC-NDT), the kth source GC in the jth cell has the jth target GC as its correspondence if its mean $\mu_k = (\mu_x, \mu_y, \mu_z)^T$ satisfies

$$d_\infty(\mu_k, \mathbf{c}_j) \leq 0.25 d_c, \tag{4.5}$$

where $\mathbf{c}_j = (c_x, c_y, c_z)^T$ is the centroid of the jth cell. For the NDT in the FCC structure (FCC-NDT), the source GC has a correspondence if it satisfies

$$\begin{aligned} d_1((\mu_{k,x}, \mu_{k,y}), (c_{k,x}, c_{k,y})) &\leq 0.5 d_c \cap \\ d_1((\mu_{k,x}, \mu_{k,z}), (c_{k,x}, c_{k,z})) &\leq 0.5 d_c \cap \\ d_1((\mu_{k,y}, \mu_{k,z}), (c_{k,y}, c_{k,z})) &\leq 0.5 d_c \cap \\ d_\infty(\mu_k, \mathbf{c}_j) &\leq 0.5 d_c, \end{aligned} \tag{4.6}$$

where d_1 is the l_1-metric. For the NDT in the BCC structure (BCC-NDT), the source GC has the correspondence if it satisfies

$$d_\infty(\boldsymbol{\mu}_k, \mathbf{c}_j) \leq 0.5 d_c \cap d_1(\boldsymbol{\mu}_k, \mathbf{c}_j) \leq 0.75 d_c. \tag{4.7}$$

The CRs can be illustrated, as shown in Fig. 4.4.

4.4 Experiments

4.4.1 Lattice Adjustment

The experiments were conducted using the KITTI lidar benchmark dataset sequence 0 [83]. We conducted two experiments: one is adjusting d_c as the moderator and fixing l_s

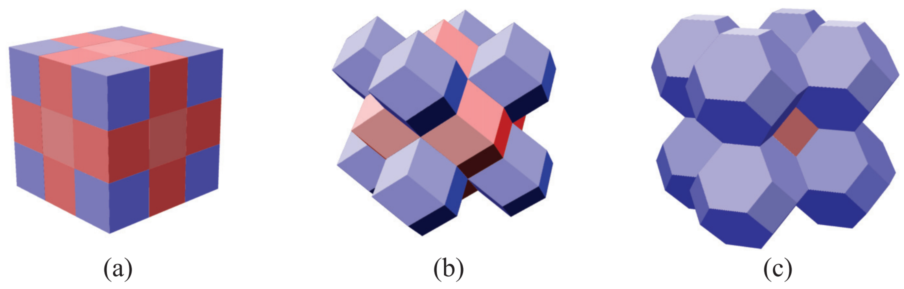


Figure 4.4: Correspondence regions for matching correspondence (a) simple cubic structure, (b) face-centered cubic structure, (c) body-centered cubic structure.

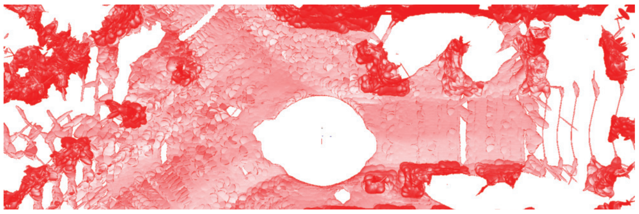
as the control variable, and another one is adjusting l_s as the moderator and fixing d_c as the control variable. The NDTs generated by the presented method can be visualized as shown in Fig. 4.5. The results in Fig. 4.5(a-c) are the case fixing l_s while the results in Fig. 4.5(c-e) are the case fixing d_c.

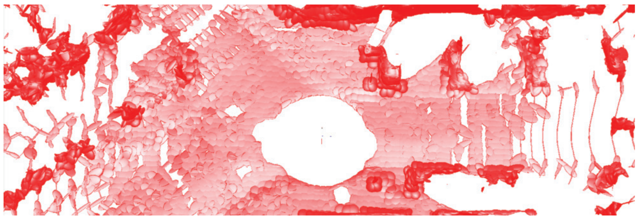
In the following experimental results, we compared the translational error e_t, the rotational error e_r, the runtime of subdivision t_s, and the runtime of registration t_r, respectively. We followed the evaluation method suggested in [83] to compute e_t and e_r which are the average errors of every possible subsequences of length 100,200,...,800m.

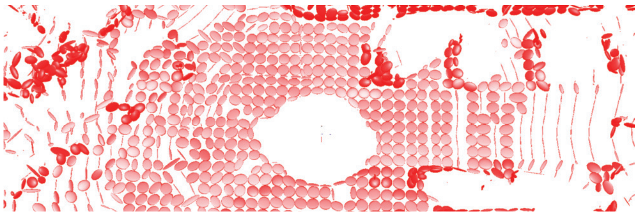
As we suggested in the previous section, since the interpolation is needed by the fixed target NDT $\mathcal{D}_T$, we extracted 3000 samples from the source NDT $\mathcal{D}_s$ for each case. Therefore, t_s indicates how fast it takes to generate interpolated NDT, while t_r indicates how fast the registration takes to be converged.

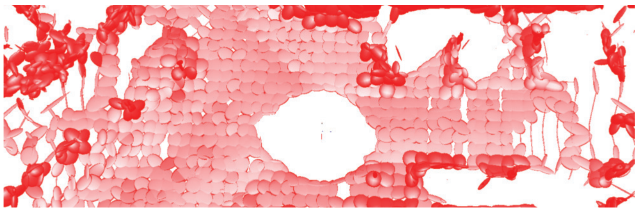
In the first experiment, we investigated the performance against the interval between cells. We fixed the length of cells to 1m and changed d_c from 0.25m to 1m with interval 0.25m, and the accuracy and runtime against d_c are summarized in Table 4.1.

The translational and rotational accuracies are improved by decreasing d_c. It indicates that the NDT can be smoothed to improve accuracy by reducing d_c. However,


(a)


(b)


(c)


(d)

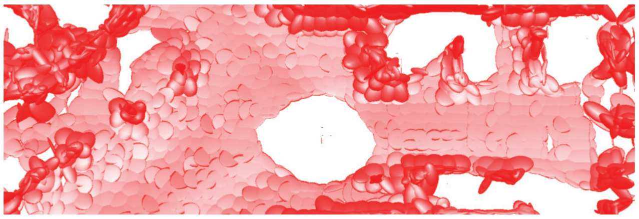

(e)

Figure 4.5: The interpolated NDTs: (a) d_c=0.25m, l=1m, (b) d_c=0.5m, l=1m, (c) d_c=1m, l=1m, (d) d_c=1m, l=1.5m , and (e) d_c=1m, l=2m.

Table 4.1 shows that the d_c and accuracy are not positively correlated. On the other hand, since the total number of GCs of target NDT $\mathcal{D}_t$ is increased as d_c decreases, t_s and t_r are increased as d_c decreases. Also, since the octree becomes deep as d_c decreases, it takes more time to search for correspondence.

Table 4.1: Accuracy and runtime against d_c.

d_c(m)	e_t(%)	e_r(deg/m)	t_s(sec)	t_r(sec)
1.00	4.7373	0.0095	0.1830	0.1676
0.75	1.9383	0.0077	0.2779	0.2159
0.50	1.6055	0.0090	0.6515	0.3670
0.25	1.7357	0.0083	5.8437	0.5205

In the second experiment, we investigated the performance against the length l_s of the cell. We fixed the interval of cells to 1m and changed l_s from 1m to 2m with interval 0.25m. Since d_c is fixed, the depth of octree of $\mathcal{D}_t$ and the number of GCs of $\mathcal{D}_s$ are fixed. The results can be summarized in Table 4.2.

As l_s increases, the covariance of each GC is swelled since the coverage of each cell is extended. As a result, the GC of the interpolated NDT are expanded as shown in Fig. 4.5(d) compared to those of the original NDT in Fig. 4.5(c), and Table 4.2 shows that e_t is improved as l_s increases from 1m to 1.5m while e_r almost remains at the same degree. However, e_t is increased when l_s is 1.75m and 2m. We found the reason in Fig. 4.5(e). The GCs in Fig. 4.5(e) are more smoothed than those in Fig. 4.5(d) so that the shapes of the GCs are similar to the neighbor GCs. In Table 4.2, we can see that t_s is increased as l_s increases. The main reason is that the number of point samples to compute the mean and covariance is increased. On the other hand, t_r does not increase dramatically against l_s since the number of the GCs of $\mathcal{D}_s$ is the same.

Table 4.2: Accuracy and runtime against l_s.

l_s(m)	e_t(%)	e_r(deg/m)	t_s(sec)	t_r(sec)
1.00	4.7373	0.0095	0.1830	0.1676
1.25	2.1023	0.0099	0.2360	0.1748
1.50	1.8166	0.0102	0.3021	0.1934
1.75	1.9806	0.0108	0.3844	0.2044
2.00	2.1745	0.0099	0.4696	0.1941

4.4.2 Performance of Crystalline NDT

Experiments of the crystalline NDT were conducted with the KITTI benchmark dataset sequences 0-10. We evaluated the averages of relative translational and rotational errors e_t and e_r, the elapsed time, and iterations of frame-to-frame PNDT-D2D.

We conducted three experiments for conventional NDT, BCC-NDT, FCC-NDT, and SC-NDT which are visualized, as shown in Fig. 4.6. First experiment is the registrations without CR, second experiment is the registrations with CR, and third experiment is registrations using randomly sampled 1000 source GCs. The side length l of the cell is set to 1m, and d_c is also set to 1m. For optimization, we used Newton method with at most 40 iterations, and r_1 and r_2 being 1 and $1/3$. As a result, the averages of e_t, e_r, elapsed time, and iteration can be summarized as Table 4.3.

Target SC-NDT without CR results in the more substantial translational error than the conventional PNDT. The reason for that is at sequence 6 e_t of SC-NDT is 4.310% while e_t of NDT is 1.320%. Without the sequence 6, the averaged e_t of the SC-NDT is

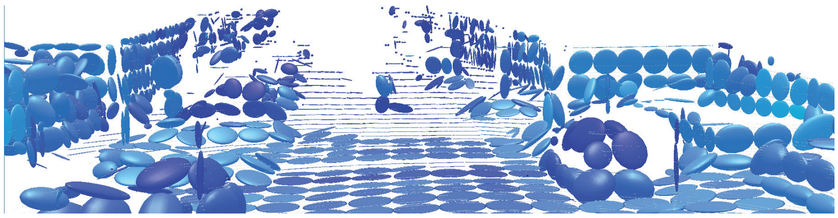
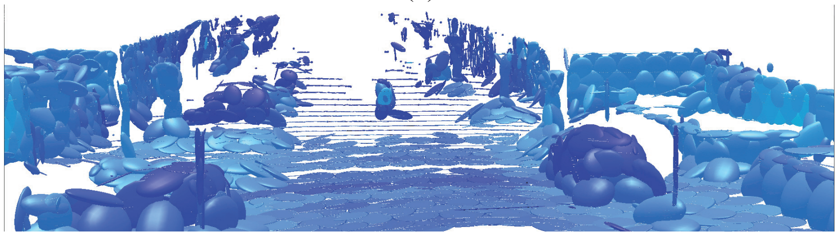
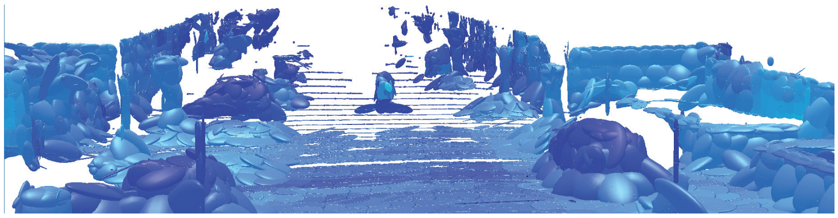
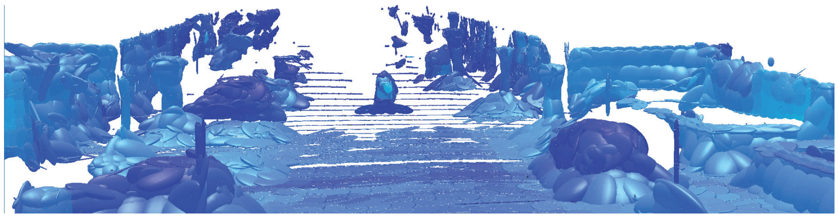
(a)

(b)

(c)

(d)

(e)

Figure 4.6: Crystal structures of cells generating GCs: (a) reference image, (b) conventional NDT, (c) BCC-NDT, (d) FCC-NDT, (e) SC-NDT.

Table 4.3: Pose errors estimated with different lattices.

	avg. e_t(%)	avg. e_r(deg/m)	avg. t (sec)	avg. iter.
conventional NDT	2.534	0.0082	1.027	9.830
SC-NDT, no CR	2.766	0.0081	6.827	10.110
FCC-NDT, no CR	1.932	0.0075	3.463	10.083
BCC-NDT, no CR	1.958	0.0074	1.785	10.164
SC-NDT	2.135	0.0076	2.560	10.365
FCC-NDT	1.940	0.0074	1.683	10.328
BCC-NDT	1.973	0.0074	0.825	10.342
conv.+sampling	2.806	0.0085	0.489	10.574
SC+sampling	2.218	0.0076	1.700	10.560
FCC+sampling	1.948	0.0076	1.430	10.168
BCC+sampling	2.130	0.0076	0.588	10.486

2.611% while that of the conventional NDT is 2.654%. On the other hand, FCC- and BCC-NDT showed similar accuracies, which are higher than the conventional NDT and SC-NDT. However, without the presented CR, each source GC has a maximum of four correspondences from target FCC-NDT and maximum of two correspondences from the target BCC-NDT. As a result, it leads to the increasing runtime and the number of iterations.

The second experiment is performing NDT-D2D applied with CR. Table 4.3 shows that the elapsed times of SC-, FCC-, and BCC-NDT using CRs were shortened by 53.8%, 51.4%, and 62.5% respectively even though the iterations are increased about 2%. Although the averages of e_t and e_r of FCC- and BCC-NDT did not change dramatically using CR, e_t of SC-NDT was significantly improved since e_t of sequence 6 was reduced to 1.031 %.

In the third experiment, the random sampling of the source GCs resulted in shorter runtime for all cases. It decreased by 52.4% , SC-NDT by 33.6%, FCC by 15.0 %, and BCC by 28.7%. However, not only e_t increased with conventional PNDT by 10.7%, BCC by 8.0%, FCC by 0.4%, and SC structure by 3.9%, but also e_r increased with the conventional PNDT by 3.66% , BCC by 2.43%, FCC by 2.57%, and SC structure by 0.13%.

The results showed that the crystalline NDT improves the registration accuracy and the CR improves the processing rate of the registration using crystalline NDT. We also confirmed that the accuracy of registration can be improved by crystalline NDT. To be more specific, the SC-NDT takes the longest time, but has the lowest accuracy. FCC-NDT shows the best accuracy but takes about twice as much time as the conventional NDT. On the other hand, although the accuracy of registration using the BCC-NDT is lower than that of the FCC-NDT, the processing rate is similar to the registration using conventional NDTs.

4.5 Summary

This chapter presents the method of the lattice adjustment for interpolation. Instead of constructing eight individual octrees, the overlapped octree by setting the length and orthogonal interval of cells is utilized to generate overlapped cells. This chapter focuses on the discontinuity of the target NDT influencing the registration performance and claims that the target NDT is the target to be interpolated. This chapter also presents a crystalline NDT representation to improve the accuracy of the pose estimated by NDT-D2D. SC-, FCC-, and BCC-NDTs are introduced as examples of crystalline NDT, and the correspondence region is presented for the processing rate improvement.

The lattice adjustment is evaluated by two experiments. One is inserting the same size of cells. It fixes the cell interval and changes the cell length. Another one is expanding the cells. It fixes the length and changes the cell interval. As a result, since the inserting cells increase the number of GCs to generate, the runtime is increased as the interval decreases. On the other hand, since the number of GCs to generate is the same in the case of expanding cells, runtime generating NDT is not dramatically increased as the length increases. The crystalline NDT is evaluated by three experiments. Among Crystalline NDTs, FCC-NDT showed the best accuracy, and BCC-NDT showed the fastest processing rate.

Chapter 5

Regeneration of Normal Distributions Transform

5.1 Introduction

Although applications of the NDT to SLAM and map matching are proposed as we introduced in Section 2.3, they neglected a problem which arises when the lattices at different poses generate the couple of NDTs which can scarcely be similar. To be specific, SLAM can update poses after a loop closure so that GCs can be transformed, as shown in Fig. 5.1(a). Moreover, if multiple robots are exploring in an environment where communication is not guaranteed, they may not align the lattices and build local NDT maps in inconsistent lattices, as shown in Fig. 5.1(b). If a robot simply fuses the GCs in Fig. 5.1(c), it can distort the NDT map, as shown in Fig. 5.1(d).

To overcome this problem, we present a method which regenerates the NDT such that it becomes suitable for the target lattice. The presented method subdivides the source NDT into truncated GCs using the target lattice and then recomputes the mean, variance-covariance matrix, and the number of points corresponding to the truncated GCs in the same cell.

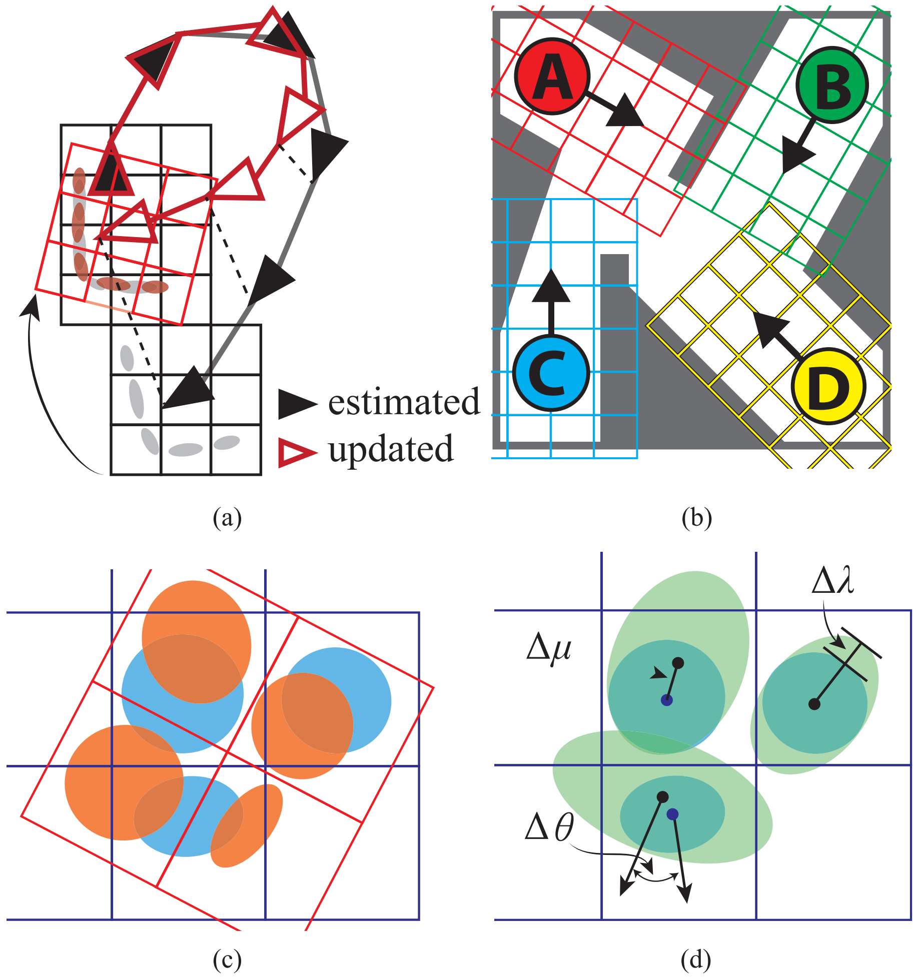


Figure 5.1: Fusion of NDTs in inconsistent lattices. (a) The lattice at each pose is transformed after the poses are updated by SLAM. (b) If multi-robots are unable to align lattices at the initial states, NDTs are generated by inconsistent lattices. (c) The aligned NDT in orange is to fuse with the target NDT in blue. (d) The simple fusion of NDTs can shift the mean by $\Delta\mu$, rotate and scale the covariance by $\Delta\theta$ and $\Delta\lambda$.

5.2 Mathematical Preliminaries

The key of the presented method is recomputing the mean and covariance of each GC truncated by the target lattice. Here, we introduce the derivations of the mean vector and variance-covariance matrix of the truncated trivariate normal distribution in [84].

5.2.1 Trivariate Normal Distribution

For a random vector $\mathbf{x} = (x_1, x_2, x_3)^T \sim \mathcal{N}(\mu, \mathbf{\Sigma})$, we can standardize $\mathbf{x}$ as $\mathbf{z} = (z_1, z_2, z_3)^T$, where $z_i = (x_i - \mu_i)/\sigma_i$, μ_i is the mean of x_i, σ_i is the standard deviation of x_i. Also, the trivariate normal distribution of $\mathbf{x}$ can be standardized as

$$p(\mathbf{z}) = \frac{1}{(2\pi)^{3/2}\Delta^{1/2}} \exp\left(-\frac{1}{2}\mathbf{z}^T A\mathbf{z}\right), \tag{5.1}$$

$$A = \frac{1}{\Delta}\begin{bmatrix} 1-\rho_{23}^2 & \rho_{13}\rho_{23}-\rho_{12} & \rho_{12}\rho_{23}-\rho_{13} \\ \rho_{13}\rho_{23}-\rho_{12} & 1-\rho_{13}^2 & \rho_{12}\rho_{13}-\rho_{23} \\ \rho_{12}\rho_{23}-\rho_{13} & \rho_{12}\rho_{13}-\rho_{23} & 1-\rho_{12}^2 \end{bmatrix}, \tag{5.2}$$

$$\Delta = 1 - \rho_{23}^2 - \rho_{13}^2 - \rho_{12}^2 + 2\rho_{23}\rho_{13}\rho_{12}, \tag{5.3}$$

where ρ_{ij} is the correlation coefficient of z_i and z_j.

5.2.2 Truncated Trivariate Normal Distribution

Given a standardized random vector $\mathbf{z}$ whose pdf is a trivariate normal distribution doubly truncated by the standardized upper truncation $u_i = (U_i - \mu_i)/\sigma_i$ and lower truncation $l_i = (L_i - \mu_i)/\sigma_i$, as shown in Fig. 5.4, the pdf can be expressed as

$$p(\mathbf{z}) = \frac{1}{(2\pi)^{3/2}\Delta^{1/2}(\Phi(u_i) - \Phi(l_i))} \exp\left(-\frac{1}{2}\mathbf{z}^T A\mathbf{z}\right), \tag{5.4}$$

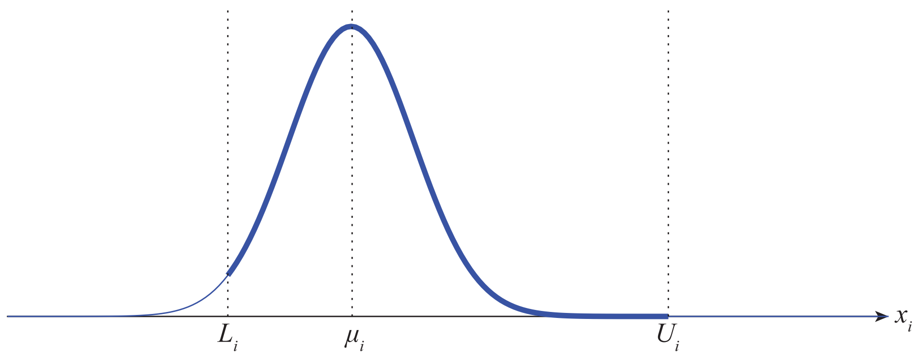


Figure 5.2: Illustration of the truncated normal distribution.

where z_i satisfies $l_i < z_i \leq u_i$, and $\Phi(z)$ is the cumulative density function (cdf) of z which is defined as

$$\Phi(z) = P(y \leq z) = \int_{-\infty}^{z} \phi(y)dy, \tag{5.5}$$

$$\phi(z) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}z^2\right). \tag{5.6}$$

To compute the mean vector and variance-covariance matrix of $\mathbf{z}$, first, the expectation and variance of z_i are computed as:

$$E[z_i] = -\frac{\phi(u_i) - \phi(l_i)}{\Phi(u_i) - \Phi(l_i)}, \tag{5.7}$$

$$Var(z_i) = 1 - \frac{u_i\phi(u_i) - l_i\phi(l_i)}{\Phi(u_i) - \Phi(l_i)} - \left(\frac{\phi(u_i) - \phi(l_i)}{\Phi(u_i) - \Phi(l_i)}\right)^2. \tag{5.8}$$

Next, for variables $z_j, j = 1, 2, 3, j \neq i$, the mean, variance, and covariance can be derived as:

$$E[z_j] = \rho_{ij}E[z_i], \tag{5.9}$$

$$Var(z_j) = \rho_{ij}^2 Var(z_i) + 1 - \rho_{ij}^2, \tag{5.10}$$

$$Cov(z_i, z_j) = \rho_{ij}Var(z_i). \tag{5.11}$$

Also, the covariance of z_j and z_k, where $k = 6 - i - j$, can be derived as

$$Cov(z_j, z_k) = \rho_{ij}\rho_{ik}Var(z_i) + \rho_{jk} - \rho_{ij}\rho_{ik}. \tag{5.12}$$

Finally, for the random vector $\mathbf{x} = (x_1, x_2, x_3)^T$, the mean, variance, and covariance can be computed as follows:

$$E[x_i] = \sigma_i E[z_i] + \mu_i, \tag{5.13}$$

$$Var(x_i) = \sigma_i^2 Var(z_i), \tag{5.14}$$

$$Cov(x_i, x_j) = \sigma_i \sigma_j Cov(z_i, z_j), j \neq i. \tag{5.15}$$

5.3 Regeneration of NDT

The regeneration method can be divided into four processes: alignment, subdivision, regeneration, and fusion. First, the alignment process aligns the source NDT D_n to the target NDT D_t with the estimated relative transformation T. Second, the subdivision process subdivides the transformed source GCs into the truncated GCs using the target lattice C_t. Third, the regeneration process fuses the truncated GCs in each cell. Finally, the fusion process fuses the regenerated GC and the existing GC in the same cell. We depict the method as shown in Fig. 5.3.

5.3.1 Alignment

The alignment process transforms the source NDT $D_n = \{\mathcal{N}(\boldsymbol{\mu}_{n,i}, \boldsymbol{\Sigma}_{n,i})\}_{i=1}^{n_D}$ with the estimated relative transformation $\boldsymbol{\Theta}$ by (2.24) and (2.25) and obtains the aligned NDT $D_a = \{\mathcal{N}(\boldsymbol{\mu}_{a,i}, \boldsymbol{\Sigma}_{a,i})\}_{i=1}^{n_D}$. After the transformation, the aligned GCs can cross two or more cells, as shown in Fig. 5.4(a).

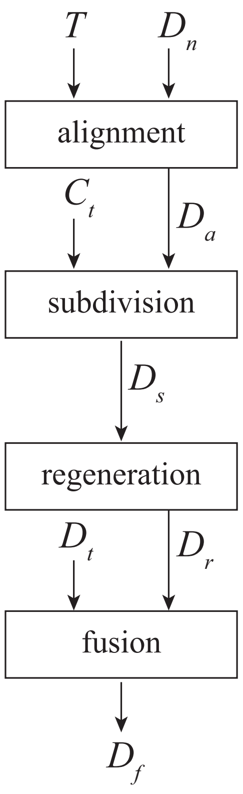


Figure 5.3: Block diagram of the regeneration method.

5.3.2 Subdivision of Gaussian Components

The presented regeneration method subdivides the NDT on the ith axis with $i = 1, 2, 3$ sequentially. Suppose that the side length of the target cell is l, the GC has an interval $[kl, (k+1)l)$ on the ith axis, as shown in Fig. 5.4(a). The subdivision of a GC can be divided into three processes: determining truncation value, examining the weights, computing the number of points, mean, and covariance matrix.

First, the bound nearer to the mean on the ith axis is chosen as the truncation value B. For example, B is set to kl shown in Fig. 5.4(a). The GC can be divided into two truncated normal distributions with truncation values $(-\infty, B)$ and $[B, \infty)$.

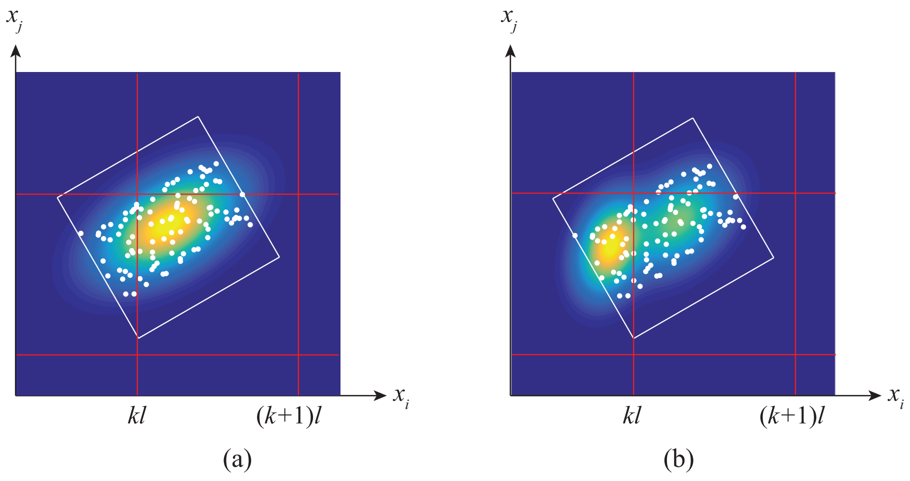


Figure 5.4: Subdivision of the GC. Red boxes are the target lattice, white box is the source cell, and white raw points are converted into a GC. The GC in (a) is subdivided by the truncation value kl, and two GCs are regenerated in (b).

Second, we define a weight threshold $w_{th} \in (0, 0.5)$ to check whether the weights $w_l = \Phi(b)$ and $w_h = 1 - \Phi(b)$, where $b = (B - \mu_{a,i})/\sigma_{a,i}$, are in the interval $(w_{th}, 1 - w_{th})$. If the weights are in the interval, the subdivision is performed; otherwise, it is ignored. Two special cases are notable. It always subdivides GCs if $w_{th} = 0$, while it does not subdivide any GCs if $w_{th} = 0.5$.

Third, if w_l and w_h are in the interval $(w_{th}, 1 - w_{th})$, the algorithm computes the weighted number n_w of points, mean μ_w, and variance-covariance matrix $\mathbf{\Sigma}_w$ for each truncated distribution. n_w is computed by $n_w = wn_a$, where n_a is the number of points corresponding to the source GC, and $\boldsymbol{\mu}_w$ and $\mathbf{\Sigma}_w$ are computed by (5.7)-(5.15). As a result, the source GC can be subdivided into two GCs, as shown in Fig. 5.4(b).

After the subdivision is completed on the ith axis, the same process is performed on the other axes. At the end, we can obtain the truncated NDT $D_s = \{\mathcal{N}(\boldsymbol{\mu}_{s,i}, \boldsymbol{\Sigma_{s,i}})\}, i = 1, 2, 3, ..., n_{D,s}$. For the worst case, the subdivision on an axis generates $2n_D$ GCs with the computational complexity $O(n_D)$. Thus, for the N dimensional subdivision, we can derive the computational complexity as $O((2^N - 1)n_D)$.

5.3.3 Fusion of Gaussian Components

Although the regeneration and fusion processes perform the same work, we present the two-step fusion for the applications such as detecting the difference between the regenerated NDT and existing NDT. Two processes fuse the NDTs using different equations. The regeneration process computes the number n_r of the points in a cell as

$$n_r = \sum_{i=1}^{n_{D,s}} n_{s,i} \tag{5.16}$$

and computes the regenerated mean μ_r and sample variance-covariance matrix $\mathbf{\Sigma}_r$ as follows:

$$\mu_r = \frac{1}{n_r} \sum_{i=1}^{n_{D,s}} n_{s,i} \boldsymbol{\mu}_{s,i}, \tag{5.17}$$

$$\mathbf{\Sigma}_r = \frac{1}{n_r} \sum_{i=1}^{n_{D,s}} n_{s,i} \left(\mathbf{\Sigma}_{s,i} + \boldsymbol{\mu}_{s,i} \boldsymbol{\mu}_{s,i}^T \right) - \mu_r \mu_r^T. \tag{5.18}$$

On the other hand, the fusion process fuses the regenerated NDT and existing NDT using (2.16)-(2.22). The time complexities of (2.16)-(2.22) and (5.16)-(5.18) are $O(n_{D,s})$. However, due to the numbers of additions, multiplications, and divisions, (5.16)-(5.18) are faster than (2.16)-(2.22) when $n_{D,s} > 2$. Thus, the regeneration uses (5.16)-(5.18) instead of (2.16)-(2.22) if $n_{D,s} > 2$. The reason is that equations (2.16)-(2.22) add $31n_{D,s} - 31$ times, multiply $31n_{D,s} - 19$ times, and divide $10n_{D,s} - 1$ times while equations (5.16)-(5.18) add $22n_{D,s} - 4$ times, multiply $21n_{D,s} + 9$ times, and divide 12 times. Although the division is slower than the multiplication, we can assume that both are the same. We referred the time for addition as t_a and the time for multiplication as t_m, and we obtained the elapsed time of equations (2.16)-(2.22), equal to

$(31n_{D,s} - 31)t_a + (41n_{D,s} - 20)t_m$, and the elapsed time of equations (5.16)-(5.18), equal to $(22n_{D,s} - 4)t_a + (21n_{D,s} + 9)t_m$. Assume that multiplication is as fast as addition, the elapsed time equations become $(72n_{D,s} - 51)t_a$ and $(43n_{D,s} + 5)t_a$. In this case, if $n_{D,s} > 2$, equations (5.16)-(5.18) are faster than equations (2.16)-(2.22). Since addition is usually much faster than multiplication, we compared two cases ignoring additions. In this case, equations (5.16)-(5.18) are also faster than equations (2.16)-(2.22) if $n > 2$.

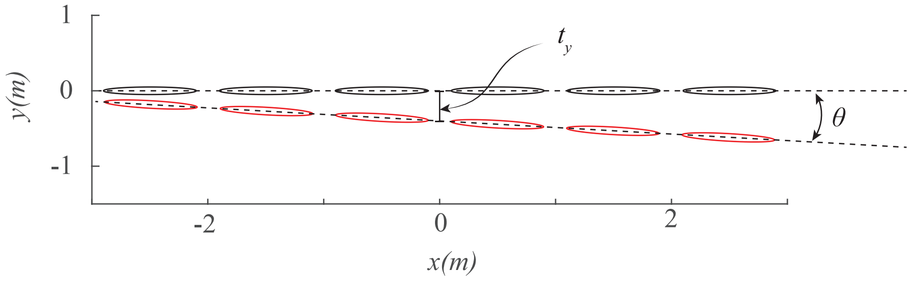
If there are no other applications between regeneration and fusion processes, it is recommended to fuse the truncated GCs and existing GC using (5.16)-(5.18).

5.4 Experiments

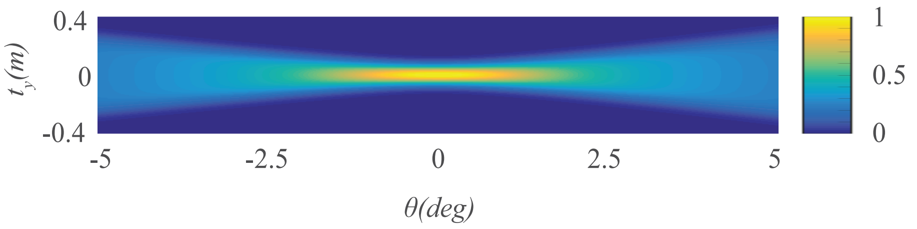
To evaluate the presented method, we used KITTI benchmark dataset captured via Velodyne HDL-64E [83], a 64-channel 3D LIDAR, at 10Hz, and Freiburg benchmark dataset captured via a structured-light based Microsoft Kinect at 30Hz [85].

5.4.1 Evaluation Metrics for Representation

To evaluate the spatial representation performance of the NDT regenerated by the target lattice, we used the following metrics. First, we evaluated the performance of the NDT in the receiver operating characteristic (ROC) domain [86]. We define a cell with a GC as in a positive state and a cell without a GC as in a negative state. In this context, the state of a cell with a regenerated GC is predicted to be positive, whereas a cell without a regenerated GC is predicted to be negative. When a cell is predicted to be positive, it is a true positive (TP) if the actual state of the cell is positive; otherwise, it is a false positive (FP) if the actual state is negative. Likewise, when a cell is predicted to



(a)



(b)

Figure 5.5: L_2 likelihood against translational value t_y and the rotational angle θ. This simulation rotates red GCs in (a) from -5 to 5 degrees and moves them from -0.4 to 0.4 on y axis. The result of the simulation is shown in (b).

be negative, it is a true negative (TN) if the actual state is negative or is a false negative (FN) if the actual state is positive. These TP, FP, TN, and FN designations can be used to compute the true positive rate (TPR) and false positive rate (FPR) as follows:

$$TPR = \frac{TP}{TP+FN}, \quad FPR = \frac{FP}{FP+TN}, \tag{5.19}$$

and the TPR and FPR determined by the discrimination threshold form a curve in the ROC domain. We also used the accuracy rate, which is computed as

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}. \tag{5.20}$$

Additional metrics are the mean L_2 value d_{L_2} and the mean error d_m used in [75]. To use these metrics, we construct a true sample covariance (TSC) NDT [75], which is converted from the accumulated point sets using the target lattice, as the ground truth. Here, d_m is the mean of the distance between the centers of the predicted GC $\mathcal{N}(\mu_p, \mathbf{\Sigma}_p)$ and the TSC GC $\mathcal{N}(\mu_t, \mathbf{\Sigma}_t)$ computed as

$$d_m = \frac{1}{n}\sum_{i=1}^{n} ||\mu_{p,i} - \mu_{t,i}||, \tag{5.21}$$

where n is the number of the predicted GCs. Additionally, d_{L_2} is the mean of the L_2 likelihood of $\mathcal{N}(\mu_p, \mathbf{\Sigma}_p)$ and $\mathcal{N}(\mu_t, \mathbf{\Sigma}_t)$, which is computed as

$$d_{L_2} = \frac{1}{n}\sum_{i=1}^{n} r_1 \exp\left(-\frac{r_2}{2}(\mu_{p,i} - \mu_{t,i})^T(\mathbf{\Sigma}_{p,i} + \mathbf{\Sigma}_{t,i})^{-1}(\mu_{p,i} - \mu_{t,i})\right), \tag{5.22}$$

where r_1 and r_2 are regularizing factors [45, 75]. However, if the source and target GCs are not sufficiently close, as shown in Fig. 5.5(a), d_{L_2} can be negatively correlated with the rotational alignment, as shown in Fig. 5.5(b).

Because d_m is the Euclidean distance between the centers of GCs, to evaluate the shape similarity of NDTs, we paid attention to the Fréchet distance between two Gaussian distributions, which is expressed as

$$d_F^2 = ||\mu_p - \mu_t||^2 + tr[\mathbf{\Sigma}_p + \mathbf{\Sigma}_t - 2(\mathbf{\Sigma}_p\mathbf{\Sigma}_t)^{1/2}], \tag{5.23}$$

where the first term is the distance between the centers in the Euclidean space, and the second term is the distance in the space of covariance matrices [87]. In this chapter, we refer to the second term as the covariance distance and define the mean covariance distance as

$$d_{cov} = \frac{1}{n}\sum_{i=1}^{n} \sqrt{tr[\mathbf{\Sigma}_{p,i} + \mathbf{\Sigma}_{t,i} - 2(\mathbf{\Sigma}_{p,i}\mathbf{\Sigma}_{t,i})^{1/2}]}. \tag{5.24}$$

5.4.2 Representation Performance of the Regenerated NDT

In this experiment, we evaluate the representation performance of the regenerated NDT against the weight threshold w_{th}. We set the ground truth Θ to $(0.5m$, $0m$, $0.5m$, 0,

0.16π, $0)$. For the KITTI dataset, the cell size varies from 0.5m to 3m at intervals of 0.5m, and for Freiburg dataset, the cell size varies from 0.05m to 0.3m at intervals of 0.05m.

In the KITTI case, the point set in Fig. 5.6(a) is converted into an NDT in Fig. 5.6(b) using the local lattice. If the NDT is naively regenerated by the target lattice, the GCs can be distorted, as shown in Fig. 5.6(c). Compared to the TSC NDT in Fig. 5.6(e), some cells in Fig. 5.6(c) have no GCs and are therefore false negatives. On the other hand, the NDT regenerated by the presented method in Fig. 5.6(d) generates GCs suitable for the target cell. Similarly, we conducted an experiment with the Freiburg dataset; the visualized results in this case are shown in the second row in Fig. 5.7. The Distortion of the GCs is also apparent in Fig. 5.7(c) as compared to Fig. 5.7(d) and (e).

To verify the performance of the presented method, we investigated the performance using the ROC curve, accuracy, L_2 value, mean error, and covariance distance, as shown in Fig. 5.8 and 5.9. The ROC curves of the NDTs of KITTI regenerated with different cell sizes are shown in Fig. 5.8(a). This figure shows that it approaches the ideal coordinate $(0, 1)$ in the ROC domain as the cell size decreases. It also shows that as w_{th} decreases, the TPR increases, as does the FPR. Moreover, we obtained accuracy plots against w_{th}, as shown in Fig. 5.8(b), which shows that the smaller the cell size becomes, the better the accuracy is. Also, the accuracy is improved as w_{th} decreases, and this relationship is obvious for large cells. However, the plot shows that the accuracy can decrease as w_{th} approaches zero. The Freiburg dataset also shows a similar pattern, as presented in Fig. 5.9(b), as the small value of w_{th} leads to a strict subdivision. For example, if $w_{th} = 0$, the presented method always subdivides the source NDT using the target lattice, and it generates GCs in the empty cells of the TSC NDT.

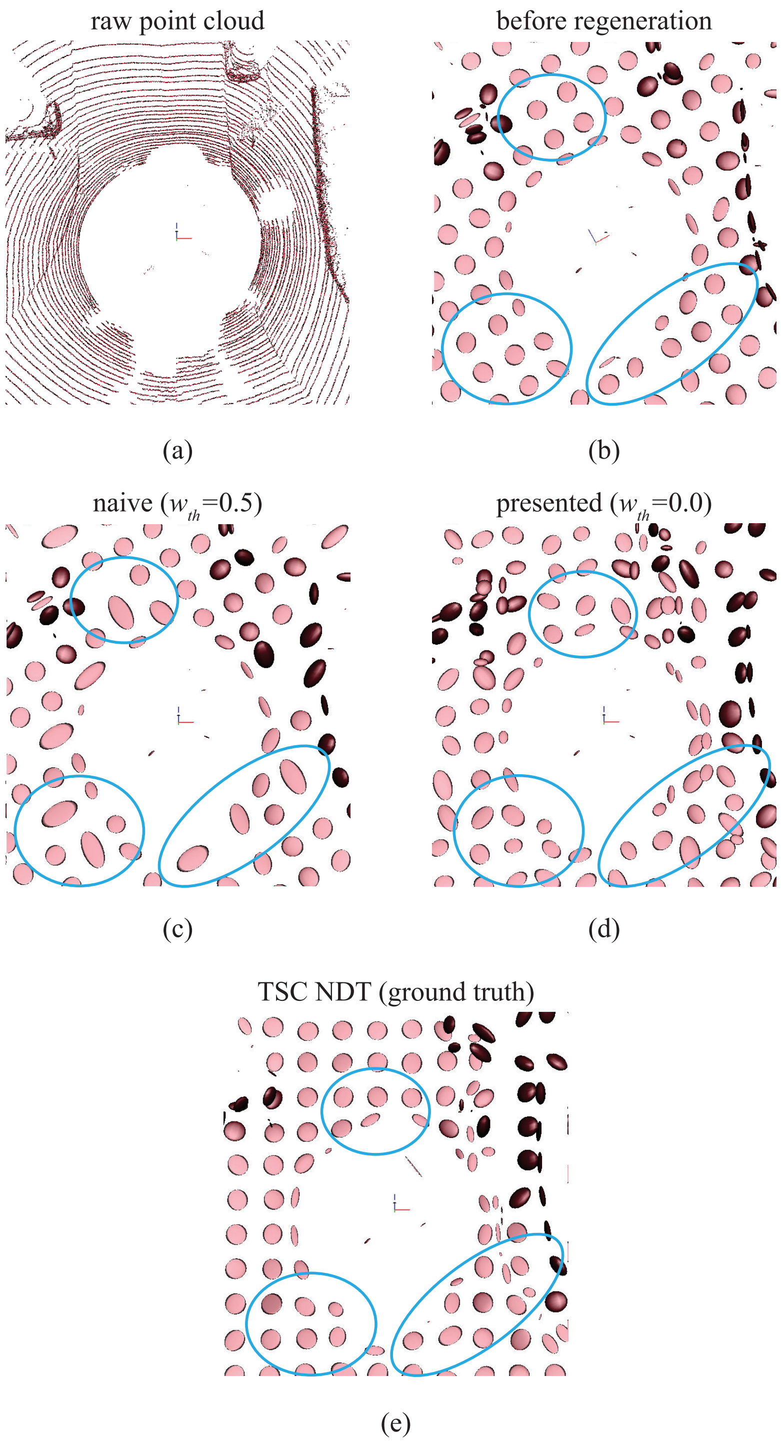


Figure 5.6: Visualization of the point set and NDTs of KITTI.

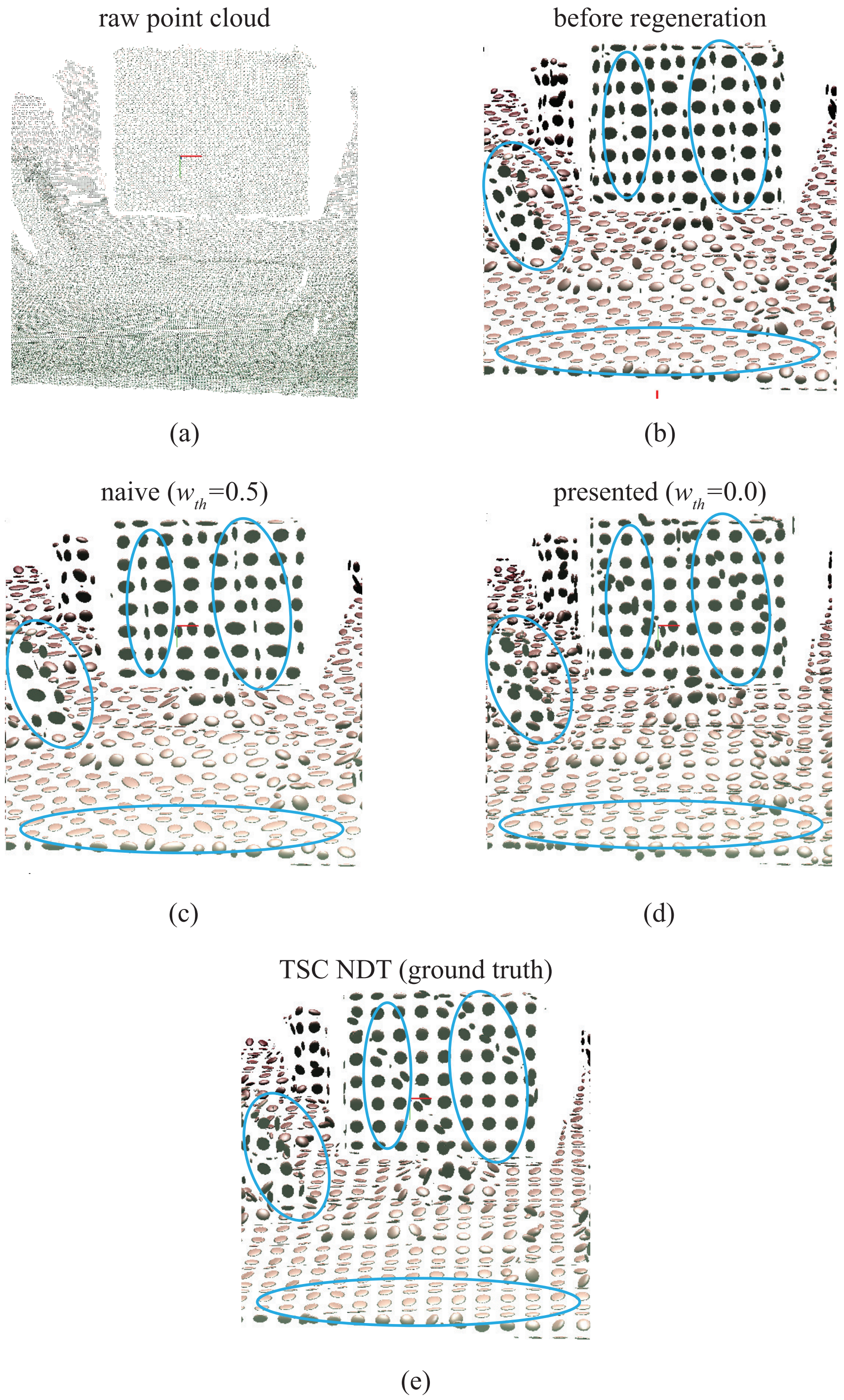
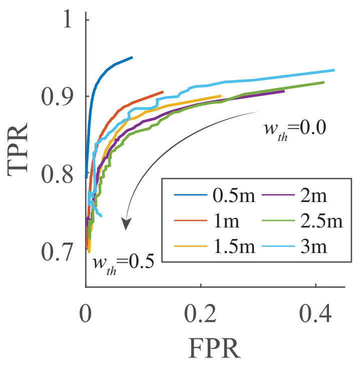


Figure 5.7: Visualization of the point set and NDTs of Freiburg.

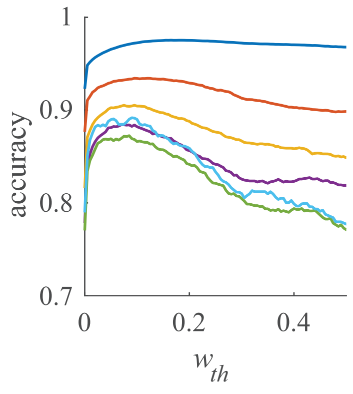
It means that the FPR increases, which in turn lowers the accuracy as computed by (5.20). The ROC curves in Fig. 5.8(a) and Fig. 5.9(a) also show that the FPRs increase as w_{th} decreases.

We also evaluated the performance using the mean L_2 value, mean error, and covariance distance. The mean L_2 values d_{L_2} of the KITTI and Freiburg data sets are shown in Fig. 5.8(c) and Fig. 5.9(c), respectively. These outcomes indicate that d_{L_2} increases as w_{th} decreases from 0.5. However, similar to the accuracy outcome, d_{L_2} declines as w_{th} approaches zero. On the other hand, we evaluated the mean error d_m and covariance distance d_{cov}. The mean errors d_m against w_{th} with different cell sizes are shown in Fig. 5.8(d) and Fig. 5.9(d). This finding indicates that d_m decreases as w_{th} decreases, but, similar to d_{L_2}, it worsens as w_{th} approaches zero. Unlike d_m, d_{cov} with different cell sizes decreases as w_{th} decreases as shown in Fig. 5.8(e) and Fig. 5.9(e).

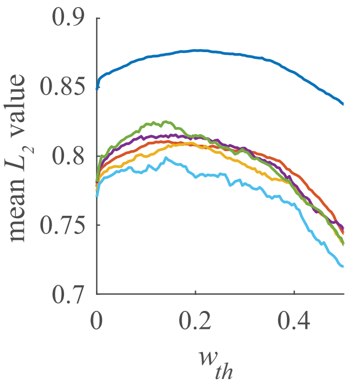
Furthermore, to determine the relationships between the metrics and w_{th}, we analyzed the KITTI case using 2.5m cells. As a result, Fig. 5.10(a) shows that the lower whisker of the L_2 value decreases as w_{th} decreases from 0.2 to 0, while the median maintains a similar L_2 value. Moreover, d_m against w_{th} in Fig. 5.10(b) shows a similar pattern. On the other hand, d_{cov} in Fig. 5.10(c) shows that the lower whisker, lower quartile, and median all decrease as w_{th} decreases, while the upper whisker and upper quartile maintain similar values. Thus, from these plots, we inferred that if w_{th} approaches zero, the regenerated covariances become similar to the corresponding ground truth covariances. However, the regenerated GCs can be far from the corresponding GCs of the TSC NDT.



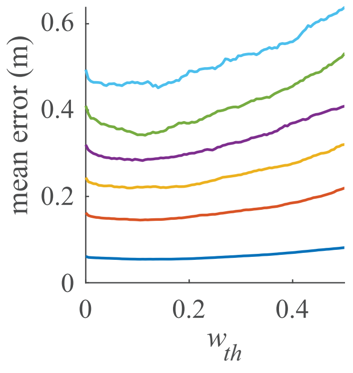
(a)



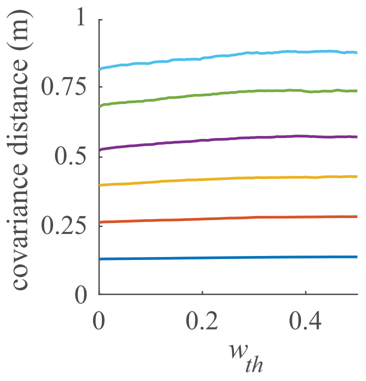
(b)



(c)

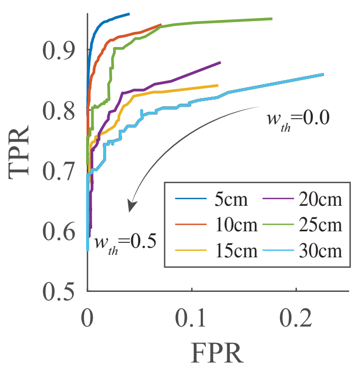


(d)

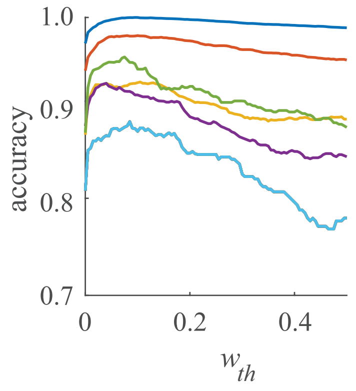


(e)

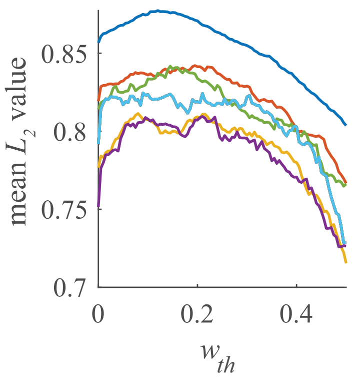
Figure 5.8: Results of KITTI point sets. (a) ROC curve (b) accuracy (c) mean L_2 value (d) mean error (e) mean covariance distance. (naive case: $w_{th} = 0.5$)



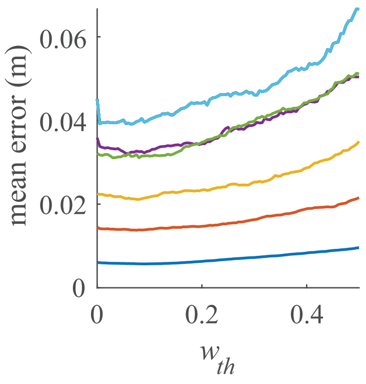
(a)



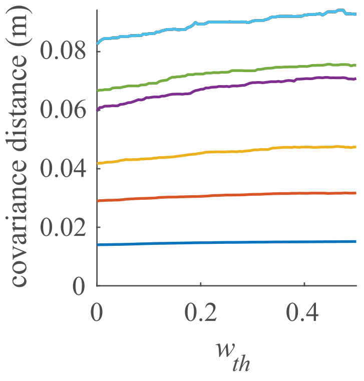
(b)



(c)



(d)



(e)

Figure 5.9: Results of Freiburg point sets. (a) ROC curve (b) accuracy (c) mean L_2 value (d) mean error (e) mean covariance distance. (naive case: $w_{th} = 0.5$)

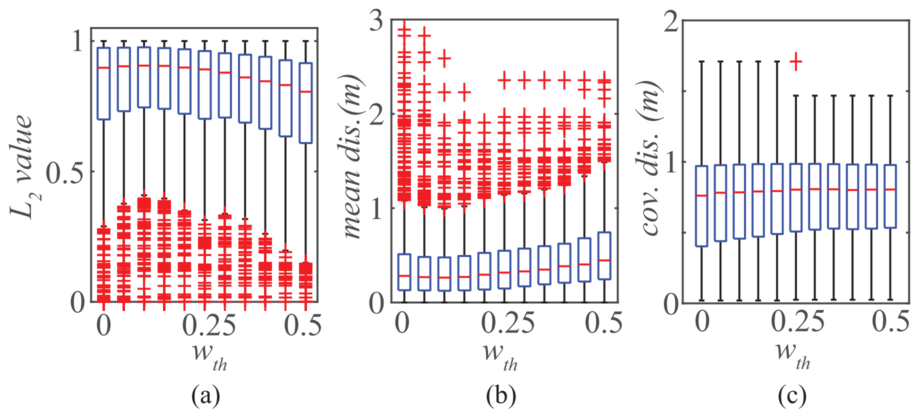


Figure 5.10: Boxplots against w_{th}: (a) L_2 value, (b) mean distance, (c) covariance distance.

5.4.3 Computation Performance of the Regeneration

We evaluated the elapsed time of regeneration and the number of values used by regeneration. Here, Fig. 5.11(a) shows that the elapsed time for the KITTI point set increases as w_{th} decreases. In addition, the elapsed time increases as the cell size decreases. This likely occurs due to the number of GCs, as the average elapsed time with different cell sizes is 12-22 μs per GC, and even the elapsed time for the 0.5m cell is smaller than that for the 3m cell case, as shown in Fig. 5.11(b). In Fig. 5.11(a), it also shows that the regeneration process can exceed 10Hz with $w_{th} = 0$ except with a 0.5m cell size. Given that the upper bound of the average elapsed time is 22 μs, we suggest holding the number of GCs to fewer than 4546 for Velodyne HDL-64E sensor.

Similar to the elapsed time, the number n_V of the values used by the NDT increases as w_{th} decreases, as shown in Fig. 5.11(c). We chose the best case of the presented method using $w_{th} = 0.155$ and a 0.5m cell size, as this method has the highest number of values relative to that of naive regeneration, and we found that n_V in this case is approximately 105.0k while n_V in the naive case is approximately 87.0k. Compared to n_V of the raw point set, equal to 374.0k, the regenerated NDT shows compactness. Also, compared to n_V before the regeneration, equal to 110.0k, n_V of the presented method is close to 95.5% while n_V in the naive case is close to 79%. These findings indicate that 21% of the GCs are fused with the neighbor GCs regenerated by the target lattice, as shown in Fig. 5.6(c).

We found that the Freiburg point set also shows similar patterns to those of the KITTI point set. The elapsed time increases as w_{th} or the cell size decreases, as indicated in Fig. 5.11(d), and the average elapsed time is 11-44 μs per GC, as shown in Fig. 5.11(e). Thus, we suggest holding the number of GC to fewer than 757 for Kinect to regenerate online. The plot in Fig. 5.11(d) also shows that the presented method regenerates the NDT with $w_{th} = 0$ at a rate faster than 30Hz, except for the 5cm cell case. Similar to the KITTI point set, n_V of the NDTs regenerated by the naive and presented methods with 5cm cells and $w_{th} = 0.085$ are 33.2k and 38.4k, respectively, while n_V of the point set is 614.6k. Compared to n_V of the NDT before regeneration, equal to 44.8k, n_V of the presented method is 6.4k less, as it does not generate GCs corresponding to all of the ground-truth GCs. On the other hand, n_V of the naive method is 11.6k less because the GCs in the same target cell are fused after regeneration, as shown in Fig. 5.7(c).

5.4.4 Application of Map Fusion

In this experiment, we compared the application of map fusion between the conventional and presented methods using sequence 0 of KITTI. The cell size is set to 1m for these methods, and w_{th} is set to 0.1 for the presented method. Also, we assumed the following scenario. Suppose that robot A R_A and robot B R_B create maps with-

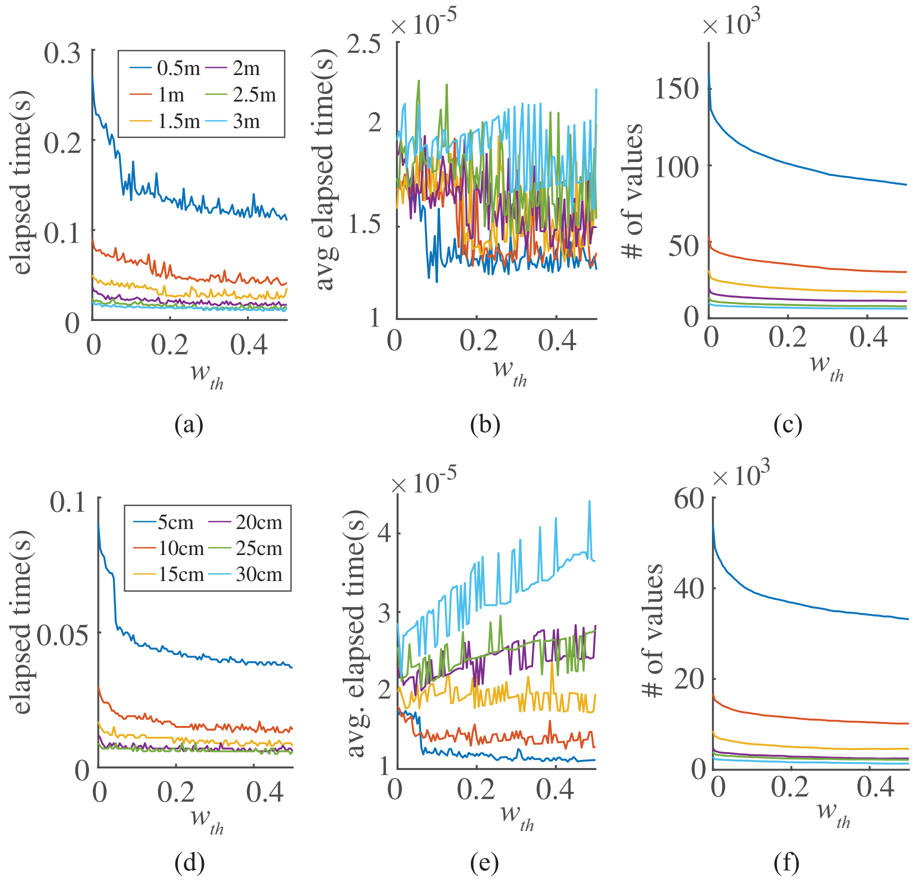


Figure 5.11: Elapsed time of regeneration, average elapsed time, and required number of values used by the regenerated NDT from left to right. The top row is the results of KITTI, and the bottom row is the results of Freiburg. (naive case: $w_{th} = 0.5$)

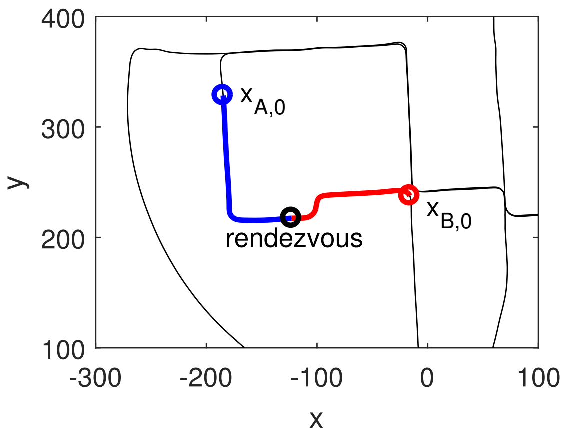
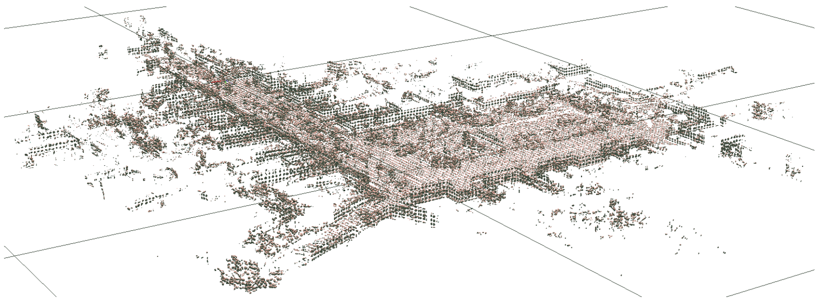


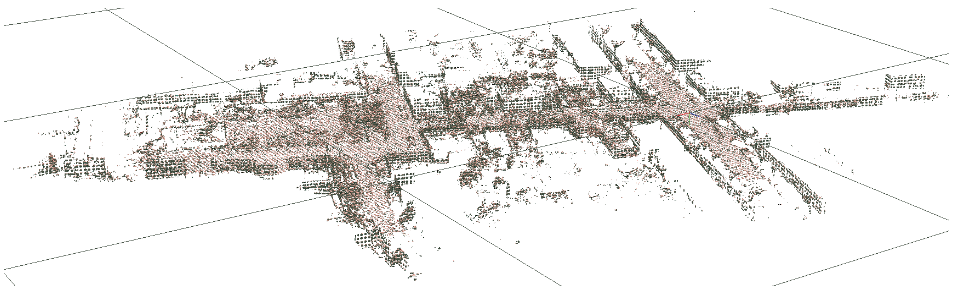
Figure 5.12: Paths of two robots.

out the complementary initial poses. If a rendezvous occurs, R_A estimates the relative pose T between R_B and itself and receives the NDT map $\mathcal{D}_{M,B}$ from R_B. Next, R_A transforms $\mathcal{D}_{M,B}$ with T and fuses it with $\mathcal{D}_{M,A}$. Because this experiment aims to verify the representation performance of NDT map in how it fuses the existing NDT and the NDT regenerated by the presented method, we applied the poses $\mathbf{x}_{gt}[k]$, $k = 0, 1, 2, ..., 4540$, provided by KITTI to exclude the error caused by the point set registration process. To be specific, R_A starting from $\mathbf{x}_A = \mathbf{x}_{gt}[1000]$ and R_B starting from $\mathbf{x}_B = \mathbf{x}_{gt}[1420]$ move to rendezvous point $\mathbf{x}_R = \mathbf{x}_{gt}[1200]$, as shown in Fig. 5.12. During this process, R_A and R_B build $\mathcal{D}_{M,A}$ and $\mathcal{D}_{M,B}$, as shown in Fig. 5.13(a) and (b), respectively. At $\mathbf{x}_R$, R_A receives $\mathcal{D}_{M,B}$ from R_B, and it regenerates $\mathcal{D}_{M,B}$ using its lattice and fuses it with $\mathcal{D}_{M,A}$. We evaluated the methods using the TSC NDT map $\mathcal{D}_{M,TSC}$, as shown in Fig. 5.14(a).

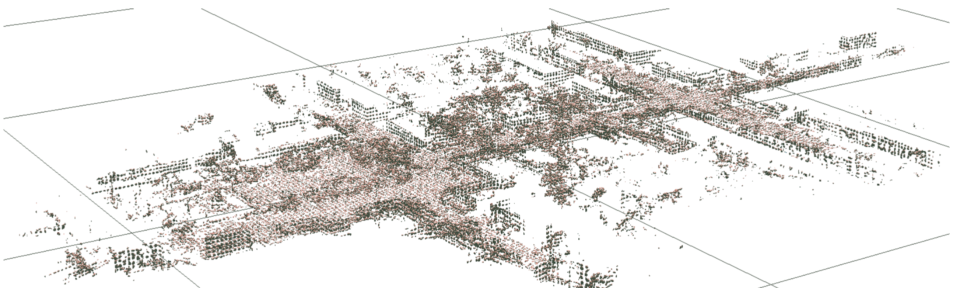
As a result, compared to the TSC NDT map in Fig. 5.15(a), the GCs are distorted by the conventional method, as shown in Fig. 5.15(b). On the other hand, compared to the GCs in Fig. 5.15(b), the presented method resulted in distinct GCs in the cells, as



(a)

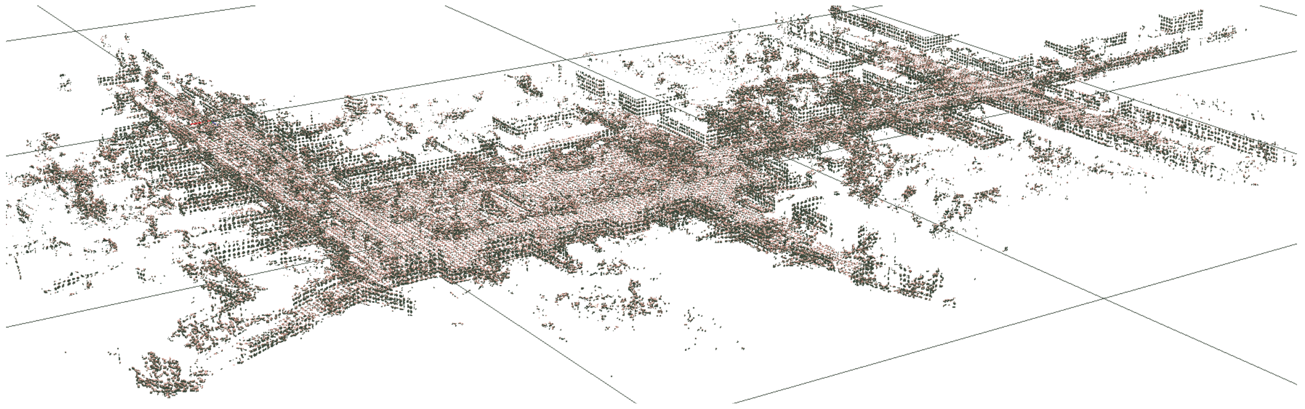


(b)

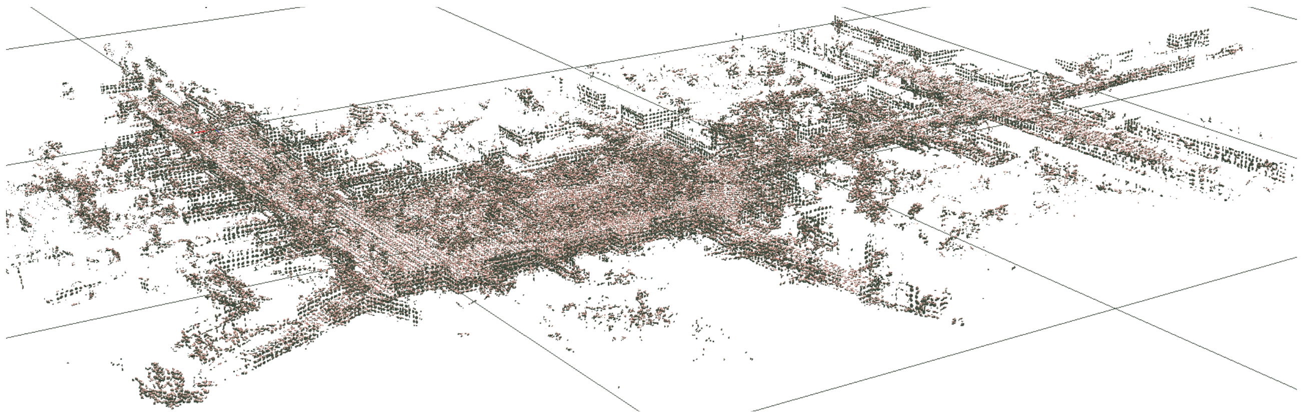


(c)

Figure 5.13: NDT maps built on the paths. (a) is the NDT map of robot A, (b) is the NDT map of robot B, and (c) is the regenerated NDT map of robot B.

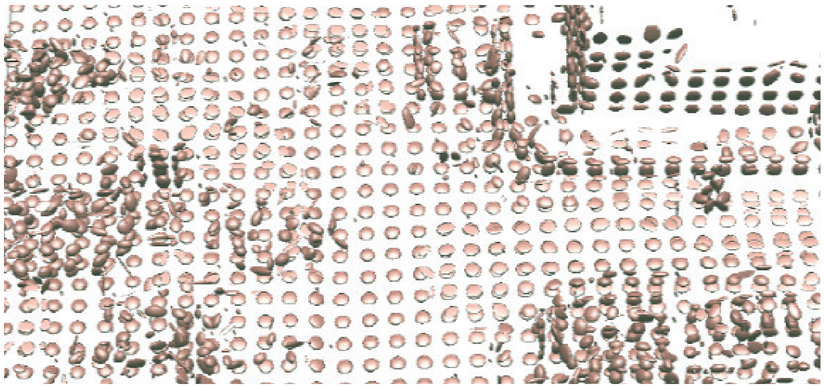


(a)

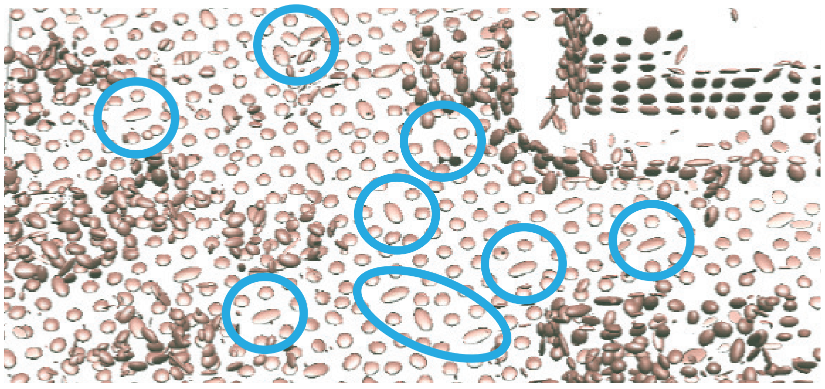


(b)

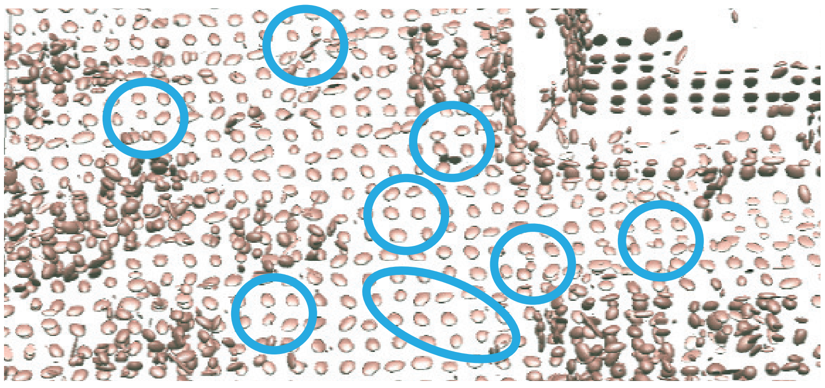
Figure 5.14: Fused NDT map and TSC NDT map: (a) fused NDT map, (b) TSC NDT map.



(a)



(b)



(c)

Figure 5.15: Fusion of the NDT maps: (a) TSC NDT map, (b) NDT map fused by the conventional method, (c) NDT map fused by the presented method.

Table 5.1: Performance of map fusion

method	conventional ($w_{th} = 0.5$)	presented ($w_{th} = 0.1$)
# of GC	49467	54137
d_{L_2}	0.9396	0.9500
d_m(m)	0.1141	0.0866
d_{cov}(m)	0.5707	0.5652
elapsed time(s)	0.422	0.824

shown in Fig. 5.15(c). Also, we evaluated the fused maps with the metrics in 5.4.1. As summarized in Table 5.1, d_{L_2} and d_{cov} are improved by about 2%, while d_m shows an improvement of approximately 24%. These outcomes indicate that the simple fusion causes large drifts of GC centers. However, because the presented method requires additional time for the subdivision of the GCs, the elapsed time is 1.95 times that needed by the conventional fusion process. To be specific, the presented method takes 0.824 seconds to subdivide 34816 GCs into 100150 truncated GCs and fuse the truncated GCs, while the conventional fusion takes 0.422 seconds to fuse $\mathcal{D}_{M,B}$ and $\mathcal{D}_{M,A}$.

5.5 Summary

In this chapter, we presente a method of regenerating an NDT fitted to a target lattice to improve the representation performance of a fused NDT map. The method subdivides the GCs of the NDT into truncated GCs sequentially on three axes. The truncated GCs located in the same cell are then fused into one GC to regenerate the NDT fitted to the target lattice. We evaluated the performance against a weight threshold using the lidar-based KITTI and Kinect-based Freiburg data sets. As a result, the ROC, accuracy, mean L_2 value, and mean error improved as the weight threshold was decreased, but they degenerated again as the weight threshold approaches zero because too low a weight threshold increases the false positive rate. On the other hand, the mean covariance distance for evaluating the covariance similarity decreases with a decrease in the weight threshold, with a lower weight threshold leading to a shorter covariance distance. In the case of map fusion application, we also found that the presented method improves the mean L_2 value, the mean error, and the mean covariance distance.

Chapter 6

Hue-Assisted Registration

6.1 Introduction

This chapter presents a colored NDT registration method. Due to the good extensibility of ICP, most of color-supported registration algorithms are based on ICP. The early algorithm is Colored ICP (cICP) [88]. Y, I, and Q from the YIQ color model were integrated to the distance function. Similarly, 4D ICP integrated the hue variable from the hue-saturation-lightness (HSL) model to the distance function [89]. Color-constrained ICP uses six classes corresponding to six intervals of the hue range. Points would be classified to classes according to hue. The points in the highest-scored class would be registered by ICP [90]. The modified Colored ICP quantizes I and Q to classify target points. The corresponding cell of a source point on the IQ plane is decided by I and Q of the source point, and it searches for the closest point in the cell [91]. Color-supported generalized ICP is a color-supported variant of the generalized ICP (gICP), which is a variant of ICP [92]. L*, a*, b* from the L*a*b* color model are integrated to the distance function of gICP. Also, there is a color-supported variant called Color

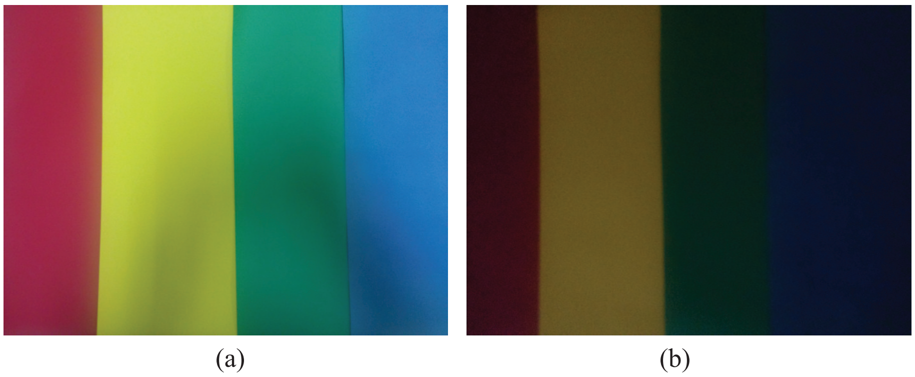


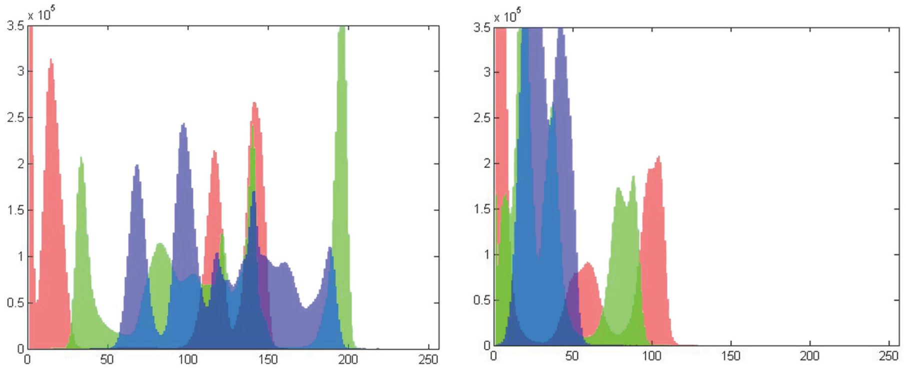
Figure 6.1: Colored papers taken with the high luminous intensity in (a) and low luminous intensity in (b).

NDT. Color NDT is improved by applying color weights to the objective function [93].

We utilized the hue from the hue-saturation-value (HSV) model because of its brightness-invariant and viewpoint-invariant properties. After the target point set is divided by the octree structure, the points in each cell are classified according to their hue values. In addition, the objective function of NDT was modified to be weighted by hue coefficients. In this chapter, the circular mean and variance of the hue are described.

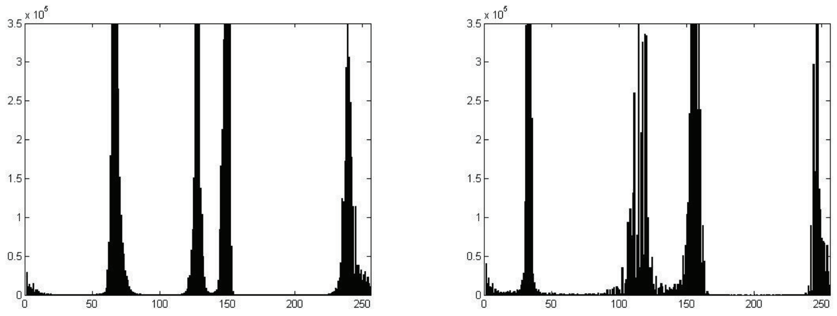
6.2 Preliminary of the HSV Model

Color information can be obtained by a vision sensor. The RGB values of an object are varying with the viewpoint of the sensor and the luminous intensity. On the contrary, the hue value from the HSV model is invariant to those effects. Two photographs are taken at different brightness levels as shown in Fig. 6.1. The distributions of RGB of photographs are as shown in Fig. 6.2. The distributions in Fig. 6.2(a) are widely distributed while those of Fig. 6.2(b) are concentrated at the left side. The hue distributions as shown in Fig. 6.3 are very close. Averages of four peaks in Fig. 6.3(a)



(a) (b)

Figure 6.2: RGB distributions of Fig. 6.1: R in red, G in green, and B in blue.



(a) (b)

Figure 6.3: Hue distributions of Fig. 6.1(a) and (b) respectively in (a) and (b).

are 32.58, 114.80, 155.34, and 246.88 while four averages in Fig. 6.3(b) are 66.79, 127.43, 149.30, and 240.26. Due to the brightness-invariant property of the hue, the hue is possible to support the registration of two colored point sets scanned at different brightness level.

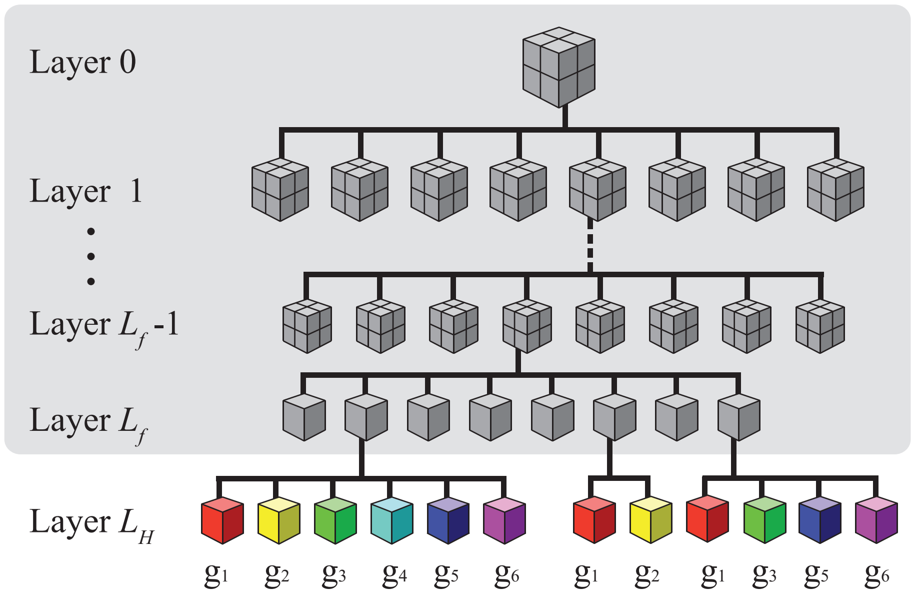


Figure 6.4: Colored octree. Tree in the gray box is the original octree.

6.3 Colored Octree for Subdivision

The colored octree structure is presented as shown in Fig. 6.4. The tree in the gray box is the conventional octree structure as introduced in Section 2.5. The structure in the box can be replaced with other structures, such as R-tree [94]. The colored octree additionally divides the leaves of the octree into at most n_{hue} color leaves which are corresponding to hue intervals. For example, in Fig. 6.4, since n_{hue} is 6, six intervals of the hue range are $[0, 1/6)$, $[1/6, 2/6)$, $[2/6, 3/6)$, $[3/6, 4/6)$, $[4/6, 5/6)$, $[5/6, 1)$. The kth target point $\mathbf{p}_{t,k}$, whose hue is in the jth interval, is classified into the jth hue group g_{ij} in the ith cell.

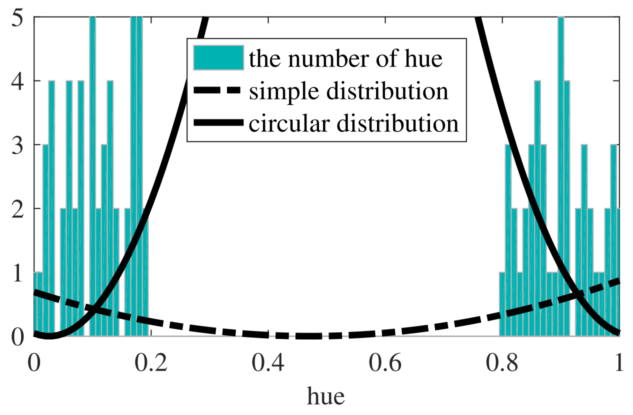


Figure 6.5: Comparison between the distribution of the simple mean and variance and the distribution of the circular mean and variance.

6.4 HA-NDT

We compute μ_{ij} and $\mathbf{\Sigma}_{ij}$ for g_{ij} by (2.3) and (2.4). Next, we computes the mean $\mu_{h,ij}$ and variance $\sigma_{h,ij}$ of the hue. Since the hue is circular, the simple mean and variance lead to the wrong distribution, as shown in 6.5. Therefore, HA-NDT uses the circular mean and variance of the hue. The circular mean μ_h is computed as

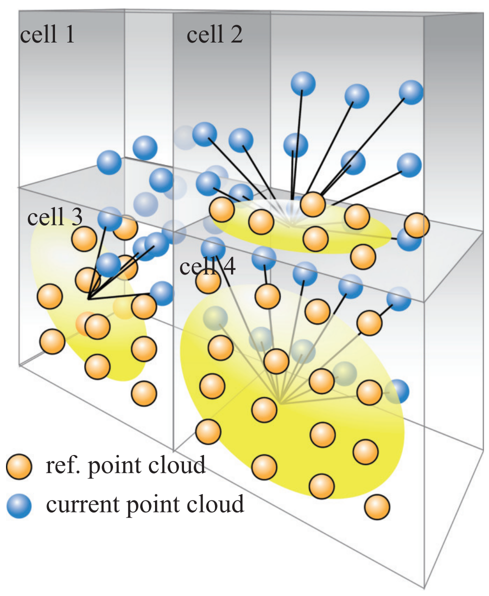
$$\mu_h = \frac{1}{2\pi} \arctan 2 \left(\sum_{j=1}^{n_{\mathcal{P}}} \sin(2\pi p_{t,j,h}), \sum_{j=1}^{n_{\mathcal{P}}} \cos(2\pi p_{t,j,h}) \right). \tag{6.1}$$

Due to the circular property of the hue, the distance between two hues should be in the range [0,0.5]. This dissertation defines the hue distance as

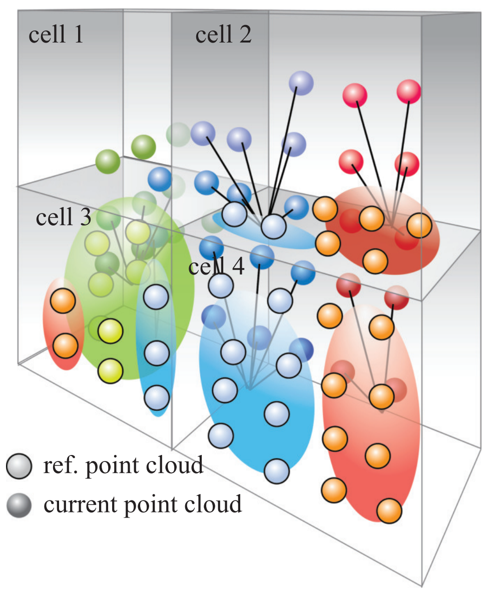
$$d_h(h_1, h_2) = \min(|h_1 - h_2|, 1 - |h_1 - h_2|), \tag{6.2}$$

and the circular variance σ_{cir} as

$$\sigma_h = \frac{1}{n_{\mathcal{P}}} \sum_{j=1}^{n_{\mathcal{P}}} d_h(p_{t,j,h}, \mu_h)^2. \tag{6.3}$$



(a)



(b)

Figure 6.6: Results of odometry estimation with cell size l=1m. Black odometry is ground truth, blue one is estimated by PNDT-D2D, and red one is estimated by PNDT-D2D-DSF. (a) 0th sequence. (b) 1st sequence. (c) 2nd sequence.

Using μ_h and σ_h, the weight w_h of a source point $\mathbf{p}_s$ and the target GC is computed as

$$w_h = \exp\left(-\frac{d_h(p_{s,h}, \mu_h)^2}{2\sigma_h}\right). \tag{6.4}$$

In contrary to the original NDT as shown in Fig. 6.6(a), there are at most n_{hue} GCs in a cell, as shown in Fig. 6.6(b).

HA-NDT registers $\mathcal{P}_s$ with $\mathcal{D}_t$. HA-NDT searches for the corresponding cell for $\mathbf{p}_s$. Next, it searches for a corresponding hue group. The group is determined by which interval its hue is in. The likelihood of observing $\mathbf{p}_s$ is weighted with w_h.

6.5 Experiments

The conventional NDT and HA-NDT are implemented by Point-Cloud-Library (PCL) [11]. We ran the implementation on Intel Core i7-3770. Data sets we use are from [12]. Data sets were recorded by Microsoft Kinect, and the ground-truth paths were captured with 100Hz tracking cameras. `'freiburg1_room'`, `'freiburg2_ desk'`, `'f-reiburg3_nostructure_texture_far'`, and `'freiburg3_structure_ texture_far'` are chosen.

First, the performance of the different n_{hue} of the HA-NDT are evaluated. Second, the performance of NDT and the HA-NDT are evaluated. The cell size is set to 5cm for the data set of the room and 10cm for other data sets, and the time stamp is 0.27 second. To avoid singular covariance, the geometric objective function of a hue group is available only if the number of points in the group is greater than 5. The performance are compared in terms of the iteration, runtime, translation error, and rotation error. The convergence criterion is the step length. If the length is smaller than a threshold which is 10^{-6}, then the algorithms terminate.

6.5.1 Evaluation of HA-NDT against n_{hue}

n_{hue}=1 is chosen to evaluate to check how the average of hue supports NDT. Due to three primary colors, n_{hue} starts with 3. Next, n_{hue} is doubled up to 24. The results are as shown in Table 6.1.

As shown in Table 6.1, the performance of the HA-NDT become better as n_{hue} increases until n_{hue}=12. Since the hue intervals become narrower as n_{hue} increases, the number of points in each hue group becomes smaller. Due to the constraint against singular covariance, the number of available groups decreases. Hence, the current scanned points are possible to be dropped because the corresponding groups are not available. Moreover, as n_{hue} increases, the hue intervals become narrower, and the hue of a current scanned point and the hue mean of the corresponding group become closer. It means that the point sets are possible to find more appropriate correspondences. Due to this reason, the errors become lower as n_{hue} increases. However, as n_{hue} is 24, the errors increase. The reason is that the algorithm drops too many current scanned points because the hue intervals are too narrow to be available. As the result shown in Table 6.1, the result of n_{hue}=12 is generally the best.

6.5.2 Evaluation of NDT and HA-NDT

In this experiment, n_{hue} of HA-NDT is set to 12. The results are as shown in Table 6.2. Plots in Fig. 6.7 show the convergence of algorithms. The point sets at 9116.45 and 9116.68 second in 'freiburg3˙structure˙texture' data set are used. The data is also used to show the objective function against the step length as shown in Fig. 6.8.

As the results shown in Table 6.2, although the runtime per iteration of the HA-NDT is shorter, the runtime is longer than NDT and HA-NDT. However, the translation and rotation errors of the HA-NDT are lower than those of NDT and HA-NDT. Fig. 6.7(a) shows the convergence of NDT. It shows that NDT would stop after the second iteration. Fig. 6.8(a) and (b) show the objective functions of NDT against the step length at the first and second iterations. The step length obtained at second iteration is approximately zero. If the step length is shorter than threshold, we regard that the registration is converged.

As shown in Table 6.2, the translation and rotation errors of HA-NDT using simple mean and variance and the HA-NDT are lower than NDT. The reason is that the target point set is more geometrically well-represented by distributions of hue groups. In

Table 6.1: Comparison between different n_{hue}.

dataset	n_{hue}	iterations	runtime(s)	error(m)	error(deg)
Freiburg1 room	1	4.965	19.695	0.119	5.130
	3	5.646	19.219	0.119	4.629
	6	5.439	14.918	0.119	5.192
	12	5.706	16.087	0.109	4.079
	24	6.127	16.344	0.116	4.492
Freiburg2 desk	1	5.181	35.196	0.077	1.637
	3	5.349	36.868	0.082	1.555
	6	5.222	29.842	0.079	1.496
	12	5.441	25.479	0.080	1.595
	24	5.846	32.859	0.081	1.501
Freiburg3 nostructure texture(far)	1	4.965	39.515	0.077	0.721
	3	4.929	39.426	0.077	0.695
	6	4.393	35.9	0.076	0.621
	12	4.729	34.821	0.079	0.678
	24	4.346	33.595	0.082	0.619
Freiburg3 nostructure texture(far)	1	4.548	36.431	0.058	0.901
	3	5.417	39.344	0.060	0.803
	6	4.833	36.092	0.060	0.824
	12	5.05	28.459	0.060	0.759
	24	5.717	40.636	0.059	0.711

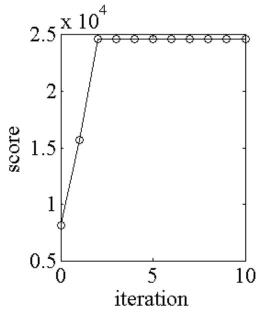
Table 6.2: Comparison Between Algorithms.

dataset	algorithm	iterations	runtime (s)	runtime per iteration (s)	error (m)	error (deg)
Freiburg1 room	NDT	3.915	16.421	4.194	0.178	4.394
	HA-NDT	5.706	16.087	2.819	0.109	4.079
Freiburg2 desk	NDT	2.876	22.774	7.919	0.075	1.231
	HA-NDT	5.05	28.46	5.636	0.060	0.759
Freiburg3 nostructure texture(far)	NDT	3.61	31.795	8.807	0.271	1.191
	HA-NDT	4.729	34.821	7.363	0.079	0.678
Freiburg3 structure texture(far)	NDT	3.62	24.392	6.738	0.135	1.912
	HA-NDT	5.441	25.479	4.683	0.080	1.595

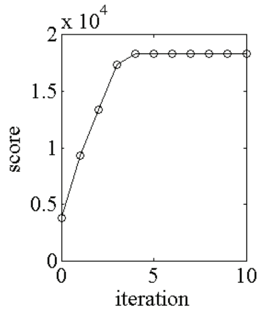
addition, the algorithms find better correspondences than NDT. However, the runtime of HA-NDT using simple mean and variance and the HA-NDT are longer than NDT. Fig. 6.7(b) and (c) show HA-NDT using simple mean and variance and the HA-NDT need more iterations until stop than NDT.

The HA-NDT performs the more accurate registration than HA-NDT using simple mean and variance. For a point far away from the mean of the cell, whose eigenvalues of the covariance matrix are small, the likelihood of the point computed by HA-NDT is significant while the likelihood computed by HA-NDT using simple mean and variance is approximately zero. It means that all of the current scanned points in the cell participate in objective function, first and second order partial derivatives. Due to this fact, the HA-NDT registers all of the current scanned points to target point set.

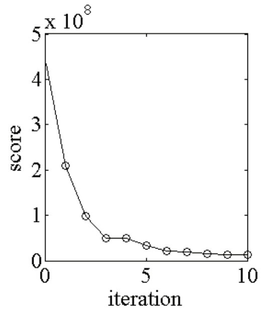
The initial guess of the step length of NDT or HA-NDT is set to one. As the graphs shown in Fig. 6.8(a) and (b), which are the objective function against the step length



(a)



(b)



(c)

Figure 6.7: The convergence of algorithms: (a) NDT, (b) HA-NDT using simple mean and variance, and (c) HA-NDT.

at the first and second iterations of NDT, the objective function increases as the step length decreases, so does HA-NDT as shown in Fig. 6.8(c) and (d). On the other hand, the initial guess of the improved HA-NDT is better not to be one because the step length obtained by Armijo's rule is usually very big as shown in Fig. 6.8(e), and it would lead to the failure of registration. In this experiment, the initial guess of the step length is chosen as $||H_k^{-1}g_k||$, which is much smaller than one. Since the initial guess usually satisfies Armijo's rule, the runtime per iteration of the HA-NDT is shorter than NDT-P2D and HA-NDT using simple mean and variance.

6.6 Summary

The performance of HA-NDT is evaluated by benchmark data sets. As the result, the accuracy of the registration is improved. The improvement is approached by changing the weighted objective function to objective function and the circular mean and hue variance. HA-NDT is expected to overcome the difficulty of the registration at flat or repeated structure by the hue.

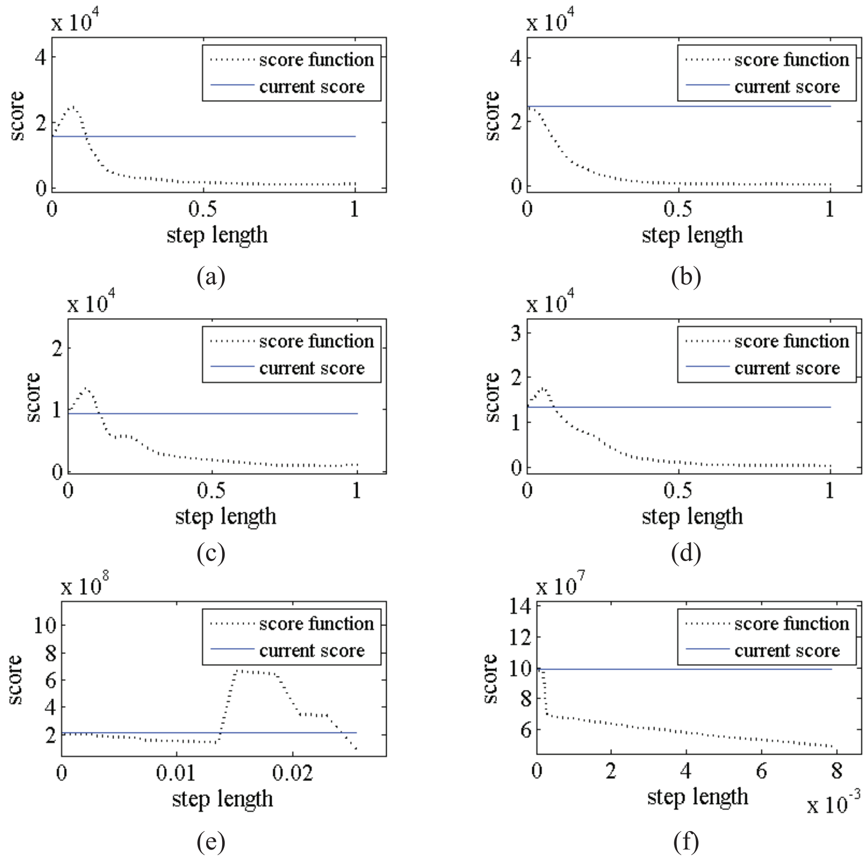


Figure 6.8: Score functions aginst step length. (a) and (b) are the objective functions of NDT plotted against the step length. (c) and (d) are those of HA-NDT using simple mean and variance. (e) and (f) are those of the HA-NDT. (a), (c), (e) are the objective functions of the 1st iterations, and (b), (d), (f) are those of the 2nd iterations.

Chapter 7

Key-Layered NDT Registration

7.1 Introduction

NDT needs regular cells to subdivide the target points into distributions. In fact, what determines the accuracy and speed of NDT is the cell size. To overcome the trade-off between the accuracy and runtime, the improved NDT registration which performs converging and adjusting stages was proposed [42]. This variant method creates multiple resolutions of cells to improve the registration. In the context of multiple resolution lattices, multi-layered NDT (ML-NDT) was proposed [60]. ML-NDT generates multiple layers which contain different resolutions of the lattices and registers the source point set to the target NDT layer by layer. The number of layers and iterations are fixed in the conventional ML-NDT. However, as shown in Fig. 7.1(a), rough GCs in the 5th layer does not represent the geometric structure of environments well. This roughness usually leads to the failure of registration. On the other hand, GCs in the 7th layer, as shown in Fig. 7.1(b), represent the structure better. Since the better representation results in the higher accuracy and higher success rate of registration, the number of

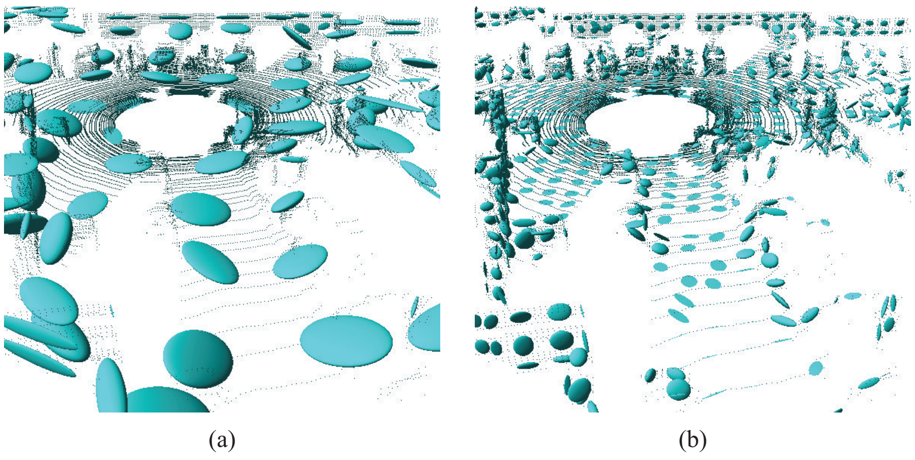


Figure 7.1: Rendering distributions of benchmark data set scanned outdoor: NDT in the 5th layer is in (a) and NDT in the 7th layer is in (b).

layers should not be limited. Another problem of ML-NDT is the fixed number of iterations. ML-NDT iterates three times of the registration in each layer except for the final layer. The registration in high layers are usually mismatched if it is mismatched in preceding layers. In addition, ML-NDT is not adaptive to the density and range of the point set which depend on the environment and the robot pose. Moreover, if the density of the point set is high enough to avoid the singularity of covariance in each cell, it can subdivide more times to represent accurately.

This chapter introduces a new key-layered NDT (KL-NDT) algorithm. The main contribution of KL-NDT is that it improves the accuracy and success rate by registering in key layers until it satisfies the condition of termination and skipping registrations in other layers.

7.2 Key-layered NDT-P2D

KL-NDT is a scan matching algorithm which registers the source point set $\mathcal{P}_s$ to the target point set $\mathcal{P}_t$ in multiple layers. For convenience, we call a cell containing at least one target point $\mathbf{p}_t$ as an activated cell and call a cell containing four or more $\mathbf{p}_t$ as a utilized cell. Also, we call a transformed source point $\mathbf{p}_T$ as an activated point if it is in an activated cell.

Compared to ML-NDT, KL-NDT iteratively executes subdividing process, generating process, and registering process as shown in Algorithm 1. First, it subdivides cells by *Keylayer*$(\mathcal{V})$ to get into the next key layer. Second, it computes the mean and covariance for the partial point set in each cell by *Generate*$(\boldsymbol{P}, \mathcal{V})$. Finally, it computes the pose update by *Register* $(\boldsymbol{P}, \mathcal{V}, \mathbf{D}, \boldsymbol{\Theta})$. Before the iteration, it obtains the initial cell, a cube covering both $\mathcal{P}_t$ and $\mathcal{P}_s$, by *Range*($\mathcal{P}_t$,$\mathcal{P}_s$). It terminates the iteration when the pose update is relatively small to the estimated pose variation, the distributions become degenerated, or the number of correspondence is fewer than the threshold.

A unique feature of KL-NDT is to register point sets in key layers. KL-NDT generates distribution set and registers point sets by the cell set $\mathcal{V}_k$ which is obtained by *Keylayer*$(\mathcal{V}_{k-1})$. As shown in Fig. 7.2, it skips generations and registrations in the initial layer L_0 and L_1. The first generation and registration are executed in L_2, which is the first key layer.

The key layer is determined by $\frac{n_{in,j+1}}{n_{in,j}}$, which is the ratio of the numbers of activated points in the jth and (j+1)th layers. The function $Keylayer(\mathcal{V}_{k-1})$ recursively subdivides $\mathcal{V}_j$ in L_j and creates $\mathcal{V}_{j+1}$ for L_{j+1} until the ratio is not bigger than threshold value τ_{kl}, which is close to 1. In Fig. 7.2, white squares are activated cells,

Algorithm 1 KLNDT

```
Require: target point set $\mathcal{P}_t$, source point set $\mathcal{P}_s$
Ensure: transformation vector $\mathbf{\Theta}$
1: the initial cell set $\mathcal{V}_0 = \mathit{Range}(\mathcal{P}_t, \mathcal{P}_s)$; $\mathbf{\Theta}_0 = \mathbf{0}$; $k \leftarrow 0$
2: repeat
3:     $k \leftarrow k + 1$
4:     the $k$th cell set $\mathcal{V}_k = \mathit{Keylayer}(\mathcal{V}_{k-1})$
5:     the NDT $\mathcal{D}_k = \mathit{Generate}(\mathcal{P}_t, \mathcal{V}_k)$
6:     $\mathbf{\Theta}_k = \mathit{Register}(\mathcal{P}_s, \mathcal{V}_k, \mathcal{D}_k, \mathbf{\Theta}_{k-1})$
7: until $\frac{\|\mathbf{\Theta}_k - \mathbf{\Theta}_{k-1}\|}{\|\mathbf{\Theta}_{k-1}\|} < \tau$ or $\mathit{degenerate}(\mathcal{D}_k, \mathcal{P}_s)$
8: return $\mathbf{\Theta}_k$
```

blue boxes are $\mathbf{p}_t$, red balls are activated $\mathbf{p}_s$, and red-lined balls are inactivated $\mathbf{p}_s$. $Keylayer(\mathcal{V}_{k-1})$ skips registrations in L_0 and L_1 since $\frac{n_{in,1}}{n_{in,0}}$ and $\frac{n_{in,2}}{n_{in,1}}$ are bigger than τ_{kl}. However, since $n_{in,3}$ is much smaller than $n_{in,2}$, the ratio becomes smaller than τ_{kl}. Thus, $Keylayer(\mathcal{V}_{k-1})$ terminates subdivision, and the cell set $\mathcal{V}_k$ in the kth key layer is obtained.

When it comes to NDT-D2D registration, the number of activated source point is computed differently. First, the module *Keylayer* extracts the GCs in the fine layer L_f. Second, it inputs the GCs and check whether the GCs are in the activated cells, as shown in Fig. 7.2. Third, it obtains the number of points which is corresponding to a activated source GC. Finally, it sums up the numbers and obtains n_{in}.

KL-NDT inherits the objective function of ML-NDT to compare with ML-NDT, which is a half of the square of Mahalanobis distance as

$$s(\boldsymbol{p}) = \frac{1}{2}(\mathbf{p} - \boldsymbol{\mu})^T \mathbf{\Sigma}^{-1}(\mathbf{p} - \boldsymbol{\mu}), \tag{7.1}$$

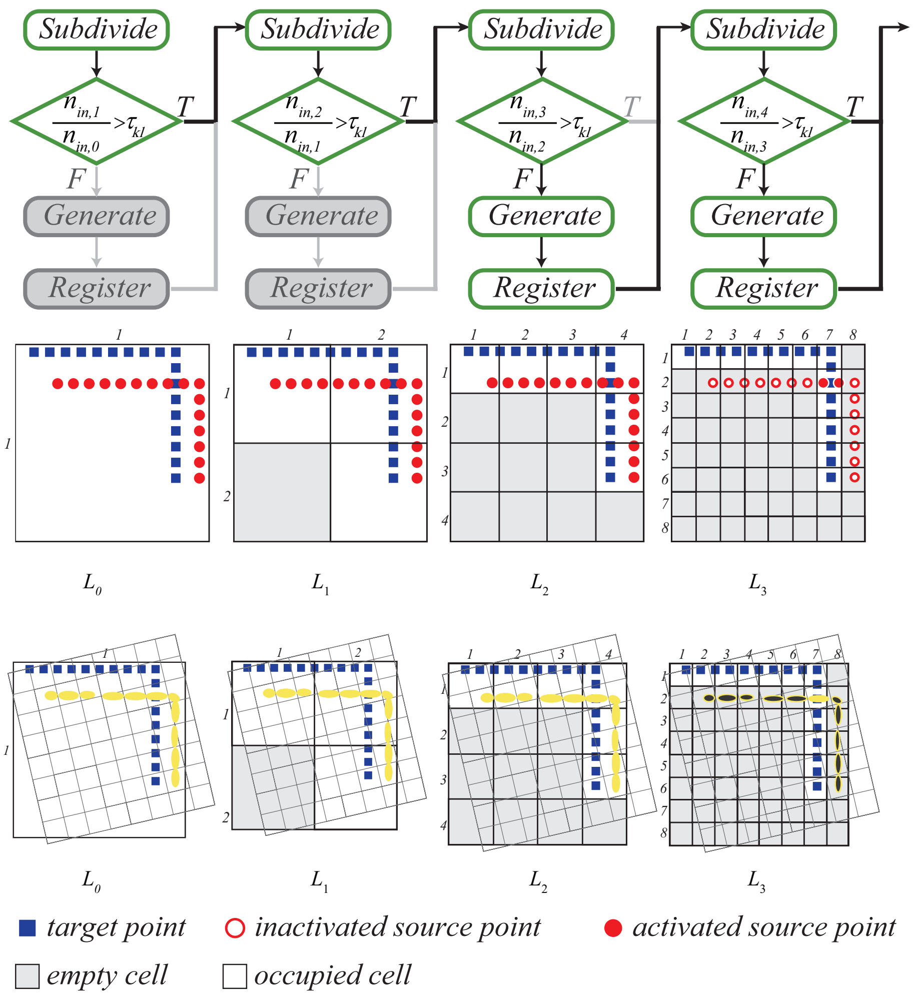

Subdivide
$\frac{n_{in,1}}{n_{in,0}} > \tau_{kl}$
$\frac{n_{in,2}}{n_{in,1}} > \tau_{kl}$
$\frac{n_{in,3}}{n_{in,2}} > \tau_{kl}$
$\frac{n_{in,4}}{n_{in,3}} > \tau_{kl}$
T
F
Generate
Register
L_0
L_1
L_2
L_3
target point
inactivated source point
activated source point
empty cell
occupied cell

Figure 7.2: The process of searching for key layers and registering in key layers.

where $s(\mathbf{p})$ is the objective function of $\mathbf{p}$, $\boldsymbol{\mu}$ is the mean vector, and $\boldsymbol{\Sigma}$ is the covariance matrix. The function *Generate*$(\mathcal{P}_t, \mathcal{V}_k)$ in Algorithm 1 computes $\boldsymbol{m}$ and $\boldsymbol{\Sigma}$ of partial point set in each activated cell, and it returns the NDT $\mathcal{D}_k$. In each key layer, KL-NDT registers two point sets by the Newton method. The function *Register* $(\mathcal{P}_s, \mathcal{V}_k, \mathcal{D}_k, \boldsymbol{\Theta}_{k-1})$ in Algorithm 1 updates $\boldsymbol{\Theta}$ until γ is smaller than the threshold value.

On the other hand, this dissertation applies the multi-layer approach to NDT-D2D registration. In this case, different from the point-to-distribution KL-NDT and ML-NDT, we use the objective function of NDT-D2D to the multi-layered NDT-D2D (ML-NDT-D2D) and the key-layered NDT-D2D (KL-NDT-D2D).

7.3 Experiments

7.3.1 Evaluation of KL-NDT-P2D and ML-NDT-P2D

This experiment compares the success rate and accuracy between KL-NDT and ML-NDT. First, the convergence of the objective function and pose estimation is compared. Second, the success rate and the accuracy are compared. We use a benchmark data set which is scanned by the Velodyne three-dimensional range finder [95]. The parameters of KL-NDT such as τ_{it} and τ_{kl} are initially set. Also, those of 4-layered ML-NDT and 6-layered ML-NDT are set. The registration of ML-NDT iterates 3 times in each preceding layer and iterates until γ is smaller than the threshold value in the final layer.

In the experiment, $\mathcal{P}_s$ is generated by rotating $\mathcal{P}_t$ 5 degrees about the yaw axis. As shown in Fig. 7.3, 4-layered NDT, 6-layered NDT, and KL-NDT successfully register the point sets. The convergence of the objective function of KL-NDT and ML-NDT is shown in Fig. 7.3(a) and Fig. 7.3(b), respectively. KL-NDT skips layers and begins in the 4th layer and registers only in the 4th and 8th layers. On the other hand, 4-

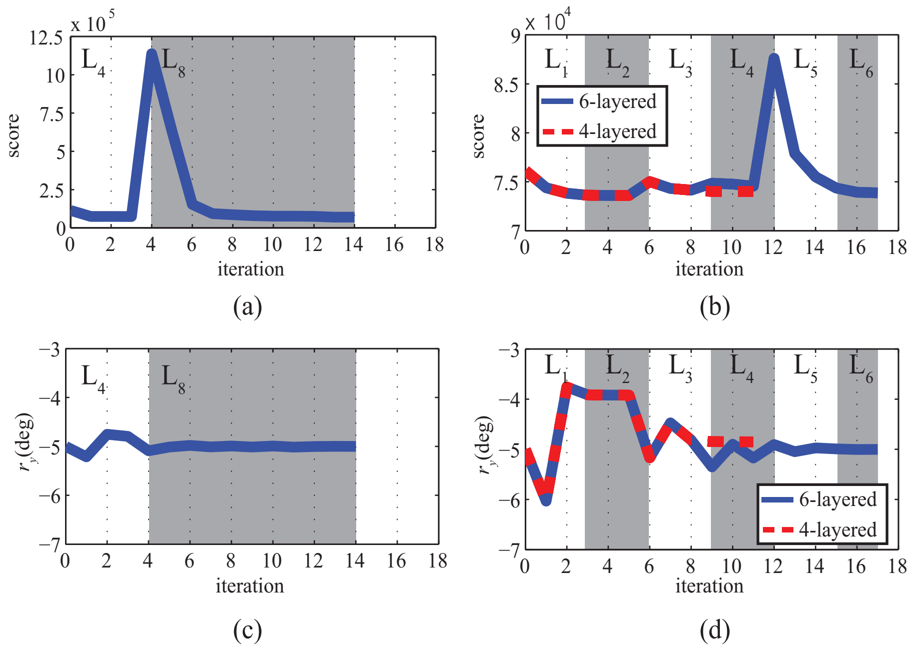


Figure 7.3: Convergence of the objective function and r_y about yaw axis: the objective functions of KL-NDT and ML-NDT are in (a) and (b) respectively, and plots of r_y of KL-NDT and ML-NDT are in (c) and (d) respectively.

layered ML-NDT and 6-layered ML-NDT begin in the 1st layer and register layer by layer. The convergence of the estimated yaw angles r_y of KL-NDT and ML-NDT is shown in Fig. 7.3(c) and Fig. 7.3(d), respectively. As shown in Table 7.1, r_y estimated by 6-layered ML-NDT is closer to 5 than that of 4-layered ML-NDT. In addition, translational transformation estimated by 6-layered ML-NDT is closer to 0 than that of 4-layered ML-NDT. On the other hand, r_y estimated by KL-NDT is closer to 5 than that by 6-layered ML-NDT, and the translational transformation is closer to 0. Also, ten scans are extracted from data set and set to be $\mathcal{P}_t$. Each $\mathcal{P}_s$ is generated by transforming each $\mathcal{P}_t$. The translational variations are set from -6 to +6 meters at intervals of 2 meters on pitch and roll axes, respectively, and the rotational variations

Table 7.1: Errors of the pose estimation.

algorithm	d(mm)	r_x(deg)	r_y(deg)	r_z(deg)
KL-NDT	0.985	2.71e-4	5.001	7.79e-4
ML-NDT(4)	25.266	-0.064	4.851	-0.051
ML-NDT(6)	18.836	2.60e-4	5.003	6.75e-4

Table 7.2: Success rates and accuracy of algorithms.

algorithm	success rate(%)	average of error	
		d (m)	r_y (deg)
KL-NDT	73.99	1.072	0.0432
ML-NDT(4)	68.67	1.437	0.1217
ML-NDT(6)	68.79	1.316	0.1144

are set from -50 to +50 degrees at intervals of 5 degrees about the yaw axis. The results of success rates are shown in Table 7.3. The success rates of 4-layered ML-NDT and 6-layered ML-NDT are similar. However, the error of pose variation estimated by 6-layered ML-NDT is lower than that by 4-layered ML-NDT. It shows that the accuracy of ML-NDT is limited if the number of layers is small. On the other hand, the success rate of KL-NDT is higher than ML-NDT since it registers point sets in key layers. Moreover, its errors of r_y and d are much smaller than those of ML-NDT since KL-NDT is available for registration in higher layers.

Table 7.3: Errors of the pose estimated by ML-NDT-D2D.

L_c	$e_t(\%)$	$e_r(10^{-3}deg/m)$	t (sec)
2	2.534	7.839	1.208
3	2.534	7.848	1.072
4	2.533	7.836	0.986
5	2.528	7.847	0.935
6	2.527	7.802	0.926
7	2.534	7.823	0.918

7.3.2 Evaluation of KL-NDT-D2D and ML-NDT-D2D

In this experiment, we evaluated the NDT-D2D which registers in the multiple layers using the KITTI sequences 0 to 10, and the cell size in the fine layer $L_f = 8$ is set to 1 m. The initial guess is set to the preceding pose variation.

First, we conducted experiments which begin registrations in the different coarse layers, where $L_c \in \{2, 3, 4, 5, 6, 7\}$. Registration in each layer iterates at most three times, and the registration in the fine layer iterates until the transformation converges. Second, we conducted experiments of KL-NDT-D2D with $\tau_{kl} \in \{0.985, 0.990, 0.995, 0.999\}$. The results of ML-NDT-D2D are summarized in Table 7.3, and the results of KL-NDT-D2D are summarized in Table 7.4. As a result, KL-NDT-D2D and ML-NDT-D2D show the similar registration performance. However, KL-NDT-D2D with τ_{kl} equal to 0.99 and 0.985 shows the faster registration compared to ML-NDT-D2D. The accuracy decreases as L_c increases because we use initial guess and the pose variation to estimate becomes smaller than the case without the initial guess.

Table 7.4: Errors of the pose estimated by KL-NDT-D2D.

τ_{kl}	$e_t(\%)$	$e_r(10^{-3}deg/m)$	t (sec)
0.999	2.534	7.845	1.118
0.995	2.526	7.834	1.090
0.99	2.529	7.838	0.877
0.985	2.531	7.831	0.867

7.4 Summary

This chapter presents a new key-layered normal distributions transform algorithm. The number of layers and the number of iterations per layer are not fixed in the presented KL-NDT. The method of searching for the key layer is introduced, and the performance is demonstrated by the experiment. Also, the terminating criteria of the algorithm and the registration in each layer are presented. The higher success rates and the lower errors of KL-NDT are verified compared to ML-NDT from the experiment.

Chapter 8

Scaled NDT and The Multi-scale Registration

8.1 Introduction

In this chapter, a registration improved by dynamic scaling factors (DSF) is presented. Two scaling factors s_t and s_s of covariances are defined for $\mathcal{D}_t$ and $\mathcal{D}_s$, respectively. The presented registration is based on NDT-D2D, and it can be divided into three steps. First is NDT-P2D-like registration which can register without the negative correlation between L_2 distance and rotational alignment. Second is NDT-D2D with decreasing s_s and s_t to roughly obtain the transformation. Third is NDT-D2D with fixed s_s and s_t determined by the translation vector. In experiments, we compared NDT-D2D and PNDT-D2D improved by the presented method. As a result, the presented DSF improves the registrations.

Although there are numerous approaches to smooth the NDT registration as introduced in 1.3.1, we still have to determine the regularizing factors experimentally. NDT-D2D has another issue caused by the objective function. Generally, L_2 likelihood is expected to increase as the point sets are aligned. However, it is negatively corre-

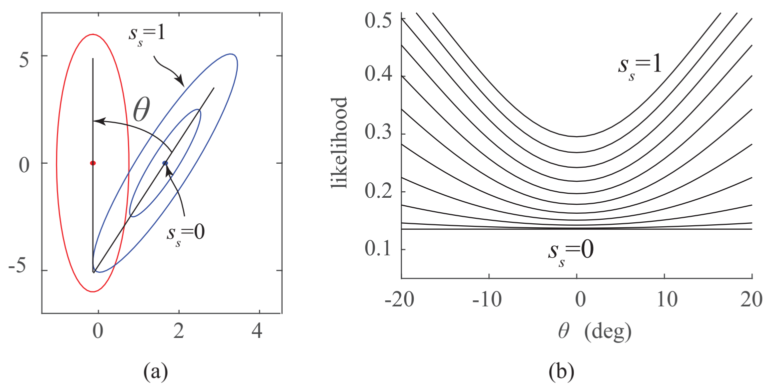


Figure 8.1: Negative correlation between L_2 likelihood and rotational alignment. (a) red: 1σ of the target covariance, blue: 1σ of the source covariance rotated with θ and scaled with s_s. (b) L_2 likelihood between two distributions against θ with different s_s.

lated to the rotational alignment at a location. Given $\mathcal{D}_t$ in red and $\mathcal{D}_s$ rotated with θ in blue as shown in Fig. 8.1(a), we can have the graph of L_2 likelihood against θ in Fig. 8.1(b). As a result, although it is rotationally aligned as $|\theta|$ decreases, the L_2 likelihood decreases as shown in Fig. 8.1(b).

To overcome the problem, we suggest to initialize s_s to 0 and gradually increase it. In Fig. 8.1(b), the L_2 likelihood against θ with different s_s from 1 to 0 can be seen. As s_s decreases, the changes caused by θ decreases, and if $s_s = 0$, it is degenerated as NDT-P2D of $\mathcal{D}_t$ and mean vectors of $\mathcal{D}_s$.

8.2 Scaled NDT representation and L_2 distance

NDT-D2D with dynamic scaling factors (NDT-D2D-DSF) is based on the scaled NDT, whose covariance matrices are scaled by a positive scaling factor s. We define a scaled covariance $\mathbf{\Sigma}_s$ as

$$\mathbf{\Sigma}_s = s\mathbf{\Sigma}. \tag{8.1}$$

It is equivalent to scale the eigenvalue of $\mathbf{\Sigma}$ with s. The eigenvector matrix V and eigenvalue matrix D of $\mathbf{\Sigma}$ obtained by eigenvalue decomposition can be expressed as

$$\mathbf{\Sigma} = VDV^T, \tag{8.2}$$

$$D = \begin{bmatrix} \sigma_1^2 & 0 & 0 \\ 0 & \sigma_2^2 & 0 \\ 0 & 0 & \sigma_3^2 \end{bmatrix}, \tag{8.3}$$

where σ_i^2 is the variance on ith eigenvector, and the scaled covariance $\mathbf{\Sigma}_s$ can be derived as

$$\mathbf{\Sigma}_s = s\mathbf{\Sigma} = V(sD)V^T = VD_sV^T. \tag{8.4}$$

Therefore, as s increases, the variance is swelled and the NDT is smoothed as shown in Fig. 8.2.

We define two individual scaling factors s_s and s_t for $\mathcal{D}_t$ and $\mathcal{D}_s$, respectively, and we modify (2.11) to

$$\mathbf{\Sigma}_{s,ij} = s_s R\mathbf{\Sigma}_i R^T + s_t\mathbf{\Sigma}_j, \tag{8.5}$$

and d_2 can be seen as the special case of $s_s = s_t = 1/d_2$. Thus, the presented objective function can be expressed as

$$f_s(\mathcal{D}_t, \mathcal{D}_s, \mathbf{\Theta}) = \sum_{i=1}^{n_s}\sum_{j=1}^{n_t} d_{s,ij}, \tag{8.6}$$

$$d_{s,ij} = -d_1 \exp\Big(-\frac{1}{2}g_{s,ij}\Big), \tag{8.7}$$

$$g_{s,ij} = \boldsymbol{\mu}_{ij}^T\mathbf{\Sigma}_{s,ij}^{-1}\boldsymbol{\mu}_{ij}. \tag{8.8}$$

8.3 NDT-D2D with dynamic scaling factors of covariances

The key idea of the presented method is varying scaling factors s_t and s_s in each iteration to improve the accuracy of the NDT-D2D. To decrease and increase the scaling factors, we define a function as follows:

$$s(k) = S(k, k_i, k_f, s_i, s_f), \tag{8.9}$$

where k_i and k_f are the initial and final iterations, respectively, and s_i and s_f are the initial and final scales, respectively.

At the first, since $\mathcal{D}_s$ is not rotationally aligned to $\mathcal{D}_t$, the negative correlation problem might exist. Therefore, the strategy is to initialize s_s to 0 so that it becomes the NDT-P2D between $\mathcal{D}_t$ and mean vectors of $\mathcal{D}_s$. Also, to attract those mean vectors of $\mathcal{D}_s$, $\mathcal{D}_t$ is swelled with $s_t = s_{max}$. After this initialization, the registration performs the following processes in order: scaling NDTs to avoid the negative correlation between L_2 distance and rotational alignment, shrinking NDTs with the same scaling factors, and scaling NDTs considering the translation.

In the first process, since $\mathcal{D}_s$ can be expected to be rotationally aligned to $\mathcal{D}_t$ as the registration iterates, s_t can gradually decrease from s_{max} to s_1 shrink $\mathcal{D}_t$ to regain the shape by $S(k, 0, k_1, s_{max}, s_1)$ and s_s can gradually increase from 0 to s_1 show the shape of $\mathcal{D}_s$ by $S(k, 0, k_1, 0, s_1))$. The process continues until $s_s = s_t$ in k_1th iteration as shown in Fig. 8.3.

In the second process, s_s and s_t decrease as the same function $S(k, k_1, k_2, s_1,$ $s_{tran}(k))$ until $s_{new} = s_{tran}$ in k_2th iteration. Since $s_s = s_t$, $\mathbf{\Sigma}_i$ and $\mathbf{\Sigma}_j$ have the same influence on the cost, gradient, and Hessian matrix of the objective function. Also, as the scaling factors decrease to s_{tran}, $\mathcal{D}_t$ and $\mathcal{D}_s$ are shrunken to regain their shapes so that the registration can be expected to estimate the more accurate transformation. If

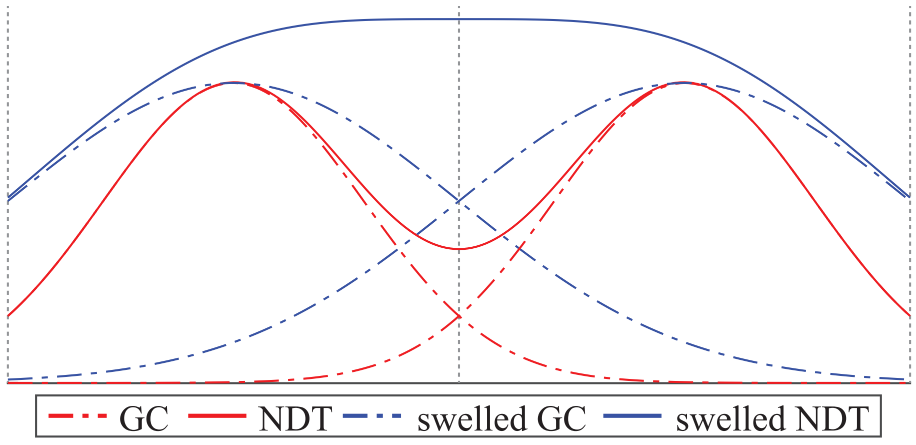


Figure 8.2: Illustration of the scaled NDT.

s_{tran} is greater than s_{min}, s_t would be equal to s_{new}. However, since the local minima of the objective function can appear again due to the small s_{tran}, we ask s_t to be at least s_{min} to avoid the local minima.

In the final process, $\mathcal{D}_s$ registers to $\mathcal{D}_t$ with fixed $s_s = s_{tran}$ and s_t. If $s_{tran} > s_{min}$, s_t is equal to s_{tran} as the case (a) in Fig. 8.3, otherwise s_t is equal to s_{min} as the case (b) in Fig. 8.3. The process terminates if the condition of the termination is satisfied.

The properties of the processes can be seen in the optimization process. In this chapter, we choose Newton method as an example. To estimate Θ by minimizing (8.6), it updates Θ iteratively by (2.28) and (2.29). The gradient and Hessian matrix of (8.6) are derived as follows.

At the initial state which $s_t = s_{max}$ and $s_s = 0$, we have the gradient and Hessian of g_{ij} as follows:

$$\frac{\partial g_{s,ij}}{\partial \theta_a} = \frac{2}{s_t} \boldsymbol{\mu}_{ij}^T \boldsymbol{\Sigma}_i^{-1} \frac{\partial \boldsymbol{\mu}_{ij}}{\partial \theta_a}, \tag{8.10}$$

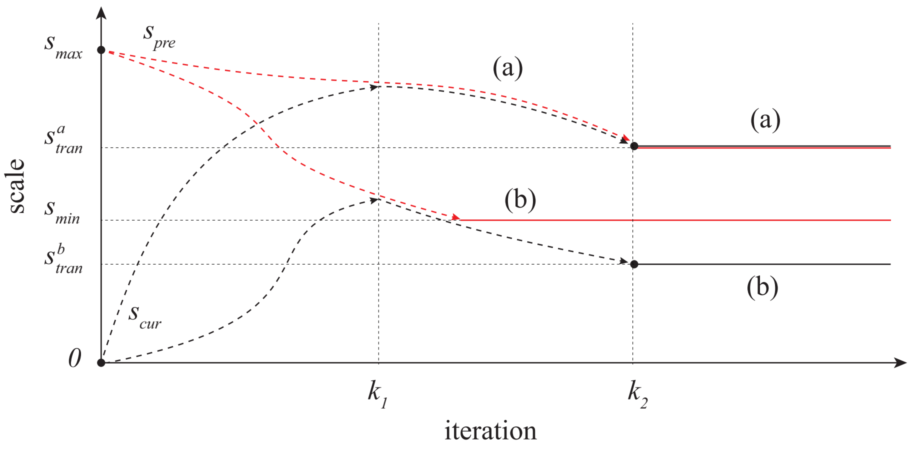


Figure 8.3: Illustration of scaling factors varying with iteration.

$$\frac{\partial^2 g_{s,ij}}{\partial\theta_a\partial\theta_b} = \frac{2}{s_t}\left(\boldsymbol{\mu}_{ij}^T\boldsymbol{\Sigma}_i^{-1}\frac{\partial^2\boldsymbol{\mu}_{ij}}{\partial\theta_a\partial\theta_b} + \frac{\partial\boldsymbol{\mu}_{ij}^T}{\partial\theta_a}\boldsymbol{\Sigma}_i^{-T}\frac{\partial\boldsymbol{\mu}_{ij}}{\partial\theta_b}\right). \tag{8.11}$$

(2.29) can be derived as

$$\Delta\Theta_t = -\gamma\Big(\sum_{i,j} d_{s,ij}\big(g_{ij}^{(2)} - \frac{1}{s_t}g_{ij}^{(1)^T}g_{ij}^{(1)}\big)\Big)^{-1}\cdot\Big(\sum_{i,j} d_{s,ij}g_{ij}^{(1)}\Big)^T, \tag{8.12}$$

where g_{ij} is (2.40). This equation leads to the same update vector obtained by NDT-P2D of $\mathcal{D}_t$ scaled by s_t and the mean vectors of $\mathcal{D}_s$. As s_t decreases, the influence of $g_{ij}^{(2)}$ decreases, and d_{ij} also decreases. If s_t is so large that all of d_{ij} are approximately 1, $\Delta\Theta_k$ is

$$\Delta\Theta_t = -\gamma\Big(\sum_{i,j} g_{ij}^{(2)}\Big)^{-1}\cdot\Big(\sum_{i,j} g_{ij}^{(1)}\Big)^T. \tag{8.13}$$

It is the upper bound of $\Delta\Theta_k$ against s_t, and it is exactly the same update vector obtained by NDT-P2D based on Mahalanobis distance [60].

In the first process, the first and second derivatives of $\boldsymbol{\Sigma}_{s,ij}^{-1}$ increases as s_s increases. It means that the registration not only aligns the mean vectors of $\mathcal{D}_s$ to $\mathcal{D}_t$ but also aligns the shapes of GCs. In the second and third processes, as s_t and s_s decrease,

$d_{s,ij}$ gap increases. Since $d_{s,ij}$ in (8.14) and (8.17) acts like a weight, the pair of GCs having larger L_2 likelihood has more influence on computing update vector.

The gradient vector of (8.6) can be derived as follows:

$$\frac{\partial f_s(\mathcal{D}_t, \mathcal{D}_s, \mathbf{\Theta})}{\partial \theta_a} = \sum_{i=1}^{n_s}\sum_{j=1}^{n_t} -\frac{1}{2} d_{s,ij} \frac{\partial g_{s,ij}}{\partial \theta_a}, \tag{8.14}$$

$$\frac{\partial g_{s,ij}}{\partial \theta_a} = 2\boldsymbol{\mu}_{ij}^T \mathbf{\Sigma}_{s,ij}^{-1} \frac{\partial \boldsymbol{\mu}_{ij}}{\partial \theta_a} + \boldsymbol{\mu}_{ij}^T \frac{\partial (\mathbf{\Sigma}_{s,ij}^{-1})}{\partial \theta_a} \boldsymbol{\mu}_{ij}, \tag{8.15}$$

$$\frac{\partial (\mathbf{\Sigma}_{s,ij}^{-1})}{\partial \theta_a} = -s_s \mathbf{\Sigma}_{s,ij}^{-1} \left(R\mathbf{\Sigma}_i \frac{\partial R^T}{\partial \theta_a} + \frac{\partial R}{\partial \theta_a} \mathbf{\Sigma}_i R^T \right) \mathbf{\Sigma}_{s,ij}^{-1}. \tag{8.16}$$

Also, the Hessian matrix can be derived as follows:

$$\frac{\partial^2 f_s(\mathcal{D}_t, \mathcal{D}_s, \mathbf{\Theta})}{\partial \theta_a \partial \theta_b} = \sum_{i=1}^{n_s}\sum_{j=1}^{n_t} \frac{\partial^2 d_{s,ij}}{\partial \theta_a \partial \theta_b}, \tag{8.17}$$

$$\frac{\partial^2 d_{s,ij}}{\partial \theta_a \partial \theta_b} = -\frac{1}{2} d_{s,ij} \left(\frac{\partial^2 g_{s,ij}}{\partial \theta_a \partial \theta_b} - \frac{1}{2} \frac{\partial g_{s,ij}^T}{\partial \theta_a} \frac{\partial g_{s,ij}}{\partial \theta_b} \right), \tag{8.18}$$

$$\begin{aligned}\frac{\partial^2 g_{s,ij}}{\partial \theta_a \partial \theta_b} &= 2\boldsymbol{\mu}_{ij}^T \mathbf{\Sigma}_{s,ij}^{-1} \frac{\partial^2 \boldsymbol{\mu}_{ij}}{\partial \theta_a \partial \theta_b} + 2 \frac{\partial \boldsymbol{\mu}_{ij}}{\partial \theta_a}^T \mathbf{\Sigma}_{s,ij}^{-1} \frac{\partial \boldsymbol{\mu}_{ij}}{\partial \theta_b} \\ &\quad + 2\boldsymbol{\mu}_{ij}^T \frac{\partial (\mathbf{\Sigma}_{s,ij}^{-1})}{\partial \theta_b} \frac{\partial \boldsymbol{\mu}_{ij}}{\partial \theta_a} + 2\boldsymbol{\mu}_{ij}^T \frac{\partial (\mathbf{\Sigma}_{s,ij}^{-1})}{\partial \theta_a} \frac{\partial \boldsymbol{\mu}_{ij}}{\partial \theta_b} \\ &\quad + \boldsymbol{\mu}_{ij}^T \frac{\partial^2 (\mathbf{\Sigma}_{s,ij}^{-1})}{\partial \theta_a \partial \theta_b} \boldsymbol{\mu}_{ij},\end{aligned} \tag{8.19}$$

$$\begin{aligned}\frac{\partial^2 (\mathbf{\Sigma}_{s,ij}^{-1})}{\partial \theta_a \partial \theta_b} &= -s_s \mathbf{\Sigma}_{s,ij}^{-1} \left(R\mathbf{\Sigma}_i \frac{\partial R}{\partial \theta_a}^T + \frac{\partial R}{\partial \theta_a} \mathbf{\Sigma}_i R^T \right) \frac{\partial (\mathbf{\Sigma}_{s,ij}^{-1})}{\partial \theta_b} \\ &\quad - s_s \frac{\partial (\mathbf{\Sigma}_{s,ij}^{-1})}{\partial \theta_b} \left(R\mathbf{\Sigma}_i \frac{\partial R}{\partial \theta_a}^T + \frac{\partial R}{\partial \theta_a} \mathbf{\Sigma}_i R^T \right) \mathbf{\Sigma}_{s,ij}^{-1} \\ &\quad - s_s \mathbf{\Sigma}_{s,ij}^{-1} \left(\frac{\partial R}{\partial \theta_a} \mathbf{\Sigma}_i \frac{\partial R^T}{\partial \theta_b} + \frac{\partial^2 R}{\partial \theta_a \partial \theta_b} \mathbf{\Sigma}_i R^T \right. \\ &\quad \left. + \frac{\partial R}{\partial \theta_b} \mathbf{\Sigma}_i \frac{\partial R^T}{\partial \theta_a} + R\mathbf{\Sigma}_i \frac{\partial^2 R^T}{\partial \theta_a \partial \theta_b} \right) \mathbf{\Sigma}_{s,ij}^{-1}.\end{aligned} \tag{8.20}$$

8.4 Range of scaling factors

Since distributions of the conventional NDT are constructed by discrete cells, local minima of the objective function exist. Therefore, smoothing the NDT is critical for NDT registration. In this chapter, we suggest a method of smoothing linear and planar distributions and a method of smoothing the distributions based on the translation distance.

Assume a dense point set in a cell whose size is l and range is (l_0, l_1) in one dimension, the mean μ_1 is equal to $l_0+l/2$, and the covariance $\mathbf{\Sigma}_1$ is equal to $l^2/12$. We consider another distribution which is $\mathcal{N}(\mu_2, \mathbf{\Sigma}_2)$, where $\mu_2 = l_0 + 2l$ and $\mathbf{\Sigma}_2 = \mathbf{\Sigma}_1$, and have a scaled Gaussian mixture function as

$$p(x) = \sum_{i=0}^{1} \exp\left(-\frac{(x-\mu_i)^2}{2s\mathbf{\Sigma}_i}\right), \tag{8.21}$$

and a gradient of the function expected to be zero is

$$\frac{dp(x)}{dx} = \sum_{i=0}^{1} \frac{(x-\mu_i)}{s\mathbf{\Sigma}_i} \exp\left(-\frac{(x-\mu_i)^2}{2s\mathbf{\Sigma}_i}\right). \tag{8.22}$$

By parameterizing x as $x(t) = \mu_1 + tl, t \in (0,1)$, we have the scaling factor $s(t)$ making the gradient zero as

$$s(t) = \frac{6(1-2t)}{\log(1/t-1)}. \tag{8.23}$$

By (8.23), it is possible to determine a scaling factor to set the gradient at x to 0. In this chapter, we define a target scale s_{min} which sets the gradient to zero at the middle of two means, and it is obtained by $\lim_{t\to 0.5} s(t) = 3$ which smooths distributions as shown in Fig. 8.2.

We also suggest a method to determine the target scaling factor s_{tgt} based on translation δ. As shown in Fig. 8.4, the s_{tgt} is defined as the factor scaling the distribution

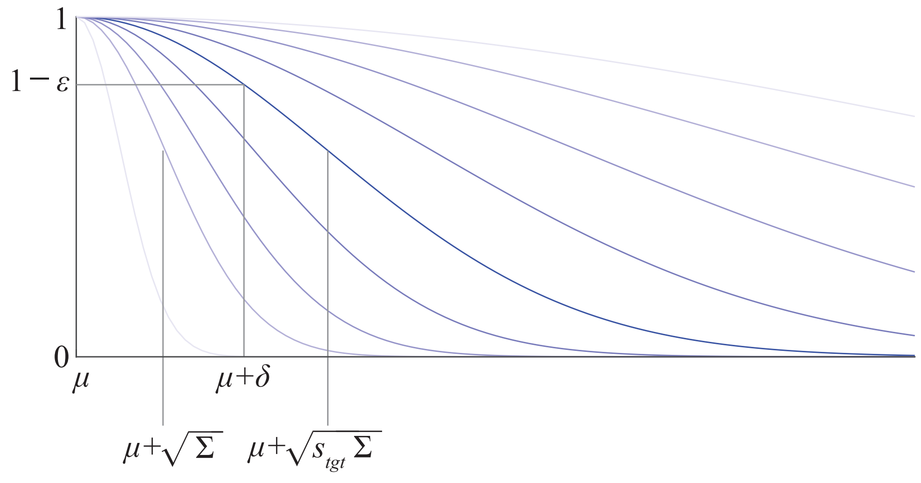


Figure 8.4: Illustration of determining s_{tgt} based on the likelihood difference limit ϵ at $\mu + \delta$.

to have likelihood equal to $1 - \epsilon$ at $\mu + \delta$ so that

$$1 - \epsilon = \exp\left(-\frac{\delta^2}{2 s_{max} \boldsymbol{\Sigma}}\right). \tag{8.24}$$

We apply the same assumption $\boldsymbol{\Sigma} = l^2/12$ as mentioned before, and s_{tgt} can be computed by

$$s_{tgt} = -\frac{6\delta^2}{l^2 \log(1 - \epsilon)}. \tag{8.25}$$

To compute s_{max} in Section 8.3, we substitute the maximum velocity v_{max} as δ. Also, the translation $|\mathbf{t}|$ can be substituted into (8.25) to obtain s_{tran} in Section 8.3.

8.5 Experiment

To evaluate the performance of the registration with the dynamic scaling factor, first, we conducted experiments on the NDT-D2D and PNDT-D2D without initial guess in the following cases: different scaling factors, different s_s with s_t increasing from 0 to

s_{ref}, and the presented method. Second, we experimented with estimating odometry by the conventional PNDT-D2D and PNDT-D2D-DSF as an application. In this case, the registration has a previous transformation as the initial guess.

8.5.1 Evaluation of the presented method without initial guess

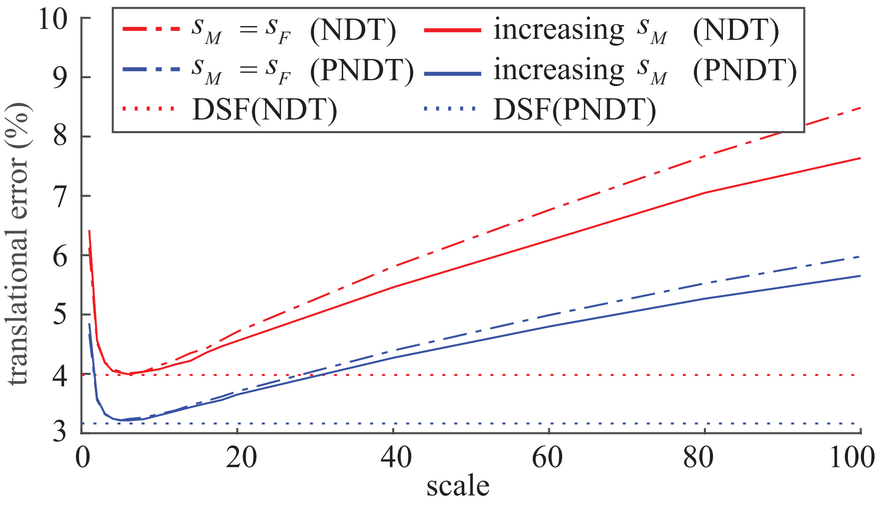
For the first case with fixed scaling factors, the NDT-D2D and PNDT-D2D register point sets with $s_t = s_s \in S = \{1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 16, 18,20, 40, 60,80, 100\}$. For the second case with dynamic s_s and fixed s_t, the NDT-D2D and PNDT-D2D perform with $s_t \in S$ and s_s increasing from 0 to s_t. For the third case of the presented DSF, we used the linear function as an example of $S(k, k_i, k_f, s_i, s_f)$:

$$s(k) = S_{de}(k, k_1, s_1, s_2) = \frac{s_f - s_i}{k_f - k_i}k, \tag{8.26}$$

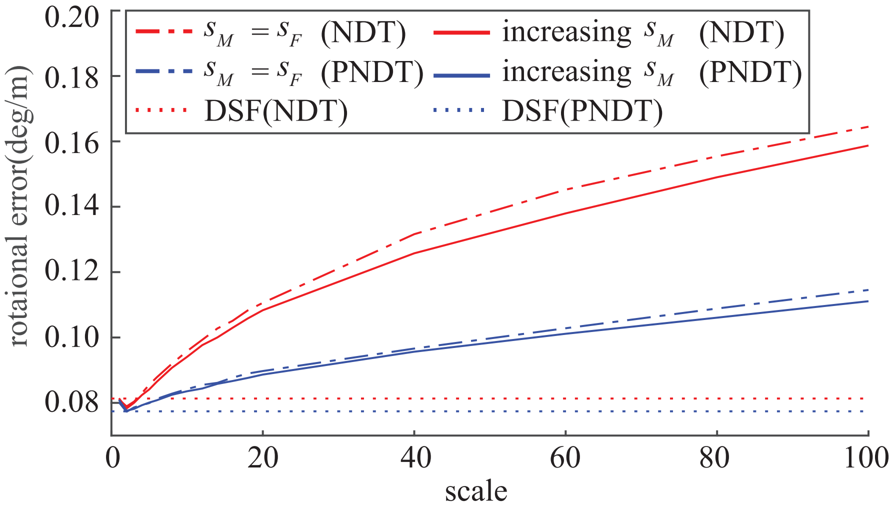
and we set k_1 to 4 and k_2 to 7. To compute s_{max}, we assumed that the maximum velocity of the car is 180km/h which is equal to 5m/Hz, and we substituted 5 as δ into (8.25). As a result, we obtained the median of the translational and rotational errors as shown in Fig. 8.5.

In Fig. 8.5, the lowest translational error of NDT-D2D can be found at $s = 6$, and the lowest rotational error can be found at $s = 2$. Also, the lowest translational and rotational errors of PNDT-D2D can be found at $s = 6$ and $s = 2$, respectively. Also, the second case which has s_s increasing from 0 to s_t shows the lower errors than the first case for both NDT-D2D and PNDT-D2D.

To compare the third case to the best results, we extract the results of $s = 2$ and $s = 6$ and demonstrate them in the form of boxplots as shown in Fig. 8.6. The translational and rotational errors of NDT-D2D-DSF and PNDT-D2D with DSF (PNDT-D2D-DSF) show the lower medians than the case $s = 6$ and $s = 2$, respectively.

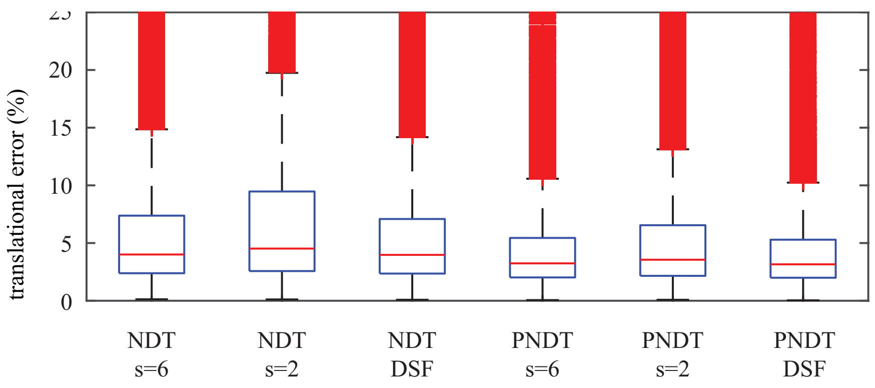


(a)

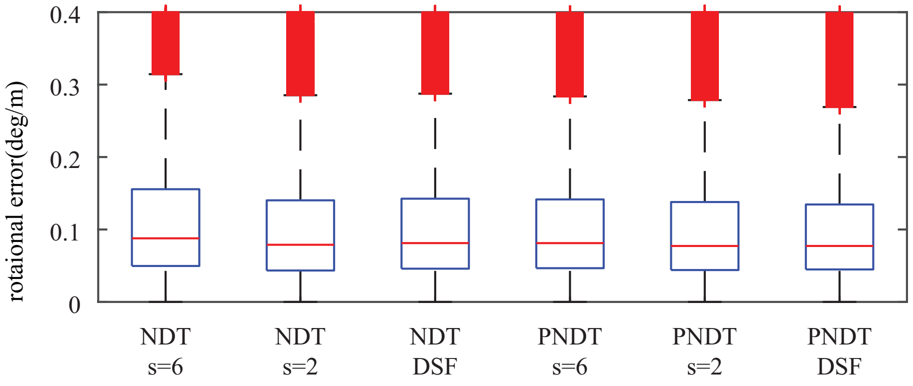


(b)

Figure 8.5: Median of the translational and rotational errors against the scale. The lowest translational error is found at $s = 6$, and the lowest rotational error is found at $s = 2$. (a) translational errors (b) rotational errors.



(a)



(b)

Figure 8.6: Errors of NDT-D2D and PNDT-D2D with $s = 6$, $s = 2$, and DSF, respectively. (a) box plots of the translational errors (b) box plots of the rotational errors.

Also, the interquartile range (IQR) of NDT-D2D-DSF is narrower than NDT-D2D with $s = 6$ in Fig. 8.6(a). On the other hand, since it is difficult to compare PNDT-D2D-DSF to PNDT-D2D with the fixed scaling factor, we have values to show the improvements as follows. For the translational error, the IQR of PNDT-D2D-DSF equal to 3.2982 is similar to the IQR of PNDT-D2D with $s = 6$ equal to 3.4233, and the median of PNDT-D2D-DSF equal to 3.1573 is less than the median of PNDT-D2D with $s = 6$ equal to 3.239 about 2.52%. Also, for the rotational error, the IQR of PNDT-D2D-

DSF equal to 0.0897 is also similar to the IQR of PNDT-D2D with $s = 6$ equal to 0.0949, and the median of PNDT-D2D-DSF equal to 0.0039 is less than the median of PNDT-D2D with $s = 2$ equal to 0.0480 about 4.8%.

8.5.2 Application of odometry estimation

The main purposes of this experiment are not only the application of the presented method but also the adaptivity to the different cell sizes. We expected that s_{tran}, the scaling factors considering translation in the second and third process, would increase as cell size decreases since δ is similar in (8.25) while l is decreasing. Hence, we conducted the experiment with cell size l equal to 0.5m, 1m, 2m, and 4m. Since PNDT-D2D-DSF shows better accuracy than NDT-D2D-DSF, we chose PNDT-D2D-DSF to compare to the conventional PNDT-D2D. To improve the accuracy, PNDT-D2D and PNDT-D2D-DSF have the previous transformation as the initial guess. As a result of 0th to 10th sequences, the errors against the cell size are as shown in Table 8.1 and 8.2. The averages of the translational and rotational errors of PNDT-D2D-DSF are almost smaller than those of PNDT-D2D. However, the errors from 7th to 9th sequences are higher than those of PNDT-D2D. The reason in the 7th sequence is that the estimated velocity becomes small on the street like a corridor. Since DSF smooths the objective function, the transformation can be converged into 0, which leads to the lower cost than the ground truth. Also, the reason in the 8th sequence is that the environment at the first is so extensive that the point set forms a shape of a disk. Hence, it leads to the cost estimating 0 is lower than the ground truth.

Table 8.1: Translation error of odometry estimated by PNDT-D2D and PNDT-D2D-DSF.

seq.	translational error(%)							
	cell size=0.5m		cell size=1m		cell size=2m		cell size=4m	
	PNDT	DSF	PNDT	DSF	PNDT	DSF	PNDT	DSF
0	2.11	1.58	2.31	1.79	1.72	1.63	2.00	1.93
1	3.06	2.21	3.22	2.39	3.46	2.17	6.91	2.38
2	10.13	2.12	3.14	2.02	4.13	2.56	7.48	5.89
3	2.78	2.10	3.64	2.85	3.22	1.79	4.39	4.50
4	80.50	91.22	3.42	2.29	4.96	2.82	14.83	3.69
5	1.59	1.53	1.72	1.63	1.66	1.64	1.83	1.82
6	12.67	1.39	2.06	1.50	1.99	1.77	1.70	1.68
7	0.80	0.92	0.94	1.07	1.22	1.03	1.62	1.70
8	1.92	2.07	2.12	2.16	2.04	4.64	2.46	2.49
9	1.43	1.44	1.73	1.76	2.59	2.20	6.10	4.19
10	2.07	2.01	2.52	2.38	2.55	2.00	3.22	3.23
mean	10.82	9.87	2.44	1.99	2.69	2.20	4.78	3.05

Table 8.2: Rotation error of odometry estimated by PNDT-D2D and PNDT-D2D-DSF.

seq.	rotational error (10^{-3}deg/m)							
	cell size=0.5m		cell size=1m		cell size=2m		cell size=4m	
	PNDT	DSF	PNDT	DSF	PNDT	DSF	PNDT	DSF
0	11.3	10.5	11.5	11.4	12.2	12.3	15.6	15.4
1	24.5	14.6	16.4	12.0	14.9	13.2	14.2	13.2
2	66.8	11.2	13.4	12.4	14.1	13.8	26.6	24.3
3	9.1	8.3	9.2	7.8	11.4	11.3	12.7	11.9
4	6.5	3.2	8.8	8.6	7.8	7.7	8.3	10.0
5	12.5	10.7	10.9	10.7	11.9	12.0	13.7	14.0
6	13.6	9.4	12.9	13.2	11.5	11.3	13.2	13.2
7	6.7	4.2	5.6	5.9	6.7	5.9	12.8	13.4
8	11.9	11.3	11.0	11.5	12.7	18.1	15.6	16.9
9	15.1	13.5	15.2	15.5	17.8	18.2	34.7	30.3
10	12.5	12.7	12.5	10.5	17.1	15.4	20.8	20.3
mean	17.3	10.0	11.6	10.9	12.6	12.7	17.1	16.6

8.6 Summary

We presented a dynamic scaling factor approach to improve the accuracy of NDT registration. To avoid the negative correlation between L_2 distance and rotational alignment, the presented method DSF initially sets s_s to 0. Also, to smooth the objective function and to avoid the discreteness of GCs, DSF sets the range for s_t.

We conducted two experiments. First, we compared the registration with the presented DSF to the registrations with different fixed scaling factors and the registrations with s_s increasing from 0. As a result, the presented DSF can improve the accuracy of NDT-D2D and PNDT-D2D. Second, we experimented with estimating odometry as an application of PNDT-D2D with the presented method. As a result, the accuracy of PNDT-D2D-DSF is higher than the conventional PNDT-D2D.

Chapter 9

Scan-to-map Registration

9.1 Introduction

Incremental registration is a scan-to-map registration [69]. Since the map has the better recall for the source point set than the scan, the incremental registration can lead to the better accuracy than the scan-to-scan registration. Therefore, the method of simultaneous mapping and tracking proposed in [70] shows considerable performance. Also, the mapping thread of LOAM is based on an incremental registration [20].

In this chapter, we present an incremental registration based on multi-layered probabilistic normal distributions transform (ML-PNDT). Incremental registration is to obtain the accurate target ML-PNDT, and the multi-layer approach is to deal with the considerable pose variation of the robot. To accurately estimate the robot pose, the presented method shares the lattice of the global map to generate the target and source ML-PNDTs. We evaluated the accuracy performance of the presented method with KITTI benchmark dataset and compared with the state-of-the-art method LOAM. Also, we evaluated the elapsed time of each process of the presented method.

9.2 Multi-layered PNDT

To overcome the problem of determining the cell size for NDT, a multi-layer, also called multi-resolution, approach which generates NDT for lattices with several lattices was proposed [58, 60, 96]. The approach is usually implemented based on the octree structure [79, 96], and the multi-layered NDT takes the strategy that generates NDTs from the fine layer L_f to the coarse layer L_c [96, 60]. Also, an efficient method of generating NDT $\mathcal{D}^{L-1}$ by reusing $\mathcal{D}^L$ has been proposed in [60, 96].

In this chapter, we define the multi-layered PNDT (ML-PNDT) $\mathcal{M}$ as follows:

$$\mathcal{M} = \{\mathcal{D}^L\}, L = L_c, L_c + 1, ..., L_f - 1, L_f, \tag{9.1}$$

and we show that the multi-layer approach is also applicable to the PNDT. In the following sections, we use the NDTs as follows:

- $\mathcal{D}^P$: point set with uncertainties as an NDT (Fig. 9.1(a)).
- $\mathcal{D}^{L_f}$: NDT converted from $\mathcal{D}^P$ by the lattice $\mathcal{V}^{L_f}$.
- $\mathcal{D}^L$: NDT converted from the NDT $\mathcal{D}^L$ in the layer $L+1$ (Fig. 9.1(c)).

It is an efficient strategy to generate the NDT $\mathcal{D}^L$ in the layer L by reusing the NDT $\mathcal{D}^{L+1}$. For example, given a cell v_i^L in the layer L to generate a GC $d_i^L = (\mathcal{N}(\mu_i^L, \mathbf{\Sigma}_i^L), n_i^L)$ as shown in Fig. 9.1(c), v_i^L has a GC set $\{d_j\}_i, j = 1, 2, ..., n_{D,i}^L$, which is an NDT $\mathcal{D}_i^L$ in Fig. 9.1(b). In short, it is a process that converts an NDT $\mathcal{D}_i^L$ into a GC d_i^L in the cell v_i^L, and the equations can be derived from (9.2)-(9.4) as follows:

$$n_i^L = n_{P,i} = \sum_{j=1}^{n_{D,i}^L} n_j^{L+1}, \tag{9.2}$$

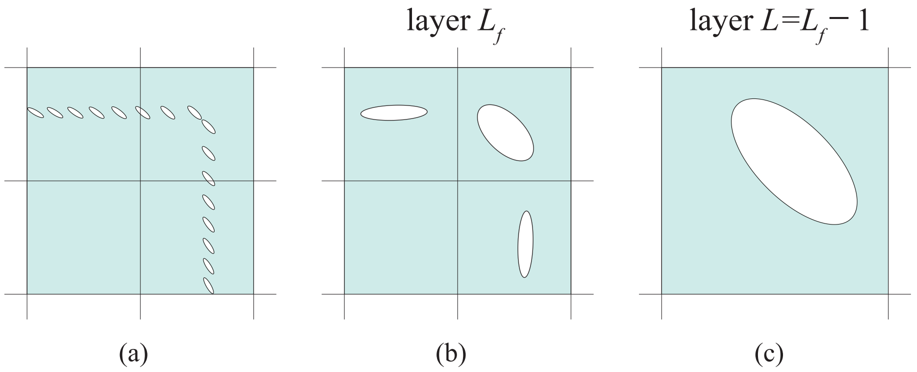


Figure 9.1: Illustration of a point set with uncertainties and the NDTs: point set in (a), NDT in layer L in (b), and NDT in layer $L-1$ in (c).

$$\mu_i^L = \frac{1}{n_i^L}\sum_{j=1}^{n_{D,i}^L}\sum_{k=1}^{n_j^{L+1}} \hat{\mathbf{p}}_{jk} = \frac{1}{n_i^L}\sum_{j=1}^{n_{D,i}^L} n_j^{L+1}\mu_j^{L+1}, \tag{9.3}$$

$$\begin{aligned}
\mathbf{\Sigma}_i^L &= \frac{1}{n_i}\sum_{j=1}^{n_{D,i}^L}\sum_{k=1}^{n_j^{L+1}} \hat{\mathbf{p}}_{jk}\hat{\mathbf{p}}_{jk}^T - \mu_i^L{\mu_i^L}^T + \frac{1}{n_i}\sum_{j=1}^{n_{D,i}^L} n_j^{L+1}\left(\frac{1}{n_j^{L+1}}\sum_{k=1}^{n_j^{L+1}} \mathbf{\Sigma}_{U,jk}\right) \\
&= \frac{1}{n_i}\sum_{j=1}^{n_{D,i}^L} n_j^{L+1}\left(\mathbf{\Sigma}_j^{L+1} + \mu_j^{L+1}{\mu_j^{L+1}}^T\right) - \mu_i^L{\mu_i^L}^T \\
&= \frac{1}{n_i}\sum_{j=1}^{n_{D,i}^L} n_j\mu_j\mu_j^T - \mu_i\mu_i^T + \frac{1}{n_i}\sum_{j=1}^{n_{D,i}^L} n_j\mathbf{\Sigma}_j.
\end{aligned} \tag{9.4}$$

Since $n_{D,i}^L \leq 8 \ll n_P$ ($n_{D,i}^L \leq 4$ for 2D case), it is efficient to generate the NDT in the layer $L > L_f$ by reusing the NDT in the layer $L+1$. As a result, the ML-PNDT in (9.1) can be constructed efficiently by the functions generating NDTs from L_f to L_c recursively.

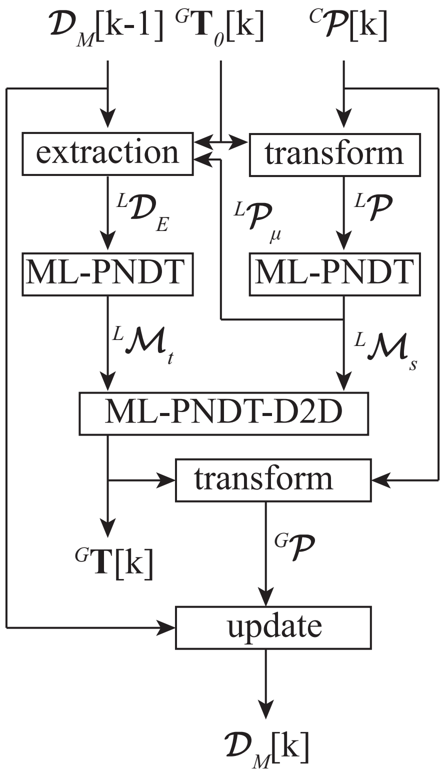


Figure 9.2: Block diagram of the presented method.

9.3 NDT Incremental Registration

We propose a method to estimate odometry and build a map based on PNDT representation and the incremental registration [69], as shown in Fig. 9.2. The key idea of the presented method is that generating target and source ML-PNDTs similar to each other. To this end, the source ML-PNDT $\mathcal{M}_s$ is converted from the transformed point set $\mathcal{P}$ by sharing the lattice of the PNDT-map $\mathcal{D}_M$, and the target ML-PNDT $\mathcal{M}_t$ is regenerated from the local PNDT in the PNDT-map $\mathcal{D}_M$.

In the rest of the chapter, we consider three coordinate systems shown in Fig. 9.3. First, the conventional coordinate system of the robot is denoted by $\{C\}$. Second, the global coordinate system $\{G\}$ is determined by the initial robot pose, and it is also the map coordinate system $\{M\}$. Third, the local coordinate system $\{L\}$ is translated from $\{G\}$, and its origin is the location of the robot. For convenience, we denote the rigid-body transformation by ${}^{B}_{A}\mathbf{T}_{H}$, which transforms coordinates from the coordinate system $\{A\}$ to $\{B\}$. The components of the presented method can be listed as follows:

- initializing the PNDT-map $\mathcal{D}_M$.
- generating source ML-PNDT $\mathcal{M}_s$.
- reconstructing target ML-PNDT $\mathcal{M}_t$.
- estimating the pose by registering $\mathcal{M}_s$ to $\mathcal{M}_t$.
- updating PNDT-map $\mathcal{D}_M$.

The details of each step are described in the following subsections.

9.3.1 Initialization of PNDT-Map

We denote the PNDT-map by $\mathcal{D}_M$ since the map only consists of the PNDT in the layer L_f. The reason is that PNDT requires less memory than ML-PNDT, and to be similar to the source PNDT, the presented method reconstructs the ML-PNDT from the PNDT-map instead of reusing PNDTs in the multiple layers.

After the robot obtains the first point set $\mathcal{P}[0]$, $\{G\}$ and $\{M\}$ are determined as $\{C\}_0$, which is the conventional coordinate system at discrete time 0. Also, $\mathcal{P}[0]$ is converted into a PNDT, which is the initial PNDT-map.

9.3.2 Generation of Source ML-PNDT

If the source NDT is converted from the point set before it is transformed, it results in the difference between the target and source NDTs as shown in Fig. 9.4. To generate $\mathcal{M}_s$ similar to $\mathcal{M}_t$ regenerated from the PNDT-map, we propose the method to share the lattice of the PNDT-map with the source point set $\mathcal{P}$. To this end, the presented method generates ML-PNDT after transforming $\mathcal{P}$.

At first, the point set ${}^{C}\mathcal{P}$ in $\{C\}$ is obtained by the sensor. Given an initial guess ${}^{G}_{C}\mathbf{T}_H$ of the robot pose, a transformation ${}^{L}_{C}\mathbf{T}_H$ and ${}^{L}_{G}\mathbf{T}_H$ can be obtained as

$$
{}^{L}_{C}\mathbf{T}_H = \begin{bmatrix} {}^{L}_{C}\mathbf{R} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} , {}^{L}_{G}\mathbf{T}_H = \begin{bmatrix} I_{3\times3} & \mathbf{x} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}, \tag{9.5}
$$

where ${}^{L}_{C}\mathbf{R}$ is the rotation matrix equal to ${}^{G}_{C}\mathbf{R}$, and $\mathbf{x}$ is the initial guess of the robot location. Next, ${}^{L}\mathcal{P}$ is subdivided by the lattice shared by the map. To share the lattice, the presented method transforms all of the centers of the cells in each layer from $\{G\}$ to $\{L\}$, and it regards the vertex which is the nearest to the robot location as the new center ${}^{L}\mathbf{c}_{new}$ for the root cell. At last, it subdivides ${}^{L}\mathbf{P}$ to generate the source ML-PNDT $\mathcal{M}_s$.

9.3.3 Reconstruction of The Target ML-PNDT

The reconstruction of the target ML-PNDT $\mathcal{M}_t$ can be divided into three processes. In the first process, the mean vector set ${}^{L}\mathcal{P}_\mu = \{{}^{L}\mu_i\}_{i=1}^{n_D^{L_c}}$ in the layer L_c is extracted from ${}^{L}\mathcal{M}_s$. Next, ${}^{L}\mathcal{P}_\mu$ obtains the GCs in the gray cells of PNDT-map, as shown in Fig. 9.3. At the last, the $\mathcal{M}_t$ sets the new center $\mathbf{c}_{new}$ as the center of the multi-layered lattices to regenerate PNDT from layer L_f to L_c recursively based on the method in Section 9.3.2.

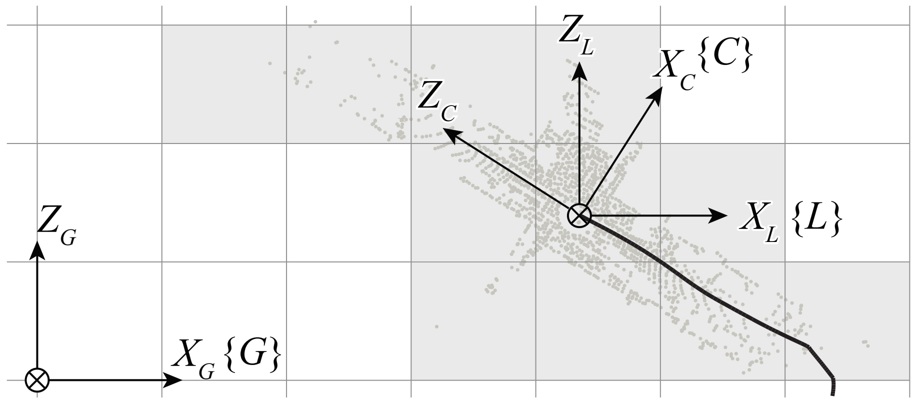


Figure 9.3: Illustration of coordinate systems and the cells providing GCs in the coarse layer (gray cells).

9.3.4 Pose Estimation Based on Multi-layered Registration

In the presented method, the pose is estimated by registering $\mathcal{D}_s^L$ to $\mathcal{D}_t^L$ based on minimizing (2.13) from the coarse layer L_c to the fine layer L_f. The presented method estimates the robot pose based on the multi-layered PNDT-D2D (ML-PNDT-D2D) since it can deal with the long-range pose variation.

As shown in Fig. 9.2, the initial guess ${}^G\mathbf{T}_{H,0}[k]$ at the discrete time k is applied to the source and target ML-PNDTs in the 'transform' and 'extraction' blocks respectively. Therefore, the presented multi-layered PNDT-D2D (ML-PNDT-D2D) estimates the pose variation ${}^L\mathbf{V}$ between the pose registering PNDTs and the initial guess pose in $\{L\}$. As a result, the estimated pose variation ${}^L\mathbf{V}$ can be obtained by ML-PNDT-D2D, and ${}^G\mathbf{T}_H[k]$ can be derived as

$$ {}^G\mathbf{T}_H[k] = {}^G_L\mathbf{T}_H \quad {}^L\mathbf{V} \quad {}^L_G\mathbf{T}_H \quad {}^G\mathbf{T}_{H,0}[k]. \tag{9.6} $$

The initial guess can be obtained by additional devices such as an inertial measurement unit and wheel odometry. For the robot without those devices, the estimated

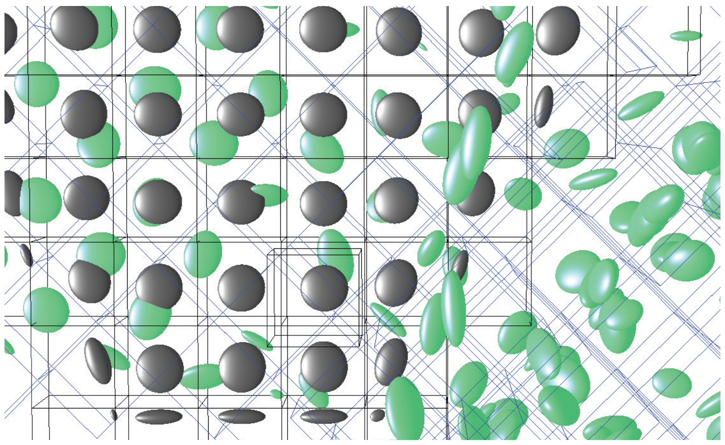


Figure 9.4: 1σ ellipsoids of PNDTs in the inconsistent lattices.

velocity can be used as an initial guess, and it is computed as

$$ {}^{G}\mathbf{T}_{H,0}[k] = {}^{G}\mathbf{T}_{H}[k-1]\left({}^{G}\mathbf{T}_{H}[k-2]^{-1}\,{}^{G}\mathbf{T}_{H}[k-1]\right). \tag{9.7} $$

9.3.5 Update of PNDT-Map

The update process integrates the source point set transformed by the robot pose ${}^{G}\mathbf{T}_{H}[k]$ with the existing PNDT-map $\mathcal{D}_s[k-1]$. We integrate the point set instead of the aligned source PNDT since the PNDT is not guaranteed to fit the lattice of $\mathcal{D}_s$. For example, the PNDT aligned as shown in Fig. 9.4 leads to one or more GCs being in the same cell v thus the update can result in the distortion of GCs, such as drifted center, enlarged eigenvalues, and changed eigenvectors. To avoid those distortions and update the map accurately, the robot should update the GC $d[k] = (\mathcal{N}(\mu[k], \mathbf{\Sigma}[k]), n[k]))$ with the point set $\mathcal{P}$ in the same cell as follows:

$$ n[k] = n[k-1] + n_P, \tag{9.8} $$

$$\mu[k] = \frac{1}{nk]}\left(n[k-1]\mu[k-1] + \sum_{i=1}^{n_P} \mathbf{p}_i\right), \tag{9.9}$$

$$\mathbf{\Sigma}[k] = \frac{1}{n[k]}\left(n[k-1]\left(\mathbf{\Sigma}[k-1] + \mu[k-1]\mu[k-1]^T\right) + \sum_{i=1}^{n_P} \mathbf{p}_i\mathbf{p}_i^T\right) - \mu[k]\mu[k]^T. \tag{9.10}$$

9.4 Experiments

We evaluated the accuracy of the pose estimated by the presented method with the KITTI data set sequence 0-10.

The side length of a cell is set to 1 m, and the maximum range is set to 100 m [83]. The cells are constructed in the octree structure, and the fine layer L_f was computed as 8. Also, we set the coarse layer L_c to 5. For PNDT, we applied the 1σ value for the range at 15mm and encoder angle at 0.026 degree quantization noise from the manufacturer specification. The regularizing factors r_1 and r_2 in (2.13) are set to 1 and $1/3$ respectively. To accelerate the presented method, the number of source GCs in each layer was limited by 3000, and GCs are randomly chosen. The maximum number of iterations is set to 40.

In the visualization as shown in Fig. 9.5, we confirmed that the blue 1σ ellipsoids in the cells of layer L_c are called by the $\mathcal{P}_\mu$. Also, the lattice of the global map is shared to generate $\mathcal{D}_t^{L_f}$ in blue and $\mathcal{D}_s^{L_f}$ in red as shown in the zoomed region in Fig. 9.5.

As a result, the presented method estimated the odometry as shown in Fig. 9.6. The mean relative translational errors of odometry estimated by the presented method are compared with the state-of-the-art LOAM [20] in Table 9.1. Also, the presented method is ranked at the 16th on KITTI visual odometry chart. Except for sequence 0,3,and 10, the presented method shows 1-31% better accuracy than LOAM. Also, the

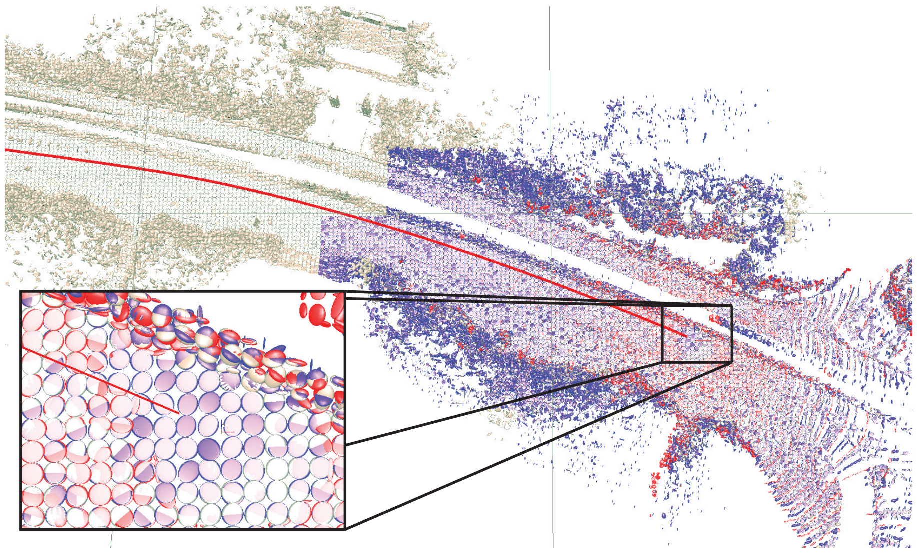


Figure 9.5: Reconstructed target NDT in the layer L_f. The target NDT is in blue, source NDT is in red, and the map NDT is in gray. The blue ellipsoids are in the cells where the mean vectors of the source GCs in the layer L_c are.

presented method shows the better mean of the errors of all sequences than LOAM does. The error of sequence 0 is 8.4% higher than LOAM, and the error of sequence 10 is 5.5% higher than LOAM. However, the error of sequence 3 is 92.6% much higher than LOAM.

We also recorded the elapsed time of each process and the total time, and the means and standard deviations are summarized in Table 9.2. Since the maximum range of HDL-64E is so wide that the number of points acquired from HDL-64E is large, the point set registration algorithm hardly process in real time. For example, in [20], LOAM takes one second to process a scan of KITTI dataset. Thus, the presented method has the mean of elapsed time equal to 0.717 seconds and the deviation is 0.217 seconds. The most time-consuming process is registration. It is possible to reduce the

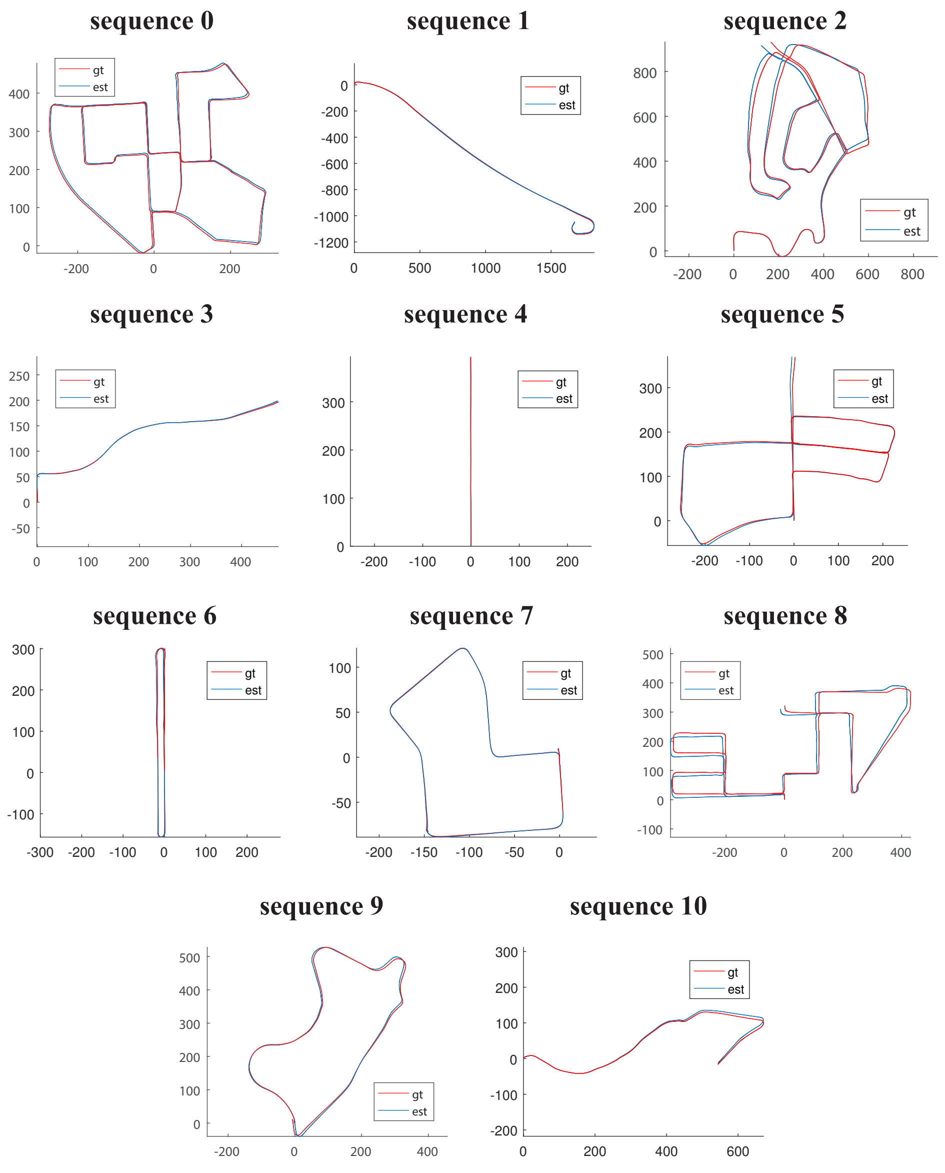


Figure 9.6: Odometry of sequence 0-10 estimated by the presented method.

Table 9.1: Results of KITTI benchmark datasets

seq. no.	configuration		mean relative translational error(%)		mean relative rotational error(deg/m)
	distance(m)	environment	[20]	presented	presented
0	3714	urban	**0.78**	0.85	0.0040
1	4268	highway	1.43	**0.99**	0.0012
2	5075	urban+country	0.92	**0.78**	0.0028
3	563	country	**0.86**	1.65	0.0036
4	397	country	0.71	**0.70**	0.0006
5	2223	urban	0.57	**0.56**	0.0030
6	1239	urban	0.65	**0.52**	0.0027
7	695	urban	0.63	**0.50**	0.0039
8	3225	urban+country	1.12	**0.92**	0.0033
9	1717	urban+country	0.77	**0.63**	0.0026
10	919	urban+country	**0.79**	0.83	0.0031
		mean	0.84	**0.81**	0.0028

maximum iteration number to accelerate the registration;however, it sacrifices accuracy since the optimization may not converge.

9.5 Summary

In this chapter, we showed that the recursive generation of multi-layered NDT can be applied to PNDT. Also, we present the method of estimating the robot pose based on

Table 9.2: Elapsed time of each process

process	mean(s)	deviation(s)
loading point set	0.081	0.116
generation of $\mathcal{M}_s$	0.104	0.04
reconstruction of $\mathcal{M}_t$	0.066	0.036
registration	0.335	0.275
map update	0.086	0.031
total	0.717	0.217

the multi-layered registration between the source ML-PNDT and target ML-PNDT reconstructed from the PNDT-map. The key idea of the presented method is improving pose accuracy by generating source and target PNDTs similar to each other. Thus, the presented method sets the vertex nearest to the robot location as the new center and uses the same global lattice to generate the source and target ML-PNDTs. In experiments, we showed that the accuracy performance of the presented method is higher than the state-of-the-art method LOAM by using KITTI benchmark data set. Also, we showed the elapsed time of each process of the presented method. The results showed that the registration process is the most time consuming one. Since enlarging cells and limiting the number of iterations sacrifice the accuracy, we came to the conclusion that the selection of the key GCs is critical to improve the runtime of PNDT registration.

Chapter 10

Conclusions

This dissertation presents methods of improving a point set registration using the NDT representation. The first approach is improving the NDT representation. First, we presented a PNDT which can avoid destruction caused by high resolution of cells using the sensor uncertainty. The mean is not changed, but the covariance is changed as the sum of conventional covariance and the average of covariances of point samples. Since each point sample has its covariance, the presented PNDT can generate distributions in all of the occupied cells. Thus, the number threshold of points samples is no more necessary. To show the improvement of the presented method, two experiments are conducted. The results of representation show that the generation of distributions in all of the occupied cells regardless of the resolution is achieved by applying PNDT. Second, we defined the overlapped cells and modified the octree to adjust the lattice. We also present a cell insertion method to generate the body-centered cubic structure and face-centered cubic structure lattice. In the experiments, we compared the odometry accuracy against the cell side length and against the distance between cell centroids, respectively. For the cell insertion method, the BCC, FCC, and SC structured lattices are compared in the experiment. We also compared the cases of applying the correspondence region method. Third, we presented a method of regenerating a source NDT fit toe the target lattice. The method subdivides an NDT into truncated GCs and fuses the truncated GCs in the same cell. In the experiment, the presented method showed the more accurate representation than the naive regeneration. As shown in experimental results against the weight threshold w_{th}, the presented method outperforms the simple fusion of NDTs.

The second approach is improving the NDT registration. First, we presented a HA-NDT weighting the likelihoods of the target NDT and the source point set by the similarity of hue distributions. As a result, it improved the accuracy of the transformation than the conventional NDT registration. Second, we presented an efficient multi-layered NDT registration using the key layer. The number of layers and the number of iterations per layer are not fixed in the presented KL-NDT. The main feature of KL-NDT called 'searching for key layers' is introduced, and the performance is demonstrated by the experiment. Also, the terminating criteria of the algorithm and the registration in each layer are presented. In the experiment, KL-NDT-P2D shows the higher success rates and the lower errors than ML-NDT-P2D. For the NDT-D2D case, the accuracy performances of ML-NDT-D2D and KL-NDT-D2D are similar; however, KL-NDT-D2D processes faster than ML-NDT-D2D. Third, we presented a dynamic scaling factor approach to improve the accuracy of NDT registration. To avoid the negative correlation between L_2 distance and rotational alignment, the presented method DSF initially sets s_s to 0. Also, to smooth the objective function and to avoid the discreteness of GCs, DSF sets the range for s_t. We conducted two experiments. First, we compared the registration with the presented DSF to the registrations with different fixed scaling factors and the registrations with s_M increasing from 0. As a result, the presented DSF is able to improve the accuracy of NDT-D2D and PNDT-D2D. Second, we conducted an experiment estimating odometry as an application of PNDT-D2D with the presented method. As a result, the accuracy of PNDT-D2D-DSF is higher than the conventional PNDT-D2D. Fourth, we presented a scan-to-map incremental NDT registration. As a result, the presented method outperforms the state-of-the-art odometry estimation method, LOAM. The reasons are as follows. The PNDT provides a dense representation even if the resolution is very high. Also, the PNDT is generated after the point set is transformed by the initial guess. Thus, the PNDT can be more similar to the submap than the PNDT which is converted from a point set and then transformed. In addition, the presented method can rapidly extract the submap which is consisted of cells where the source GCs are. Moreover, due to the multi-layered registration of the source NDT and the target submap, the coarse registration provides a reasonable initial guess to the fine registration.

Bibliography

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[2] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[3] S.-H. Lee, "Cooperative rao-blackwellized particle filter based slam framework using geometric information and inter-robot measurements," Ph.D. dissertation, Seoul National University, 2015.

[4] N. Ayache and O. D. Faugeras, "Maintaining representations of the environment of a mobile robot," *IEEE transactions on Robotics and Automation*, vol. 5, no. 6, pp. 804–819, 1989.

[5] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, *et al.*, "Fastslam: A factored solution to the simultaneous localization and mapping problem," *American Association for Artificial Intelligence*, pp. 593–598, 2002.

[6] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based slam," *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4,

pp. 31–43, 2010.

[7] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, “isam2: Incremental smoothing and mapping using the bayes tree,” *The International Journal of Robotics Research (IJRR)*, vol. 31, no. 2, pp. 216–235, 2012.

[8] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, “g 2 o: A general framework for graph optimization,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 3607–3613.

[9] K. Ohno, T. Tsubouchi, B. Shigematsu, and S. Yuta, “Differential gps and odometry-based outdoor navigation of a mobile robot,” *Advanced Robotics*, vol. 18, no. 6, pp. 611–635, 2004.

[10] J. Biswas and M. Veloso, “Wifi localization and navigation for autonomous indoor mobile robots,” in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 4379–4384.

[11] D. Hahnel, W. Burgard, D. Fox, K. Fishkin, and M. Philipose, “Mapping and localization with rfid technology,” in *Robotics and Automation (ICRA), 2004 IEEE International Conference on*, vol. 1. IEEE, 2004, pp. 1015–1020.

[12] S. Park and S. Hashimoto, “Autonomous mobile robot navigation using passive rfid in indoor environment,” *IEEE Transactions on Industrial Electronics*, vol. 56, no. 7, pp. 2366–2373, 2009.

[13] A. N. Raghavan, H. Ananthapadmanaban, M. S. Sivamurugan, and B. Ravindran, “Accurate mobile robot localization in indoor environments using bluetooth,” in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 4391–4396.

[14] J. Yi, J. Zhang, D. Song, and S. Jayasuriya, “Imu-based localization and slip estimation for skid-steered mobile robots,” in *Intelligent Robots and Systems (IROS), 2007 IEEE/RSJ International Conference on*. IEEE, 2007, pp. 2845–2850.

[15] J. Shen, D. Tick, and N. Gans, “Localization through fusion of discrete and continuous epipolar geometry with wheel and imu odometry,” in *American Control Conference (ACC), 2011*. IEEE, 2011, pp. 1292–1298.

[16] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, “Monoslam: Real-time single camera slam,” *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 6, pp. 1052–1067, 2007.

[17] K. Konolige, M. Agrawal, R. C. Bolles, C. Cowan, M. Fischler, and B. Gerkey, “Outdoor mapping and navigation using stereo vision,” in *Experimental Robotics*. Springer, 2008, pp. 179–190.

[18] J.-P. Tardif, Y. Pavlidis, and K. Daniilidis, “Monocular visual odometry in urban environments using an omnidirectional camera,” in *Intelligent Robots and Systems (IROS), 2008 IEEE/RSJ International Conference on*. Nice, France: IEEE, 2008, pp. 2531–2538.

[19] C. Glennie and D. D. Lichti, “Static calibration and analysis of the velodyne hdl-64e s2 for high accuracy mobile scanning,” *Remote Sensing*, vol. 2, no. 6, pp. 1610–1624, 2010.

[20] J. Zhang and S. Singh, “Low-drift and real-time lidar odometry and mapping,” *Autonomous Robots*, vol. 41, no. 2, pp. 401–416, 2017.

[21] P. Checchin, F. Gérossier, C. Blanc, R. Chapuis, and L. Trassoudaine, "Radar scan matching slam using the fourier-mellin transform," in *Field and Service Robotics*. Springer, 2010, pp. 151–161.

[22] J. D. Tardós, J. Neira, P. M. Newman, and J. J. Leonard, "Robust mapping and localization in indoor environments using sonar data," *The International Journal of Robotics Research (IJRR)*, vol. 21, no. 4, pp. 311–330, 2002.

[23] S. May, D. Droeschel, D. Holz, C. Wiesen, S. Fuchs, *et al.*, "3d pose estimation and mapping with time-of-flight cameras," in *Intelligent Robots and Systems (IROS) 3D Mapping workshop, 2008 IEEE/RSJ International Conference on*. Nice, France: IEEE, 2008.

[24] H. Sarbolandi, D. Lefloch, and A. Kolb, "Kinect range sensing: Structured-light versus time-of-flight kinect," *Computer vision and image understanding*, vol. 139, pp. 1–20, 2015.

[25] T. Whelan, M. Kaess, H. Johannsson, M. Fallon, J. J. Leonard, and J. McDonald, "Real-time large-scale dense rgb-d slam with volumetric fusion," *The International Journal of Robotics Research (IJRR)*, vol. 34, no. 4-5, pp. 598–626, 2015.

[26] F. Fraundorfer and D. Scaramuzza, "Visual odometry: Part ii: Matching, robustness, optimization, and applications," *IEEE Robotics & Automation Magazine*, vol. 19, no. 2, pp. 78–90, 2012.

[27] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1. IEEE, 2004, pp. 652–659.

[28] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on.* IEEE, 2007, pp. 225–234.

[29] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[30] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "Dtam: Dense tracking and mapping in real-time," in *Computer Vision (ICCV), 2011 IEEE International Conference on.* IEEE, 2011, pp. 2320–2327.

[31] J. Engel, J. Sturm, and D. Cremers, "Semi-dense visual odometry for a monocular camera," in *Computer Vision (ICCV), 2013 IEEE International Conference on.* IEEE, 2013, pp. 1449–1456.

[32] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European Conference on Computer Vision.* Springer, 2014, pp. 834–849.

[33] T. Keisuke, T. Federico, and L. Iro, "Cnn-slam: Real-time dense monocular slam with learned depth prediction," in *Computer Vision and Pattern Recognition, 2017. CVPR 2017. Proceedings of the 2017 IEEE Computer Society Conference on.* IEEE, 2017, pp. 6565–6574.

[34] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2018.

[35] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *Robotics and*

Automation (ICRA), 2017 IEEE International Conference on. IEEE, 2017, pp. 2043–2050.

[36] R. Li, S. Wang, Z. Long, and D. Gu, "Undeepvo: Monocular visual odometry through unsupervised deep learning," in *Robotics and Automation (ICRA), 2018 IEEE International Conference on.* IEEE, 2018, pp. 7286–7291.

[37] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor Fusion IV: Control Paradigms and Data Structures*, vol. 1611. International Society for Optics and Photonics, 1992, pp. 586–607.

[38] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *Robotics and Automation (ICRA), 2009 IEEE International Conference on.* IEEE, 2009, pp. 3212–3217.

[39] A. Censi, "An icp variant using a point-to-line metric," in *Robotics and Automation (ICRA), 2008 IEEE International Conference on.* IEEE, 2008, pp. 19–25.

[40] K. Pathak, A. Birk, N. Vaskevicius, and J. Poppinga, "Fast registration based on noisy planes with unknown correspondences for 3-d mapping," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 424–441, 2010.

[41] P. Biber and W. Straßer, "The normal distributions transform: A new approach to laser scan matching," in *Intelligent Robots and Systems (IROS), 2003 IEEE/RSJ International Conference on*, vol. 3. IEEE, 2003, pp. 2743–2748.

[42] E. Takeuchi and T. Tsubouchi, "A 3-d scan matching using improved 3-d normal distributions transform for mobile robotic mapping," in *Intelligent Robots and Systems (IROS), 2006 IEEE/RSJ International Conference on.* IEEE, 2006, pp. 3068–3073.

[43] A. Segal, D. Haehnel, and S. Thrun, “Generalized-icp,” in *Robotics: science and systems*, vol. 2, no. 4, 2009.

[44] B. Jian and B. C. Vemuri, “Robust point set registration using gaussian mixture models,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1633–1645, 2011.

[45] T. Stoyanov, M. Magnusson, H. Andreasson, and A. J. Lilienthal, “Fast and accurate scan registration through minimization of the distance between compact 3d ndt representations,” *The International Journal of Robotics Research (IJRR)*, vol. 31, no. 12, pp. 1377–1393, 2012.

[46] J.-H. Oh, “Sequence-based place recognition using deep learning features in changing environments,” Ph.D. dissertation, Seoul National University, 2018.

[47] R. M. Haralick, H. Joo, C.-N. Lee, X. Zhuang, V. G. Vaidya, and M. B. Kim, “Pose estimation from corresponding point data,” in *Machine Vision for Inspection and Measurement*. Elsevier, 1989, pp. 1–84.

[48] A. Diosi and L. Kleeman, “Laser scan matching in polar coordinates with application to slam,” in *Intelligent Robots and Systems (IROS), 2005 IEEE/RSJ International Conference on*. IEEE, 2005, pp. 3317–3322.

[49] A. Censi, L. Iocchi, and G. Grisetti, “Scan matching in the hough domain,” in *Robotics and Automation (ICRA), 2005 IEEE International Conference on*. IEEE, 2005, pp. 2739–2744.

[50] D. A. Simon, “Fast and accurate shape-based registration,” Ph.D. dissertation, Carnegie Mellon University, 1996.

[51] D. Chetverikov, D. Svirko, D. Stepanov, and P. Krsek, "The trimmed iterative closest point algorithm," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 3. IEEE, 2002, pp. 545–548.

[52] F. Lu and E. Milios, "Robot pose estimation in unknown environments by matching 2d range scans," *Journal of Intelligent and Robotic systems*, vol. 18, no. 3, pp. 249–275, 1997.

[53] J. Minguez, L. Montesano, and F. Lamiraux, "Metric-based iterative closest point scan matching for sensor displacement estimation," *IEEE Transactions on Robotics*, vol. 22, no. 5, pp. 1047–1054, 2006.

[54] L. Armesto, J. Minguez, and L. Montesano, "A generalization of the metric-based iterative closest point technique for 3d scan matching," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 1367–1372.

[55] J. Serafin and G. Grisetti, "Nicp: Dense normal based point cloud registration," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 742–749.

[56] J. Yang, H. Li, and Y. Jia, "Go-icp: Solving 3d registration efficiently and globally optimally," in *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1457–1464.

[57] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing icp variants on real-world data sets," *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, 2013.

[58] M. Magnusson, A. Lilienthal, and T. Duckett, “Scan registration for autonomous mining vehicles using 3d-ndt,” *Journal of Field Robotics*, vol. 24, no. 10, pp. 803–827, 2007.

[59] M. Magnusson, A. Nuchter, C. Lorken, A. J. Lilienthal, and J. Hertzberg, “Evaluation of 3d registration reliability and speed-a comparison of icp and ndt,” in *Robotics and Automation (ICRA), 2009 IEEE International Conference on*. IEEE, 2009, pp. 3907–3912.

[60] C. Ulaş and H. Temeltaş, “3d multi-layered normal distribution transform for fast and long range scan matching,” *Journal of Intelligent and Robotic systems*, vol. 71, no. 1, pp. 85–108, 2013.

[61] A. Das and S. L. Waslander, “Scan registration using segmented region growing ndt,” *The International Journal of Robotics Research (IJRR)*, vol. 33, no. 13, pp. 1645–1663, 2014.

[62] J. W. Kim and B. H. Lee, “Robust and fast 3-d scan registration using normal distributions transform with supervoxel segmentation,” *Robotica*, vol. 34, no. 7, pp. 1630–1658, 2016.

[63] A. Diosi and L. Kleeman, “Fast laser scan matching using polar coordinates,” *The International Journal of Robotics Research (IJRR)*, vol. 26, no. 10, pp. 1125–1153, 2007.

[64] S.-H. Lee, H.-C. Lee, and B.-H. Lee, “A scan restoration method for robust polar scan matching in dynamic environments,” *Advanced Robotics*, vol. 27, no. 11, pp. 877–891, 2013.

[65] S. Carpin, “Fast and accurate map merging for multi-robot systems,” *Autonomous Robots*, vol. 25, no. 3, pp. 305–316, 2008.

[66] H.-C. Lee, S.-H. Lee, M. H. Choi, and B.-H. Lee, “Probabilistic map merging for multi-robot rbpf-slam with unknown initial poses,” *Robotica*, vol. 30, no. 2, pp. 205–220, 2012.

[67] H.-C. Lee and B.-H. Lee, “Improved feature map merging using virtual supporting lines for multi-robot systems,” *Advanced Robotics*, vol. 25, no. 13-14, pp. 1675–1696, 2011.

[68] A. Censi and S. Carpin, “Hsm3d: Feature-less global 6dof scan-matching in the hough/radon domain,” in *Robotics and Automation (ICRA), 2009 IEEE International Conference on*, 2009, pp. 3899–3906.

[69] D. Holz and S. Behnke, “Sancta simplicitas-on the efficiency and achievable results of slam using icp-based incremental registration,” in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 1380–1387.

[70] T. Stoyanov, J. Saarinen, H. Andreasson, and A. J. Lilienthal, “Normal distributions transform occupancy map fusion: Simultaneous mapping and tracking in large scale dynamic environments,” in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 4702–4708.

[71] J. Zhang and S. Singh, “Visual-lidar odometry and mapping: Low-drift, robust, and fast,” in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 2174–2181.

[72] M. Magnusson, A. Nuchter, C. Lorken, A. J. Lilienthal, and J. Hertzberg, "Evaluation of 3d registration reliability and speed-a comparison of icp and ndt," in *Robotics and Automation (ICRA), 2009 IEEE International Conference on*. IEEE, 2009, pp. 3907–3912.

[73] H. Hong and B. Lee, "Key-layered normal distributions transform for point cloud registration," *Electronics Letters*, vol. 51, no. 24, pp. 1986–1988, 2015.

[74] Q. Li, R. Xiong, and T. Vidal-Calleja, "A gmm based uncertainty model for point clouds registration," *Robotics and Autonomous Systems*, vol. 91, pp. 349–362, 2017.

[75] J. P. Saarinen, H. Andreasson, T. Stoyanov, and A. J. Lilienthal, "3d normal distributions transform occupancy maps: An efficient representation for mapping in dynamic environments," *The International Journal of Robotics Research (IJRR)*, vol. 32, no. 14, pp. 1627–1644, 2013.

[76] E. Einhorn and H.-M. Gross, "Generic ndt mapping in dynamic environments and its application for lifelong slam," *IEEE Robotics & Automation Magazine*, vol. 69, pp. 28–39, 2015.

[77] R. W. Wolcott and R. M. Eustice, "Robust lidar localization using multiresolution gaussian mixture maps for autonomous driving," *The International Journal of Robotics Research (IJRR)*, vol. 36, no. 3, pp. 292–319, 2017.

[78] T. Schmiedel, E. Einhorn, and H.-M. Gross, "Iron: A fast interest point descriptor for robust ndt-map matching and its application to robot localization," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, 2015, pp. 3144–3151.

[79] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, “Octomap: An efficient probabilistic 3d mapping framework based on octrees,” *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013.

[80] G. Guennebaud, B. Jacob, *et al.*, “Eigen v3,” http://eigen.tuxfamily.org, 2018.

[81] Boost, “Boost C++ Libraries,” http://www.boost.org/, 2019, last accessed 2019-01-29.

[82] FreeGLUT, “FreeGLUT Libraries,” http://freeglut.sourceforge.net, 2019, last accessed 2019-01-29.

[83] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in *Computer Vision and Pattern Recognition, 2012. CVPR 2012. Proceedings of the 2017 IEEE Computer Society Conference on*. IEEE, 2012, pp. 3354–3361.

[84] N. Balakrishnan, “Continuous multivariate distributions,” *Wiley StatsRef: Statistics Reference Online*, 2014.

[85] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, “A benchmark for the evaluation of rgb-d slam systems,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, Vilamoura Portugal, 2012, pp. 573–580.

[86] T. Stoyanov, M. Magnusson, H. Almqvist, and A. J. Lilienthal, “On the accuracy of the 3d normal distributions transform as a tool for spatial representation,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, Shanghai China, May 2011, pp. 4080–4085.

[87] D. Dowson and B. Landau, "The fréchet distance between multivariate normal distributions," *Journal of multivariate analysis*, vol. 12, no. 3, pp. 450–455, 1982.

[88] A. E. Johnson and S. B. Kang, "Registration and integration of textured 3d data1," *Image and vision computing*, vol. 17, no. 2, pp. 135–147, 1999.

[89] H. Men, B. Gebre, and K. Pochiraju, "Color point cloud registration with 4d icp algorithm," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1511–1516.

[90] S. Druon, M.-J. Aldon, and A. Crosnier, "Color constrained icp for registration of large unstructured 3d color data sets," in *Information Acquisition, 2006 IEEE International Conference on*. IEEE, 2006, pp. 249–255.

[91] J. H. Joung, K. H. An, J. W. Kang, M. J. Chung, and W. Yu, "3d environment reconstruction using modified color icp algorithm by fusion of a camera and a 3d laser range finder," in *Intelligent Robots and Systems (IROS), 2009 IEEE/RSJ International Conference on*. IEEE, 2009, pp. 3082–3088.

[92] M. Korn, M. Holzkothen, and J. Pauli, "Color supported generalized-icp," in *Computer Vision Theory and Applications (VISAPP), 2014 International Conference on*, vol. 3. IEEE, 2014, pp. 592–599.

[93] B. Huhle, M. Magnusson, W. Straßer, and A. J. Lilienthal, "Registration of colored 3d point clouds with a kernel-based extension to the normal distributions transform," in *Robotics and Automation (ICRA), 2008 IEEE International Conference on*. IEEE, 2008, pp. 4025–4030.

[94] S. Khan, A. Dometios, C. Verginis, C. Tzafestas, D. Wollherr, and M. Buss, "Rmap: a rectagular cuboid approximatio framework for 3d eviromet mappig," *Autonomous Robots*, vol. 37, no. 3, pp. 261–277, 2014.

[95] F. Moosmann and C. Stiller, "Velodyne slam," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*. IEEE, 2011, pp. 393–398.

[96] J. Stückler and S. Behnke, "Multi-resolution surfel maps for efficient dense 3d modeling and tracking," *Journal of Visual Communication and Image Representation*, vol. 25, no. 1, pp. 137–147, 2014.

초 록

로봇은 자신이 갖춘 지능으로 스스로 작동하는 기기이다. 자율 이동 지능은 로봇이 가져야 하는 중요한 지능이다. 본 논문은 이러한 자율 이동 지능을 위한 3차원 거리 센서 기반 위치 추정 및 지도 작성을 위한 방법을 제시한다.

로봇은 거리센서를 이용하여 위치한 환경의 공간 정보를 점군(point set) 형태로 수집할 수 있는데, 이렇게 수집한 정보를 환경의 복원에 이용할 수 있다. 또한, 로봇은 점군과 모델을 정합하는 위치를 추정할 수 있다. 거리센서가 수집한 점군이 2차원에서 3차원으로 확장되고 해상도가 높아지면서 점의 개수가 크게 증가하면서, NDT (normal distributions transform)를 이용한 정합이 ICP (iterative closest point)의 대안으로 부상하였다. NDT는 점군을 분포로 변환하여 공간을 표현하는 압축된 공간 표현 방법이다. 분포의 개수가 점의 개수에 비해 월등히 작기 때문에 ICP에 비해 빠른 성능을 가졌다. 그러나 NDT 정합 기반 위치 추정의 성능을 좌우하는 셀의 크기, 셀의 중첩 정도, 셀의 방향, 분포의 스케일, 대응쌍의 비중 등 파라미터를 설정하기가 매우 어렵다. 본 학위 논문에서는 이러한 어려움에 대응하여 NDT 정합 기반 위치 추정의 정확도를 향상할 수 있는 방법을 제안하였다.

본 논문은 표현법과 정합법 2개 파트로 나눌 수 있다. 표현법에 있어 본 논문은 다음 3개 방법을 제안하였다. 첫째, 본 논문에서는 분포의 퇴화를 막기 위해 경험적으로 공분산 행렬의 고유값을 수정하여 공간적 형태의 왜곡을 가져오는 문제점과

고해상도의 NDT를 생성할 때 셀당 점의 개수가 감소하며 구조를 반영하는 분포가 형성되지 않는 문제점을 주목했다. 이를 해결하기 위하여 각 점에 대해 불확실성을 부여하고, 평균과 분산의 기대값으로 수정한 확률적 NDT (PNDT, probabilistic NDT) 표현법을 제안하였다. 공간 정보의 누락 없이 모든 점을 분포로 변환한 NDT를 통해 향상된 정확도를 보인 PNDT는 샘플링을 통한 가을을 가능하도록 하였다. 둘째, 본 논문에서는 정육면체를 셀로 다루며, 셀을 중심좌표와 변의 길이로 정의한다. 또한, 셀들로 이뤄진 격자를 각 셀의 중심점 사이의 간격과 셀의 크기로 정의한다. 이러한 정의를 토대로, 본 논문에서는 셀의 확대를 통하여 셀을 중첩시키는 방법과 셀의 간격 조절을 통하여 셀을 중첩시키는 방법을 제안하였다. 본 논문은 기존 2D NDT에서 사용한 셀의 삽입법을 주목하였다. 단순입방구조를 이루는 기존 방법 외에 면심입방구조와 체심입방구조의 셀로 이뤄진 격자가 생성하였다. 그 다음 해당 격자를 이용하여 NDT를 생성하는 방법을 제안하였다. 또한, 이렇게 생성된 NDT를 정합할 때 많은 시간을 소요하기 때문에 대응쌍 검색 영역을 정의하여 정합 속도를 향상하였다. 셋째, 저사양 로봇들은 점군 지도를 NDT 지도로 압축하여 보관하는 것이 효율적이다. 그러나 로봇 포즈가 갱신되거나, 다개체 로봇간 랑데뷰가 일어나 지도를 공유 및 결합하는 경우 NDT의 분포 형태가 왜곡되는 문제가 발생한다. 이러한 문제를 해결하기 위하여 NDT 재생성 방법을 제안하였다.

정합법에 있어 본 논문은 다음 4개 방법을 제안하였다. 첫째, 점군의 각 점에 대해 대응되는 색상 정보가 제공될 때 색상 hue를 이용한 향상된 NDT 정합으로 각 대응쌍에 대해 hue의 유사도를 비중으로 사용하는 목적함수를 제안하였다. 둘째, 본 논문은은 다양한 크기의 위치 변화량에 대응하기 위한 다중 레이어 NDT 정합 (ML-NDT, multi-layered NDT)의 한계를 극복하기 위하여 키레이어 NDT 정합 (KL-NDT, key-layered NDT)을 제안하였다. KL-NDT는 각 해상도의 셀에서 활성화된 점의 개수 변화량을 척도로 키레이어를 결정한다. 또한 키레이어에서 위치의 추정값이 수렴할 때까지 정합을 수행하는 방식을 취하여 다음 키레이어에 더 좋은

초기값을 제공한다. 셋째, 본 논문은 이산적인 셀로 인해 NDT간 정합 기법인 NDT-D2D (distribution-to-distribution NDT)의 목적 함수가 비선형이며 국소 최저치의 완화를 위한 방법으로 신규 NDT와 모델 NDT에 독립된 스케일을 정의하고 스케일을 변화하며 정합하는 동적 스케일 기반 NDT 정합 (DSF-NDT-D2D, dynamic scaling factor-based NDT-D2D)을 제안하였다. 마지막으로, 본 논문은 소스 NDT와 지도간 증대적 정합을 이용한 주행계 추정 및 지도 작성 방법을 제안하였다. 이 방법은 로봇의 현재 포즈에 대한 초기값을 소스 점군에 적용한 뒤 NDT로 변환하여 지도 상 NDT와 가능한 한 유사한 NDT를 작성한다. 그 다음 로봇 포즈 및 소스 NDT의 GC (Gaussian component)를 고려하여 부분지도를 추출한다. 이렇게 추출한 부분지도와 소스 NDT는 다중 레이어 NDT 정합을 수행하여 정확한 주행계를 추정하고, 추정 포즈로 소스 점군을 회전 및 이동 후 기존 지도를 갱신한다. 이러한 과정을 통해 이 방법은 현재 최고 성능을 가진 LOAM (lidar odometry and mapping)에 비하여 더 높은 정확도와 더 빠른 처리속도를 보였다.

주요어: normal distributions transform, registration, odometry, mapping

학번: 2013-20911

감사의 글

우선, 제가 박사 학위 논문 연구에 집중할 수 있도록 혈세로 등록금 부담을 덜어주시고 국내 최고의 연구 환경을 조성주신 국가와 국민들께 감사드립니다. 세계 최고를 목표로 기술을 연구하여 국가와 국민들께 보답하겠습니다. 또한, 본 논문을 완성할 수 있도록 지도해주신 지도 교수님, 심사해주신 교수님, 배경 지식을 가르쳐주신 교수님, 함께 고민해주신 연구실 동료, 물심양면으로 지원해주신 가족, 곁에서 응원해주신 친구들 모두에게 이 자리를 빌려 감사드립니다.