공학석사 학위논문

3-D Scan Registration Using Normal Distributions Transform with Supervoxel Segmentation

Supervoxel Segmentation기반의

Normal Distributions Transform을 이용한

3차원 스캔 정합 기술

2015년 2월

서울대학교 대학원

전기·컴퓨터공학부

김 지 웅

3-D Scan Registration Using Normal Distributions Transform with Supervoxel Segmentation

지도 교수 이 범 희

이 논문을 공학석사 학위논문으로 제출함
2015년 2월

서울대학교 대학원
전기·컴퓨터공학부
김 지 웅

김지웅의 공학석사 학위논문을 인준함
2015년 2월

위 원 장 _____ (인)

부위원장 _____ (인)

위 원 _____ (인)

Abstract

In order to use mobile robots in various applications, such as exploration, rescue, surveillance, and military, the most fundamental required capability is an autonomous navigation. Moreover, one of the most important problem of autonomous navigation is that a robot should build a map of its surroundings and identify its location on its own map, i.e., a simultaneous localization and mapping (SLAM) problem. The information about the surroundings of a robot, which is used in SLAM algorithms, has two types, sparse features and dense point clouds. However, in order to perform a detailed path planning and collision avoidance, a map with dense point clouds is necessary because dense point clouds have rich information on the surrounding obstacles. Also, the map has to be three-dimensional (3-D) so that various shapes of robots carry out wide-ranging tasks. Therefore, the SLAM algorithms using dense point clouds are required for an autonomous navigation, but in order to guarantee the performance of SLAM, a high performance 3-D scan registration algorithm is essential.

This thesis presents what is termed the supervoxel normal distributions transform (SV-NDT), a novel three-dimensional registration algorithm which improves the performance of the three-dimensional normal distributions transform (3-D NDT) significantly. The 3-D NDT partitions a model scan using a 3-D regular grid. However, generating normal distributions using the 3-D regular grid causes considerable information loss because the 3-D regular grid does not use any information pertain-

ing to the local surface structures of the model scan. The best type of surface (the constituent unit of each scan) for modeling with one normal distribution is known to be the plane. The SV-NDT reduces the loss of information using a supervoxel-generating algorithm at the partitioning stage. In addition, it uses the information of the local surface structures from the data scan by replacing the Euclidean distance with a function that uses local geometries as well as the Euclidean distance when each point in the data scan is matched to the corresponding normal distribution.

Experiments demonstrate that the use of the supervoxel-generating algorithm increases the modeling accuracy of the normal distributions and that the proposed 3-D registration algorithm outperforms the 3-D NDT and other widely used 3-D registration algorithms in terms of robustness and speed on both synthetic and real-world datasets. Additionally, the positive effect of changing the function to create correspondences on the performance of registration is also verified.

Keywords: Normal distributions transform, Scan registration, Supervoxel segmentation, SLAM, Mobile robotics

Student Number: 2013-20777

Contents

List of Tables **iv**

List of Figures **v**

1 Introduction **1**
1.1 Motivation and objectives ····· 1
1.2 Contributions ····· 4
1.3 Outline ····· 5

2 Related work **7**
2.1 Simultaneous localization and mapping ····· 7
2.2 Scan registration ····· 10
2.3 3-D point cloud segmentation ····· 13

3 Supervoxel-NDT **15**
3.2 Normal distributions transform ····· 15
3.2 Motivation ····· 23
3.3 Supervoxel-generating algorithm ····· 26
3.3.1 Initialization of a set of supervoxels ····· 27
3.3.2 Expansion of supervoxels by means of flow-constrained clustering ····· 30
3.4 Proposed method ····· 31

4 Evaluation **38**
4.1 Supervoxel-generating algorithm ····· 38
4.1.1 Speed ····· 41
4.1.2 Modeling accuracy ····· 42
4.2 Registration ····· 52
4.2.1 Synthetic dataset ····· 53
4.2.2 Real-world dataset ····· 59

5 Conclusion **70**

Reference **72**

List of Tables

4.1 Average runtime for the supervoxel generation using each kind of scans ····· 41

4.2 Detailed surroundings of the test scan pairs ····· 59

4.3 Performances of each algorithm ····· 66

4.4 Runtimes of the supervoxel-generating algorithm for each scan pair ····· 68

List of Figures

1.1 Typical applications of intelligent robots such as exploration, rescue, surveillance, and household chores. ····· 2

1.2 An example of sparse features and a dense point cloud ····· 3

1.3 An example of a 2-D and a 3-D map ····· 4

3.1 The normal distributions which employ a 3-D regular grid to transform the model scan in a structured environment from the dataset ·····24

3.2 Several examples of 2-D scans which are transformed into identical normal distributions but with different local geometries ·····25

3.3 Flowchart of the supervoxel-generating algorithm·····27

3.4 A 2-D example of generation of a set of seed voxels ·····29

3.5 Flowchart of the expansion of supervoxels using the flow-constrained clustering ·····31

3.6 An example of the traversing strategy of the flow-constrained clustering in a 2-D case ·····32

3.7 Difference Δ between and the Euclidean distance ·····34

3.8 Flowchart of the SV-NDT registration algorithm·····35

3.9 The normal distributions which employ the supervoxel-generating algorithm to transform the model scan which is used in Fig. 3.1 ··· ·····37

4.1 Scenes of the synthetic scan data ·····39

4.2 Scans created by varying the type of surfaces from the second

scene ·····40

4.3 Normal distributions created by each method according to the type of surfaces with zero noise level·····43

4.4 Normal distributions created by each method according to the type of surfaces with 10 centimeters noise level ·····44

4.5 Confusion matrix ·····47

4.6 The estimated performance indices for the supervoxel-generating algorithm and the 3-D regular grid ·····51

4.7 Registration results of each registration algorithm according to the noise levels using the type of large planar surfaces·····54

4.8 Registration results of each registration algorithm according to the noise levels using the type of large rounded surfaces ·····55

4.9 Registration results of each registration algorithm according to the noise levels using the type of large planar surfaces with objects ·····56

4.10 Registration results of each registration algorithm according to the noise levels using the type of large rounded surfaces with objects ·····57

4.11 Positions of the test scan pairs with full path of the dataset ·····59

4.12 The boxplots of the final translation errors of each registration algorithm according to the initial translation errors when the initial transformation errors consist of only the initial translation errors ·····60

4.13 The boxplots of the final rotation errors of each registration algorithm according to the initial rotation errors when the initial transformation errors consist of only the initial rotation errors 61

4.14 The boxplots of the final translation errors of each registration algorithm according to the initial translation errors when the initial transformation errors consist of both initial translation and initial rotation errors·····62

4.15 The boxplots of the final rotation errors of each registration algorithm according to the initial rotation errors when the initial transformation errors consist of both initial translation and initial rotation errors ·····63

4.16 Success rate of each algorithm for each scan pair ……66

4.17 Accuracy of each registration algorithm ……67

4.18 Runtime of each algorithm ……68

Chapter 1

Introduction

1.1 Motivation

To date, most robots are used as industrial machinery and automatically do various repetitive works in a fixed position. However, robot intelligence, which involves the ability to recognize the surroundings, to judge the situation, and to act appropriately, has been improved drastically, and robots will penetrate diverse fields in the future. The typical applications of intelligent robots are exploration, surveillance, rescue, medical care, military, and household chores, but one of the most fundamental ability for them is autonomous navigation. In addition, autonomous mobile robots need a localization algorithm which requires a map of the surroundings of the robot. However, there are many situations in which human cannot input the map, and it is not easy to update the map in real time although that is possible. Therefore, autonomous mobile robots should build maps and identify their locations on their maps. This is the simultaneous localization and mapping (SLAM)

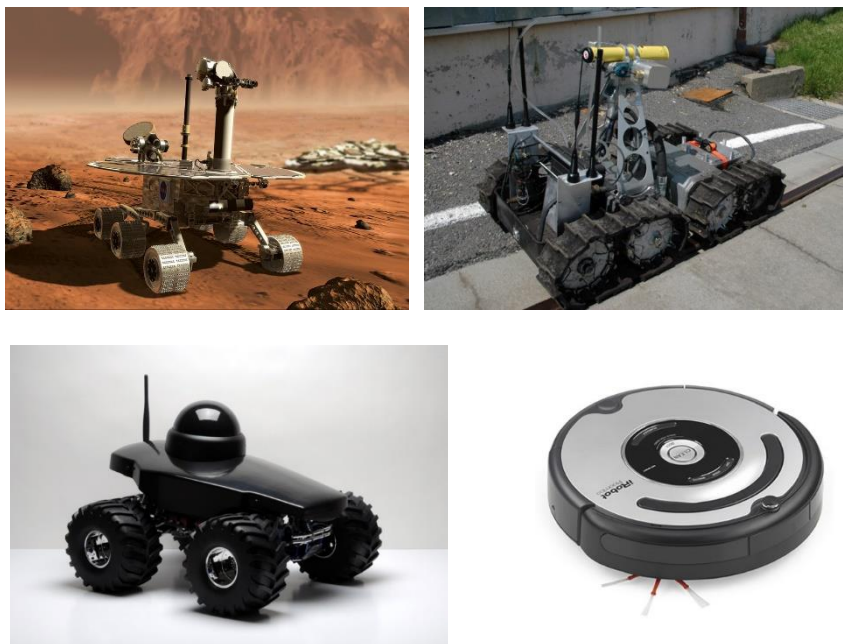

Figure 1.1: Typical applications of intelligent robots such as exploration (top left), rescue (top right), surveillance (bottom left), and household chores (bottom right).

problem which is essential for autonomous mobile robots.

There are two types of the information, sparse features and dense point clouds, about surrounding areas to perform a SLAM algorithm. The sparse features are characteristic points in the surroundings, thus algorithms which uses them are fast and can distinguish places by using only few points. On the other hand, algorithms which exploit dense point clouds are able to build a more accurate map and be robust to sensor noise because they do utilize detailed shape information and do not need a data association. Thus, SLAM algorithms that use dense point clouds are needed for path planning and collision avoidance.

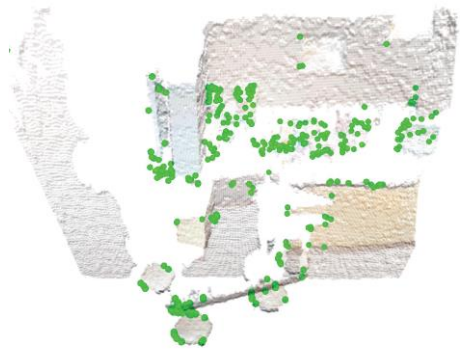

Figure 1.2: An example of sparse features (green) and a dense point cloud (rest)

Scan registration algorithms estimate the relative transformation between sensor poses, positions and orientations of sensors, in which two different scans, model scan and data scan, are captured and are necessary for the SLAM algorithm with dense point clouds. In the past, because the SLAM algorithms had been used for building two-dimensional (2-D) maps (Fig. 1.3(a)), 2-D scan registration algorithms were mainly studied. However, 2-D maps only represent a cross section of the environment, thus a robot whose height is not small cannot perform path planning with a 2-D map. In addition, 2-D maps cannot be built where the ground is not flat. For those reasons, three-dimensional (3-D) maps are essential for the sake of various applications, hence many 3-D scan registration algorithms have been studied for several years (Fig. 1.3(b)).

This thesis focuses on the normal distributions transform (NDT) algorithm [1] among various scan registration algorithms. The NDT algorithm estimates the optimal rigid transformation by using the data scan and normal distributions which

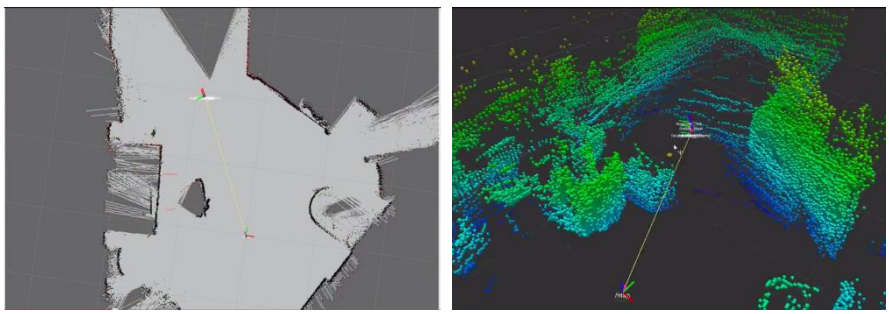

Figure 1.3: An example of a 2-D (left) and a 3-D (right) map

model the model scan. Moreover, the NDT which is a 3-D surface representation can apply to not only registration but also several applications such as occupancy mapping [2, 3], loop detection [4], and path planning [5]. These techniques based on the NDT depend heavily on the modeling accuracy of the normal distributions generated from model scan; however, they use a 3-D regular grid which does not take into account local surface structures of the model scan and partitions the model scan by cube-shaped cells. Thus, the speed of those techniques is fast, but the modeling accuracy of those techniques has limits [6]. In this thesis, we restrict an application of the NDT to the scan registration, and the objective is to increase the modeling accuracy of the normal distributions while maintaining fast speed and to improve the performance of registration.

1.2 Contributions

The objectives of this thesis are addressed, and the main contributions of this thesis include:

- The supervoxel-NDT (SV-NDT) which improves the model scan modeling accuracy of the normal distributions in the NDT algorithm by using a supervoxel-generating algorithm.
- Modified criterion to create correspondences between each point in the data scan and each normal distribution, which associates robustly by using not only a mean vector but also a normal vector of each normal distribution.
- Analysis of performances of the supervoxel-generating algorithm and the modified criterion to create correspondences between each point in the data scan and each normal distribution through several indices on the synthetic dataset which can control various variables, which verifies the improvement of each performance.
- Experimental results which demonstrate that the SV-NDT outperforms the 3-D NDT and several representative 3-D scan registration algorithms on the synthetic and real-world dataset.

1.3 Outline

This thesis is organized as follows. In Chapter 2, background information on SLAM and scan registration is described. Chapter 3 provides the problem formulation for 3-D scan registration on the basis of the 3-D NDT and the SV-NDT are presented in sequence. In Chapter 4, an analysis of the supervoxel-generating algorithm, a component of the SV-NDT, is carried out, and a performance of SV-NDT

is evaluated by comparing to other widely used 3-D registration algorithms on both synthetic and real-world dataset. Finally, this thesis is concluded in Chapter 5.

Chapter 2

Related work

2.1 Simultaneous localization and mapping

The SLAM problem is one of the most important problems in mobile robotics, thus it has studied for over two decades. Because of the inherent uncertainty in robot motions and sensor measurements, most SLAM algorithms adopt a probabilistic formulation [7]. The SLAM algorithms can be categorized into three groups.

The first is a Kalman filter-based [8] approach which assumes a motion and a measurement noise are distributed according to the Gaussian distribution. This approach is an online SLAM method, which estimates the state vector of the last time step, and the state vector is comprised of the pose of a robot and the positions of landmarks because this approach uses sparse features as the information of the surroundings. However, motion models of robots and measurement models of sensors are nonlinear, thus the SLAM algorithms of this approach should use the extended

Kalman filter (EKF). Accordingly, these algorithms are called EKF-SLAM [9, 10, 11]. The EKF-SLAM is the first solution of the SLAM problem and quite simple because the update step has a closed form solution by linearization steps and an assumption of Gaussian noises, whereas a time complexity per step and a space complexity are $O(N^2)$, where N is the number of landmarks. Thus, although the EKF-SLAM works quite well in medium-scale environments, it cannot be utilized in large-scale environments because of its quadratic complexities. However, because most pairs of landmarks are nearly conditional independent of each other, the sparse extended information filter SLAM (SEIF-SLAM) [12] solves the scalable problem by sparsification of an information matrix. Even though degrading its accuracy, the SEIF-SLAM algorithm achieves that the time complexity per step and the space complexity are $O(1)$ and $O(N)$, respectively. Nevertheless, the Kalman filter-based approach is likely to fail to build a map because of the accumulated linearization errors and the errors from the assumption that the distributions of the motion and the measurement noise are not the Gaussian distributions in reality. In addition, the performance of this approach relies heavily on a data association, thus many research has studied about a data association [13, 14].

The second is a particle filter-based [15] approach. Because the particle filter is a nonparametric filter, it can model an arbitrarily distributed noise. Also, as the particle filter is a sampling-based approach, its state space should have low dimensions to work well. However, the dimension of the state space of the SLAM algorithms which use the sparse features increases with the number of landmarks. The FastSLAM [16, 17] solves this problem by Rao-Blackwellization [18, 19]. Conse-

quently, its time complexity per step is $O(K \log N)$, where N is the number of landmarks and K is the number of particles. In addition, the FastSLAM is robust to data association ambiguities because each particle carries out a data association, individually. Furthermore, the particle filter-based SLAM algorithms which uses not sparse features but dense point clouds based on a grid map were proposed [20, 21, 22, 23].

The final approach is a graph-based SLAM [24] which is actively studied in these days. The graph-based SLAM is a full SLAM, which estimates the whole trajectory of a robot and positions of landmarks, and it solves the full SLAM problem by optimizing the pose/feature graph. In the pose/feature graph, each node represents either a pose of a robot or a position of a landmark, and each edge involves spatial information, such as a rigid transformation and a Euclidean distance with a bearing angle, between two end nodes. The graph-based SLAM was first formulated as a nonlinear optimization problem in study [25]. After that, many algorithms, such as multi-level relaxation (MLR) [26], square root smoothing and mapping (SAM) [27], tree-based network optimizer (TORO) [28], general graph optimization (g^2o) [29], and so on [30, 31], were proposed. Furthermore, some of them were extended to lifelong [32, 3] or incremental [33, 34, 35, 36] versions. Most graph-based SLAM algorithms are back-end which optimizes the graph constructed in front-end composed of loop detection [4, 37, 38, 39], scan registration, and so on. However, the back-end relies heavily on the information of the constructed graph, thus front-ends are critical for the performance of the graph-based SLAM. Hence, to overcome errors of graph from front-ends, a robust graph-based SLAM approach

was proposed [40, 41, 42, 43, 44]. In spite of that, because the robust algorithms still utilize the information of the edges judged as inliers, the significance of front-ends has not changed in order to guarantee the performance of the graph-based SLAM.

2.2 Scan registration

Scan registration algorithms allow robots to collect more information about the surrounding environment by integrating two scans taken at different times or places. Scan registration algorithms can be broadly categorized into local methods and global methods. Local methods conduct a scan registration by iteratively optimizing a cost function which represents the registration error between two scans. Given that the cost function has local minima, in most cases, the results from local methods depend on the initial transformation. The initial transformation can be obtained by odometry, an inertial measurement unit (IMU), or a GPS if this type of system is available. If the initial transformation is sufficiently close to the ground truth, local methods can estimate the relative transformation finely compared to global methods. There are various algorithms used with local methods, such as the iterative closest point (ICP) [45, 46, 47], the normal distributions transform (NDT) [1], and the polar scan matching (PSM) [48]. Global methods undertake scan registration with the distinct local geometrical features of each scan. They vary depending on their means of extracting features such as the Hough transform [49, 50], the fast point feature histogram (FPFH) [51], the phase only matched filtering (POMF) [52], and other methods.

One of the most popular scan registration algorithms is the ICP algorithm. The ICP algorithm is a point-to-point algorithm that estimates the optimal transformation to overlap two scans, a model and a data scan, by iteratively minimizing the sum of the squared Euclidean distances between the corresponding points. The ICP algorithm regards the closest points in different scans as corresponding points. Because a closed-form solution exists for optimizing the sum of the squared Euclidean distances between associated pairs, the ICP is easily implemented. Although the nearest neighbor search is a bottleneck when using the ICP due to the high computational cost, using the k-d tree [53] or approximate k-d tree [54] can mitigate this problem. However, the assumption that the closest points in different scans are the corresponding points is satisfied well only when the relative rotation difference between the two scans is small. Given a large relative rotation difference, points that are far from the sensor move far away, with many of these points not associated correctly as a result. For this reason, iterative dual correspondence (IDC) [55] generates corresponding points for rotation and translation separately and alternatively minimizes the sum of the squared Euclidean distances. Metric-based ICP (MbICP) [56, 57] establishes correspondences between two scans with a new metric which takes into account rotation as well as translation. Every scan is composed of the closest surfaces in the surroundings to the sensor. Generalized-ICP (G-ICP) [58] uses local surface structures while retaining the simplicity of the ICP. The G-ICP models the vicinity of each point as a locally planar surface by means of a covariance matrix and applies this information to the cost function to decrease the effect of incorrectly associated points.

Another algorithm for scan registration is the NDT algorithm. The NDT algorithm was initially proposed as a 2-D scan registration algorithm and was later extended to three dimensions [59]. The 3-D NDT algorithm is not a point-to-point method which performs registration between two scans directly but is instead a point-to-distribution method that carries out registration between the data scan and a set of distributions generated from the model scan. Because the NDT does not need to search for the closest points or store the raw data from the model scan, it has low computational complexity and can greatly reduce the amount of memory required. In addition, the gradient vector and the Hessian matrix of its score function have analytic forms, allowing the simple use of standard nonlinear optimization methods to estimate the optimal transformation.

However, the use of a regular grid by the NDT causes several fundamental problems. The first is the discontinuity of the score function. When a data scan point passes one of its cell boundaries, the value of the score function jumps. Because this can cause a problem, a method using trilinear interpolation between distributions within neighboring voxels, which relieves the effects of discontinuities [60], was proposed. Furthermore, an alternative method was suggested that modifies the score function so that it becomes continuous [61]. Because this method employs greedy clustering to partition the scan, there are few distributions. Thus, the modified score function includes the scores of all of the normal distributions for each point in the data scan. In addition, a distribution-to-distribution registration approach was proposed which transforms the data scan as well as the model scan into normal distributions [62]. The second fundamental problem is that the registration performance relies on the cell size of the regular grid. In one study [63], the

cell size of each cell was made to vary with the distance from the sensor in an effort to solve this problem. In addition, the use of multi-layered NDT (ML-NDT) [64] changes the cell size from large to small during the iterative optimization process. A more serious problem, though, is that a normal distribution in each voxel does not represent the local structure of the point subset within the voxel accurately because the regular grid does not take into account the structures of the model scan. When part of the scan data, i.e., that composed of surfaces, is modeled by a normal distribution, the shape of the point subset which minimizes information loss is a plane. Thus, in order to utilize the merits of the NDT and minimize the loss of information, the model scan should be divided into locally planar surfaces by means of segmentation techniques.

2.3 3-D Point cloud segmentation

Segmentation techniques for a 3-D point cloud can be divided into object-based segmentation techniques, which separate a point set based on objects, and over-segmentation techniques, which partition more finely into point subsets by gathering points that have locally similar geometries. Object-based segmentation methods mostly consist of two stages. The first stage splits the point cloud into ground points and non-ground points, and the second stage generates objects by grouping the non-ground points. In another study [65], Gaussian process incremental sample consensus (GP-INSAC) is used at the first stage, and the non-ground points are clustered according to the principle of local voxel adjacency. To increase the speed of the ground segmentation process, the point cloud is divided into sever-

al sectors and one-dimensional GP-INSAC is then carried out for each sector [66]. Another approach with which to extract ground points is a graph-based method [67]. In that study [67], segmentation is performed by means of local convexity. The segment whose normal vector has the largest z-component is regarded as the ground, and the other segments are considered as objects. For urban environments, random sample consensus (RANSAC) and a Kalman filter are used to extract the road and street furniture [68].

Over-segmentation techniques are used mainly as a preprocessing stage for image segmentation in computer vision. Well-known image segmentation approaches such as the Markov random field (MRF) and the conditional random field (CRF) usually estimate the class of each pixel. However, because most present-day images have a great number of pixels, a considerable amount of computing time is required. To reduce the number of regions to be estimated, over-segmentation techniques such as the use of the superpixels, which groups similar pixels into one region, are used. Recently, this idea was extended to three dimensions, and some algorithms were proposed in this context. In one such study [69], non-ground points are partitioned into super-segments according to connectivity and similarity in the normal direction. Another approach constructs evenly spaced supervoxels by clustering a voxel-cloud based on the similarity of local geometries [70]. Furthermore, over-segmentation is performed by spheres of different sizes that are determined by the local curvatures and densities [71].

Chapter 3

Normal Distributions Transform with Supervoxel Segmentation

3.1 Normal Distributions Transform

The goal of 3-D scan registration is to estimate the optimal transformation between two scans. The model and the data scan are denoted as $Y = \{\mathbf{y}_1, \cdots, \mathbf{y}_{N_Y}\}$ and $X = \{\mathbf{x}_1, \cdots, \mathbf{x}_{N_X}\}$ respectively, and the set of 3-D rigid transformations is $\mathbb{SE}(3)$. A 3-D rigid transformation T is composed of a translation vector $\mathbf{t} \in \mathbb{R}^3$ and a rotation matrix $\mathbf{R} \in \mathbb{SO}(3)$. A 3-D rotation matrix can be represented in various ways, such as a unit quaternion and the Euler angles. In our approach, rotation matrices are represented by the roll, pitch, and yaw convention, which is a sequence of three basic rotations about the fixed axes x, y, and z. Therefore, the transformation parameter $\mathbf{p}$ can be represented by $\begin{bmatrix} t_x & t_y & t_z & \theta_x & \theta_y & \theta_z \end{bmatrix}^T$. If a data scan point $\mathbf{x}_n$ is

transformed by $\mathrm{T}(\mathbf{p})$, it can be denoted as $\mathrm{T}(\mathbf{p}) \cdot \mathbf{x}_n$, and the transformed point $\mathbf{x}'_n$ can be written as

$$\mathbf{x}'_n = \mathrm{T}(\mathbf{p}) \cdot \mathbf{x}_n = \mathbf{R}_z(\theta_z)\mathbf{R}_y(\theta_y)\mathbf{R}_x(\theta_x)\mathbf{x}_n + \mathbf{t} \tag{3.1}$$

where $\mathbf{R}_x(\theta_x)$, $\mathbf{R}_y(\theta_y)$, and $\mathbf{R}_z(\theta_z)$ are the basic 3-D rotation matrices for rotations about each of the axes by angles θ_x, θ_y, and θ_z, respectively. In the same manner, the transformed data scan X' with the transformation parameter $\mathbf{p}$ is denoted by

$$X' = \{\mathbf{x}'_1, \cdots, \mathbf{x}'_{N_X}\} = \{\mathrm{T}(\mathbf{p}) \cdot \mathbf{x}_1, \cdots, \mathrm{T}(\mathbf{p}) \cdot \mathbf{x}_{N_X}\} \tag{3.2}$$

The 3-D NDT performs the registration between the data scan and the normal distributions generated from the model scan using a 3-D regular grid. To be more concrete from a statistical point of view, the 3-D NDT adopts the model scan as Gaussian mixture model (GMM) which can model an arbitrarily smooth probability density function and estimates the optimal transformation by maximizing the likelihood function of the transformed data scan [72]. The likelihood function of a point, $\mathbf{x}$, is defined as

$$p(\mathbf{x} \mid GMM_{\text{model}}) = \sum_{i=1}^{N} w_i \frac{1}{(2\pi)^{\frac{3}{2}} \sqrt{\det \boldsymbol{\Sigma}_i}} \exp\left(-\frac{(\mathbf{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i)}{2} \right) \tag{3.3}$$

where GMM_{model} is a set of parameters of the GMM for the model scan, N is the number of components of the GMM, and w_i, $\boldsymbol{\mu}_i$, and $\boldsymbol{\Sigma}_i$ are the weight, the mean vector, and the covariance matrix of the *i*th component of the GMM, respectively. The likelihood function of the transformed data scan X' can then be represented by a function of the transformation parameter $\mathbf{p}$:

$$\Psi(\mathbf{p}) = \mathrm{p}(X' \mid GMM_{\text{model}}) = \prod_{n=1}^{N_X} \mathrm{p}(\mathbf{x}'_n \mid GMM_{\text{model}}) \tag{3.4}$$

where $\mathbf{x}'_n$ is the *n*th point in the data scan and N_X is the number of points in the data scan. The optimal transformation parameter can be estimated by means of the maximum likelihood estimation (MLE) for Eq. (3.4). This is equivalent to minimizing the negative log-likelihood which is given by

$$\bar{\Psi}(\mathbf{p}) = -\log \Psi(\mathbf{p}) = -\sum_{n=1}^{N_X} \log p(\mathbf{x}'_n \mid GMM_{\text{model}}) \tag{3.5}$$

In order to perform the MLE, GMM_{model} needs to be obtained from the model scan. Usually, when the observed data and the number of components of a GMM are given, the set of parameters GMM_{model} is computed by means of an expectation-maximization (EM) algorithm. However, GMM_{model} is obtained without complicated calculations by several simplifications in the 3-D NDT. First, the model

scan Y is divided into the point subsets $Y_n = \{\mathbf{y} \in Y : \mathbf{y} \in v_n\}$ contained within the voxel v_n using a 3-D regular grid. The mean vector $\boldsymbol{\mu}_n$ and the covariance matrix $\boldsymbol{\Sigma}_n$ of each voxel v_n are then calculated by

$$\boldsymbol{\mu}_n = \frac{1}{|Y_n|} \sum_{k:\mathbf{y}_k \in Y_n} \mathbf{y}_k \tag{3.6}$$

$$\boldsymbol{\Sigma}_n = \frac{1}{|Y_n| - 1} \sum_{k:\mathbf{y}_k \in Y_n} (\mathbf{y}_k - \boldsymbol{\mu}_n)(\mathbf{y}_k - \boldsymbol{\mu}_n)^T \tag{3.7}$$

Because a covariance matrix of fewer than four points is always singular, voxels which have fewer than four points are regarded as unoccupied. The normal distribution in each occupied voxel models the local geometry within each voxel and is considered as a component of the GMM. The weight w_i of each component is set to $1/N_X$. However, for the sake of simplicity, the likelihood of each $\mathbf{x}'_n$ takes into account only the closest component of the GMM from $\mathbf{x}'_n$. The simplified negative log-likelihood function can be written as

$$\tilde{\Psi}(\mathbf{p}) = -\sum_{n=1}^{N_X} \log p(\mathbf{x}'_n \mid \boldsymbol{\mu}_{c_n}, \boldsymbol{\Sigma}_{c_n}) = \sum_{n=1}^{N_X} \frac{(\mathbf{x}'_n - \boldsymbol{\mu}_{c_n})^T \boldsymbol{\Sigma}_{c_n}^{-1} (\mathbf{x}'_n - \boldsymbol{\mu}_{c_n})}{2} + const \tag{3.8}$$

where c_n is the index of the closest component of the GMM from $\mathbf{x}'_n$. Because Eq. (3.8) is proportional to the sum of the squared distances between each trans-

formed point $\mathbf{x}'_n$ in the data scan and the mean vector $\boldsymbol{\mu}_{c_n}$ of the corresponding distribution of $\mathbf{x}'_n$, it is sensitive to outliers. This problem is alleviated by substituting the likelihood function of each point with a normal-uniform mixture distribution [73]:

$$\hat{p}(\mathbf{x}'_n \mid \boldsymbol{\mu}_{c_n}, \boldsymbol{\Sigma}_{c_n}) = \xi_1 \exp\left(-\frac{(\mathbf{x}'_n - \boldsymbol{\mu}_{c_n})^T \boldsymbol{\Sigma}_{c_n}^{-1} (\mathbf{x}'_n - \boldsymbol{\mu}_{c_n})}{2} \right) + \xi_2 p_o \qquad (3.9)$$

where ξ_1 and ξ_2 are constants such that Eq. (3.9) integrates over the c_n th voxel to one, and p_o is the expected ratio of the outliers. The negative log-likelihood of Eq. (3.9) is somewhat complicated and can therefore also be approximated by

$$\begin{aligned} \log \hat{p}(\mathbf{x}'_n \mid \boldsymbol{\mu}_{c_n}, \boldsymbol{\Sigma}_{c_n}) &= \log\left(\xi_1 \exp\left(-\frac{(\mathbf{x}'_n - \boldsymbol{\mu}_{c_n})^T \boldsymbol{\Sigma}_{c_n}^{-1} (\mathbf{x}'_n - \boldsymbol{\mu}_{c_n})}{2} \right) + \xi_2 p_o \right) \\ &\approx d_1 \exp\left(-d_2 \frac{(\mathbf{x}'_n - \boldsymbol{\mu}_{c_n})^T \boldsymbol{\Sigma}_{c_n}^{-1} (\mathbf{x}'_n - \boldsymbol{\mu}_{c_n})}{2} \right) + d_3 \end{aligned} \qquad (3.10)$$

where d_1, d_2, and d_3 are constants which are determined by ξ_1, ξ_2, and p_o. After dropping the constant d_3, which plays no role in the optimization process, the score of *n*th data scan point can be defined as

$$\mathrm{s}(\mathbf{p}, \mathbf{x}_n, c_n) = d_1 \exp\left(-d_2 \frac{(\mathbf{x}'_n - \boldsymbol{\mu}_{c_n})^T \boldsymbol{\Sigma}_{c_n}^{-1} (\mathbf{x}'_n - \boldsymbol{\mu}_{c_n})}{2} \right) \tag{3.11}$$

Finally, the cost function of the 3-D NDT, which is the negative of the score function, can be defined as

$$\mathrm{cost}(\mathbf{p}) = -\sum_{n=1}^{N_X} \mathrm{s}(\mathbf{p}, \mathbf{x}_n, c_n) \tag{3.12}$$

The optimal $\mathbf{p}$ can be estimated by minimizing Eq. (3.12). Given that the first and second derivatives of this cost function have analytic forms, a standard nonlinear optimization algorithm can easily be used, such as Newton's method or the Levenberg-Marquardt method. Newton's method is used in our approach.

Newton's method minimizes the cost function by updating arguments iteratively. Each update is performed to optimize the second-order approximation of the cost function; therefore, Newton's step $\Delta\mathbf{p}$ can be calculated by solving the following equation:

$$\mathbf{H}\Delta\mathbf{p} = -\mathbf{g} \tag{3.13}$$

where $\mathbf{g}$ and $\mathbf{H}$ are the gradient vector and the Hessian matrix of the cost function at $\mathbf{p}$, respectively. The step size γ is obtained by a line search, after which $\mathbf{p}$ is updated by

$$\mathbf{p} \leftarrow \mathbf{p} + \gamma \Delta \mathbf{p} \tag{3.14}$$

These two steps, computing $\Delta \mathbf{p}$ and γ at $\mathbf{p}$ and updating $\mathbf{p}$, are iterated until $\mathbf{p}$ converges.

The *i*th component of $\mathbf{g}$ and the entry in the *i*th row and *j*th column of $\mathbf{H}$ are given by

$$g_i = \frac{\partial \text{cost}}{\partial p_i} = -\sum_{n=1}^{N_X} \frac{\partial \text{s}_n}{\partial p_i} = d_2 \sum_{n=1}^{N_X} (\mathbf{x}'_n - \boldsymbol{\mu}_{c_n})^T \boldsymbol{\Sigma}_{c_n}^{-1} \frac{\partial \mathbf{x}'_n}{\partial p_i} \text{s}_n \tag{3.15}$$

$$\begin{aligned} H_{ij} = \frac{\partial^2 \text{cost}}{\partial p_i \partial p_j} = -\sum_{n=1}^{N_X} \frac{\partial^2 \text{s}_n}{\partial p_i \partial p_j} = d_2 \sum_{n=1}^{N_X} \Big(& \frac{\partial \mathbf{x}'^T_n}{\partial p_i} \boldsymbol{\Sigma}_{c_n}^{-1} \frac{\partial \mathbf{x}'_n}{\partial p_j} + (\mathbf{x}'_n - \boldsymbol{\mu}_{c_n})^T \\ & \boldsymbol{\Sigma}_{c_n}^{-1} \frac{\partial^2 \mathbf{x}'_n}{\partial p_i \partial p_j} - d_2 (\mathbf{x}'_n - \boldsymbol{\mu}_{c_n})^T \boldsymbol{\Sigma}_{c_n}^{-1} \frac{\partial \mathbf{x}'_n}{\partial p_i} (\mathbf{x}'_n - \boldsymbol{\mu}_{c_n})^T \boldsymbol{\Sigma}_{c_n}^{-1} \frac{\partial \mathbf{x}'_n}{\partial p_j} \Big) \text{s}_n \end{aligned} \tag{3.16}$$

where $\text{s}_n = \text{s}(\cdot, \mathbf{x}_n, c_n)$ is the score of the *n*th point of the data scan, $\partial \mathbf{x}'_n / \partial p_i$ is the *i*th column of the Jacobian matrix $\partial \mathbf{x}'_n / \partial \mathbf{p}$ of $\mathbf{x}'_n$, and $\partial^2 \mathbf{x}'_n / \partial p_i \partial p_j$ is the entry in the *i*th row and *j*th column of the Hessian matrix of $\mathbf{x}'_n$. $\partial \mathbf{x}'_n / \partial \mathbf{p}$ and $\partial^2 \mathbf{x}'_n / \partial p_i \partial p_j$ are described by the following equations:

$$\frac{\partial \mathbf{x}'_n}{\partial \mathbf{p}} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{j}_{rx} & \mathbf{j}_{ry} & \mathbf{j}_{rz} \end{bmatrix} \tag{3.17}$$

$$\mathbf{j}_{rx} = \begin{bmatrix} (\mathbf{x}_n)_y (s_x s_z + c_x s_y c_z) + (\mathbf{x}_n)_z (c_x s_z - s_x s_y c_z) \\ -(\mathbf{x}_n)_y (s_x c_z - c_x s_y s_z) - (\mathbf{x}_n)_z (c_x c_z + s_x s_y s_z) \\ (\mathbf{x}_n)_y c_x c_y - (\mathbf{x}_n)_z s_x c_y \end{bmatrix} \tag{3.18}$$

$$\mathbf{j}_{ry} = \begin{bmatrix} -(\mathbf{x}_n)_x s_y c_z + (\mathbf{x}_n)_y s_x c_y c_z + (\mathbf{x}_n)_z c_x c_y c_z \\ -(\mathbf{x}_n)_x s_y s_z + (\mathbf{x}_n)_y s_x c_y s_z + (\mathbf{x}_n)_z c_x c_y s_z \\ -(\mathbf{x}_n)_x c_y - (\mathbf{x}_n)_y s_x s_y - (\mathbf{x}_n)_z c_x s_y \end{bmatrix} \tag{3.19}$$

$$\mathbf{j}_{rz} = \begin{bmatrix} -(\mathbf{x}_n)_x c_y s_z - (\mathbf{x}_n)_y (c_x c_z + s_x s_y s_z) + (\mathbf{x}_n)_z (s_x c_z - c_x s_y s_z) \\ (\mathbf{x}_n)_x c_y c_z - (\mathbf{x}_n)_y (c_x s_z - s_x s_y c_z) + (\mathbf{x}_n)_z (s_x s_z + c_x s_y c_z) \\ 0 \end{bmatrix} \tag{3.20}$$

$$\frac{\partial^2 \mathbf{x}'_n}{\partial p_i \partial p_j} = \begin{cases} \begin{bmatrix} (\mathbf{x}_n)_y (c_x s_z - s_x s_y c_z) - (\mathbf{x}_n)_z (s_x s_z + c_x s_y c_z) \\ -(\mathbf{x}_n)_y (c_x c_z + s_x s_y s_z) + (\mathbf{x}_n)_z (s_x c_z - c_x s_y s_z) \\ -(\mathbf{x}_n)_y s_x c_y - (\mathbf{x}_n)_z c_x c_y \end{bmatrix} & , (i,j) = (4,4) \\ \begin{bmatrix} (\mathbf{x}_n)_y c_x c_y c_z - (\mathbf{x}_n)_z s_x c_y c_z \\ (\mathbf{x}_n)_y c_x c_y s_z - (\mathbf{x}_n)_z s_x c_y s_z) \\ -(\mathbf{x}_n)_y c_x s_y + (\mathbf{x}_n)_z s_x s_y \end{bmatrix} & , (i,j) = (4,5) \text{ or } (5,4) \\ \begin{bmatrix} (\mathbf{x}_n)_y (s_x c_z - c_x s_y s_z) + (\mathbf{x}_n)_z (c_x c_z + s_x s_y s_z) \\ (\mathbf{x}_n)_y (s_x s_z + c_x s_y c_z) + (\mathbf{x}_n)_z (c_x s_z - s_x s_y c_z) \\ 0 \end{bmatrix} & , (i,j) = (4,6) \text{ or } (6,4) \\ \begin{bmatrix} -(\mathbf{x}_n)_x c_y c_z - (\mathbf{x}_n)_y s_x s_y c_z - (\mathbf{x}_n)_z c_x s_y c_z \\ -(\mathbf{x}_n)_x c_y s_z - (\mathbf{x}_n)_y s_x s_y s_z - (\mathbf{x}_n)_z c_x s_y s_z \\ (\mathbf{x}_n)_x s_y - (\mathbf{x}_n)_y s_x c_y - (\mathbf{x}_n)_z c_x c_y \end{bmatrix} & , (i,j) = (5,5) \\ \begin{bmatrix} (\mathbf{x}_n)_x s_y s_z - (\mathbf{x}_n)_y s_x c_y s_z) - (\mathbf{x}_n)_z c_x c_y s_z \\ -(\mathbf{x}_n)_x s_y c_z + (\mathbf{x}_n)_y s_x c_y c_z) + (\mathbf{x}_n)_z c_x c_y c_z \\ 0 \end{bmatrix} & , (i,j) = (5,6) \text{ or } (6,5) \\ \begin{bmatrix} -(\mathbf{x}_n)_x c_y c_z + (\mathbf{x}_n)_y (c_x s_z - s_x s_y c_z) - (\mathbf{x}_n)_z (s_x s_z + c_x s_y c_z) \\ -(\mathbf{x}_n)_x c_y s_z - (\mathbf{x}_n)_y (c_x c_z + s_x s_y s_z) + (\mathbf{x}_n)_z (s_x c_z - c_x s_y s_z) \\ 0 \end{bmatrix} & , (i,j) = (6,6) \\ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} & , \textit{else} \end{cases} \tag{3.21}$$

where $s_x = \sin(\theta_x)$, $s_y = \sin(\theta_y)$, $s_z = \sin(\theta_z)$, $c_x = \cos(\theta_x)$, $c_y = \cos(\theta_y)$, and $c_z = \cos(\theta_z)$.

3.2 Motivation

The 3-D NDT algorithm does not use the model scan directly but instead employs normal distributions that are generated from the model scan. A mean vector and a covariance matrix can represent all of the characteristics of a normal distribution; in other words, an ellipsoid is able to contain the entire set of information pertaining to a normal distribution. When a surface is modeled as a normal distribution, the lower its curvature is, the less information it loses. A scan is composed of the closest surfaces in each direction from the sensor; therefore, the scan should be partitioned into planar surfaces to reduce the loss of information. If a scan is transformed into normal distributions in the manner of the 3-D NDT, which uses a 3-D regular grid without considering the surface structures of the model scan, there will be some distributions whose mean vectors are not on any surfaces modeled by those distributions, or whose normal vectors, i.e., the eigenvectors corresponding to the minimum eigenvalue of each covariance matrix, are not parallel to any normal directions of surfaces modeled by those distributions (Fig. 3.1). This means that the voxels which have those distributions contain surfaces having different local geometries concurrently, and those normal distributions represent the average local geometry. As a result, the surfaces which belong to such voxels cannot be specified precisely due to the loss of information. Figure 3.2 shows several examples of two-dimensional scans which are transformed into identical normal distributions but with different local geometries. Decreasing the size of the voxels in order to allevi ate this problem increases the computational load owing to the increase in the

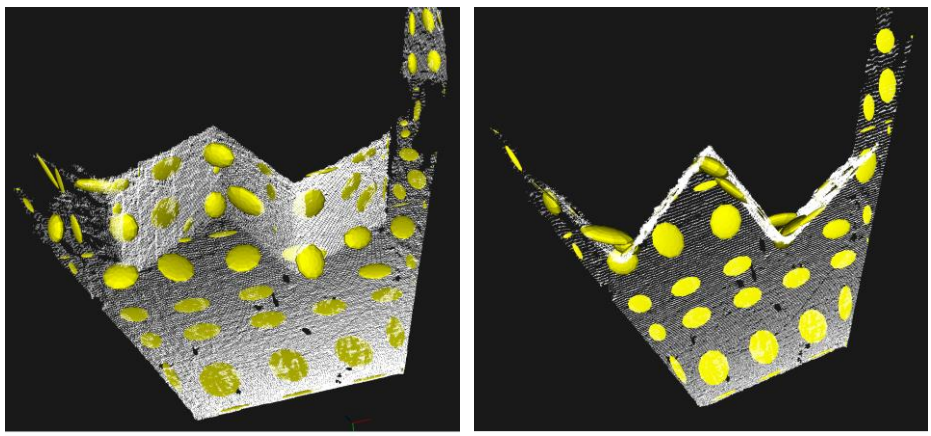

Figure 3.1: The normal distributions which employ a 3-D regular grid to transform the model scan in a structured environment from the dataset [74]. Each ellipsoid represents a normal distribution, and its surface is a set of the points whose Mahalanobis distance from the mean is equal to 1. Most of the distributions around the corners cannot model the local surfaces. (a) Front view. (b) Top view.

number of distributions. Moreover, the performance of the 3-D NDT can be diminished due to the increased number of unoccupied voxels. Thus, a segmentation technique which partitions the model scan into planar surfaces using the information of the local surface structures is required.

Researchers have recently proposed several scan registration algorithms which use object-based segmentation methods [75, 76]. If a model scan is partitioned by means of object-based segmentation instead of the 3-D regular grid in the 3-D NDT, each object is modeled by one normal distribution. However, when an object has complex structures, it cannot be represented accurately by one normal distribution. achieve this goal, over-segmentation techniques are required. The segmentation

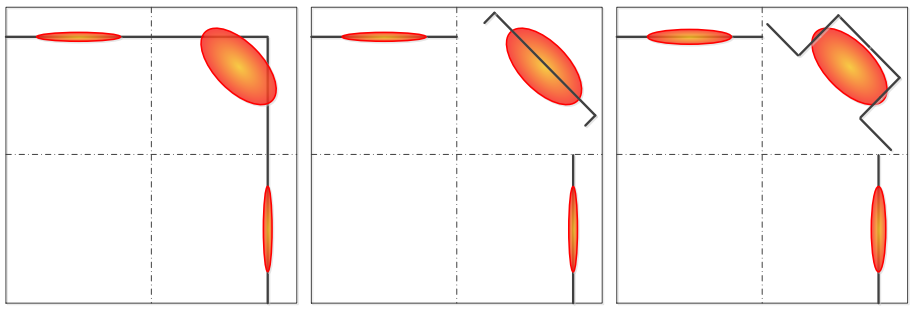

Figure 3.2: Several examples of 2-D scans which are transformed into identical normal distributions but with different local geometries

algorithm in one study [71] creates spherical segments with radii determined by local convexities and densities. However, the spherical shape of each segment is not As mentioned earlier, in order for the model scan to be modeled by normal distributions accurately, it should be partitioned into locally planar surfaces. To appropriate for representing a local surface. In another study [69], tiny clusters are constructed using only the Euclidean distances between points, with the clusters then grouped into segments after a comparison of features of the local geometry, such as the connectivity and normal directions. Although this segmentation method can partition the model scan into planar surfaces, it is possible that a segment would be very large, such as the case of a wall. When this segment is modeled by one normal distribution, it is impossible to detect holes, such as windows and open doors. In other work [70], a voxel-cloud is formed using a 3-D regular grid with small voxel size at the beginning. Then, evenly spaced seed voxels are selected, and each seed voxel is considered as a supervoxel. Finally, flow-constrained clustering which expands each supervoxel by including voxels that have locally similar geometries is per-

formed. The sizes of the supervoxels generated when using this method are limited to a certain upper bound; thus, the supervoxels consist of only local points. In this regard, this 3-D supervoxel-generating algorithm is consistent with the goal of dividing the model scan into locally planar surfaces. Therefore, the model scan is partitioned by the 3-D supervoxel-generating algorithm in our approach.

3.3 Supervoxel-generating algorithm

The 3-D supervoxel-generating algorithm in an aforementioned study [70] which uses voxel cloud connectivity segmentation (VCCS) was proposed to carry out the segmentation of a 3-D colored scan from a RGB-D camera. However, 3-D scans dealt with by our approach are only non-colored; thus, the supervoxel-generating algorithm is modified so that it does not use color. In addition, although the original VCCS uses the FPFH as the local geometric information within each voxel, this information is simplified, taking the form of a normal vector. Along with these simplifications, a flowchart of the method used to generate the supervoxels is shown in Fig. 3.3. A 3-D scan is transformed into a voxel-cloud with a voxel resolution of R_v, after which a supervoxel set is initialized using seed voxels which are spaced about a seed resolution that is R_s apart. Next, all supervoxels are expanded by flow-constrained clustering with an adjacency graph. After that, the mean and recalculated. These two steps, an expansion and a recalculation step, are repeated n times, and the supervoxel set is finally obtained. The number of repetitions n is defined as $\lfloor \sqrt{3} R_s / R_v \rfloor$ so that every occupied voxel is contained in one of the

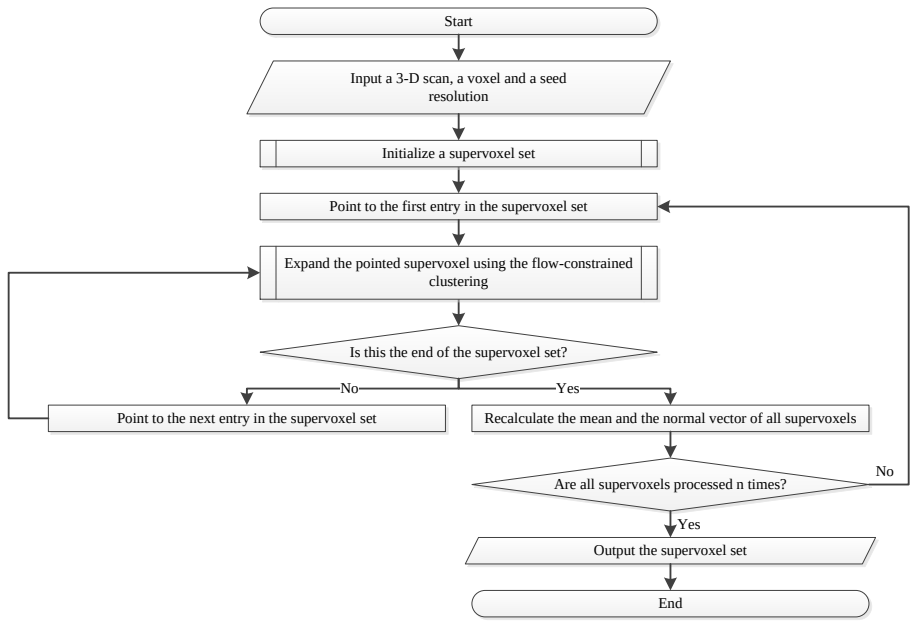

Figure 3.3: Flowchart of the supervoxel-generating algorithm

supervoxels. In this algorithm, the seed resolution R_s denotes the voxel size of the 3-D regular grid in the 3-D NDT. The voxel resolution R_v should be sufficiently smaller than R_s. R_v is set to $R_s / 10$ in our approach. Detailed descriptions of two sub-processes, the initialization of a set of supervoxels and the expansion of each supervoxel by means of flow-constrained clustering, are presented subsections 2.3.1 and 2.3.2, respectively.

3.3.1 Initialization of a set of supervoxels

First, a voxel-cloud $V = \{v_1, \cdots, v_{N_V}\}$ is generated from the 3-D scan X and the 3-D regular grid G^v_{grid} with a voxel resolution of R_v. Each v_i is an occupied voxel which contains more than three points and has indices of points contained within it. After that, the data of all voxels v_i in V are initialized. The initialization stage of the voxel data calculates the mean vector $\boldsymbol{\mu}_i$ and normal vector $\mathbf{n}_i$ and sets d_i to zero. d_i is the distance between v_i and the supervoxel containing v_i. Next, the adjacency graph $AG = (U, E)$ of the voxel-cloud V is generated. A set of nodes is denoted as $U = \{1, \cdots, N_V\}$, and each node *i* corresponds to the *i*th voxel v_i. The undirected edge set E of AG is defined as follows:

$$E = \{(i, j) \mid v_j \in V \bigcap \mathrm{N}_{G^v_{grid}}(v_i);\ v_i \in V\} \tag{3.22}$$

where $\mathrm{N}_{G^v_{grid}}(v_i)$ is a set of 26 neighbors of v_i in G^v_{grid}. Then, a set of seed voxels spaced at intervals of approximately the seed resolution R_s, denoted as $V_{seed} = \{v_{s_1}, \cdots v_{s_{N_S}}\}$, is created. Initially, a new 3-D regular grid G^s_{grid} is generated with the seed resolution of R_s. Next, the initial seed voxels which contain the center of the occupied voxels in G^s_{grid} are selected in G^v_{grid}. The closest occupied voxel in G^v_{grid} from each initial seed voxel is then considered as the final seed voxel. A 2-D example is shown in Fig. 3.4. After that, N_S supervoxels are created and initialized. The data of the *i*th supervoxel to initialize are sv_i, $\mathbf{M}_i$, $\mathbf{N}_i$, and Q_i.

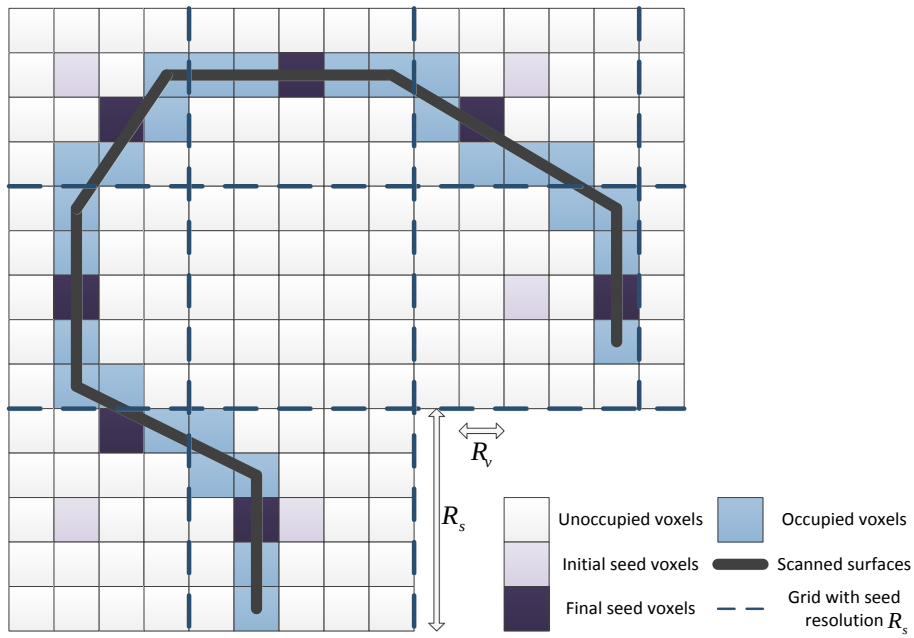

Figure 3.4: A 2-D example of generation of a set of seed voxels

sv_i is a set of voxels belonging to the *i*th supervoxel, and $\mathbf{M}_i$ and $\mathbf{N}_i$ are respectively a mean vector and a normal vector which are calculated by all points contained within the voxels in sv_i. Q_i is the set of voxels to visit at the current level in the breadth-first search (BFS) of the *i*th supervoxel. In order to perform flow-constrained clustering, the BFS traverses AG from the node corresponding to the *i*th seed voxel v_{s_i}. To initialize the clustering process, all instances of sv_i and Q_i are set to $\{v_{s_i}\}$ and the voxels adjacent to v_{s_i} in the AG respectively, and $\mathbf{M}_i$ and $\mathbf{N}_i$ are calculated.

3.3.2 Expansion of supervoxels by means of flow-constrained clustering

The flow-constrained clustering stage is based on the BFS of each supervoxel. This clustering stage proceeds in a breadth-first fashion, which means that the level to traverse increases after every BFS traverses the same level in turn. A flowchart of this sub-process is shown in Fig. 3.5. A new queue, $\tilde{Q}_i$, is the set of voxels to visit at the next level in the BFS for the *i*th supervoxel; it is emptied before each level is traversed. A detailed example of the traversing strategy is shown in Fig. 3.6. The clustering algorithm visits all of the adjacent voxels v_j of every voxel v in Q_i, and each v_j is dealt with by dividing it into three cases. The first case is one in which v_j does not belong to any supervoxel. In this case, sv_i and $\tilde{Q}_i$ include v_j, and d_j is calculated. The distance function between the *i*th supervoxel and v_j is defined as

$$\mathrm{D}(i,j) = \frac{\left\| \mathbf{M}_i - \boldsymbol{\mu}_j \right\|}{R_s} + (1 - \mathbf{N}_i \cdot \mathbf{n}_j) \tag{3.23}$$

The second case is that v_j already belongs to the *i'*th supervoxel, but the *i*th supervoxel is closer to v_j than the *i'*th supervoxel. In this case, v_j moves from the *i'*th supervoxel to the *i*th supervoxel with the update of sv_i, $\tilde{Q}_i$, $sv_{i'}$, $\tilde{Q}_{i'}$, and d_j. The last case is the final case, during which all of the clustering data remains

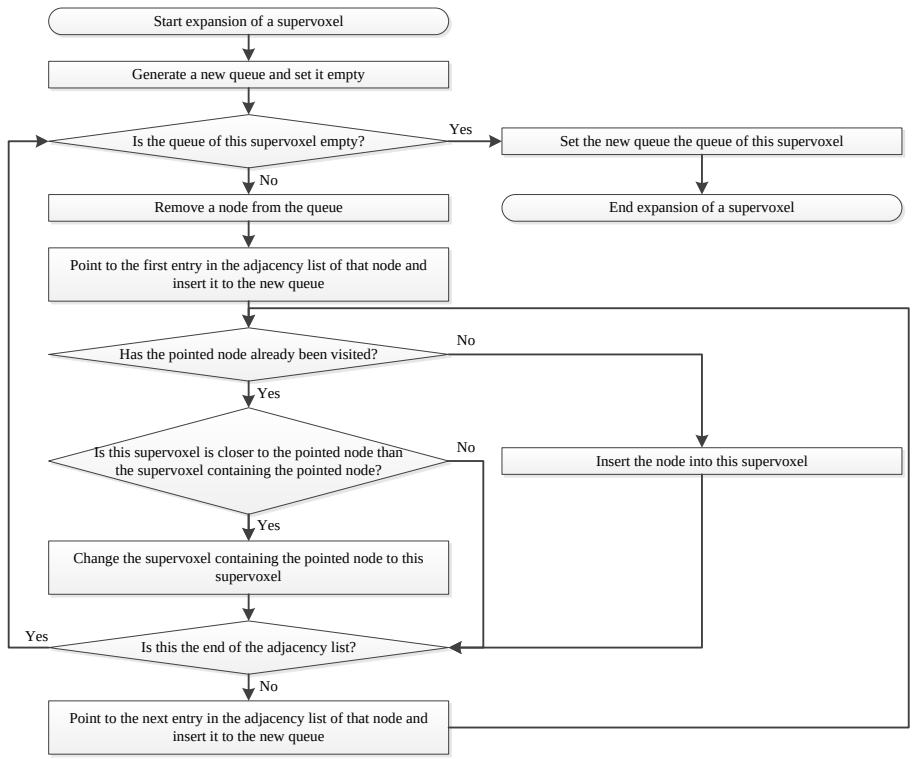

Figure 3.5: Flowchart of the expansion of supervoxels using the flow-constrained clustering

unchanged. After visiting all voxels in from Q_1 to Q_{N_S}, each Q_i is replaced with $\tilde{Q}_i$.

3.4 Supervoxel-NDT

The SV-NDT partitions the model scan effectively with the supervoxel- generating algorithm presented in the previous section instead of a 3-D regular grid,

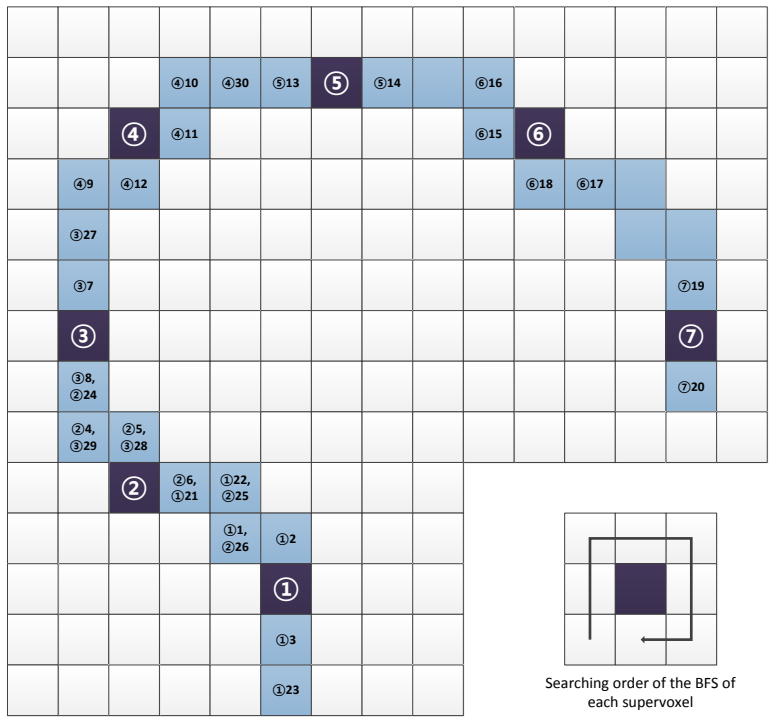

Figure 3.6: An example of the traversing strategy of the flow-constrained clustering in a 2-D case. The circled number within each seed voxel represents the index of each supervoxel. The pairs of a circled number and a number in each occupied voxel, ⓘj, represent that this voxel is visited by the *i*th supervoxel at the *j*th traversing step. The traversing steps are written up to 30 steps in this figure. The voxels in the same level of each BFS are visited clockwise direction from left-bottom corner in this figure.

which does not take into account the surface structures of the model scan. This reduces the loss of information when approximating the local geometries of the model scan to ellipsoids.

In addition, the criterion to create correspondences between each point in the data scan and each normal distribution is modified. In the 3-D NDT, each point in

the data scan corresponds to the normal distribution of the voxel containing it. If a point does not belong to an occupied voxel, it corresponds to the normal distribution of the closest occupied voxel. Similarly, each point in the data scan corresponds to its closest normal distribution in the SV-NDT as well. However, the distance function is not the Euclidean distance but the new distance function Δ. Δ, taking into account not only the Euclidean distance but the similarity of the normal vectors between the *i*th point in the data scan and the normal distribution of the *j*th supervoxel, is defined as

$$\Delta(i, j, \mathbf{p}) = \left(1 - \log_2 \left(1 - \frac{\arccos \left| \mathbf{n}_{\mathbf{x}'_i} \cdot \mathbf{n}_{Y_j} \right|}{\frac{\pi}{2}} \right) \right) \cdot \left\| \mathbf{x}'_i - \boldsymbol{\mu}_j \right\|_2 \qquad (3.24)$$

where $\mathbf{x}'_i$ is the *i*th point in the transformed data scan X', $\boldsymbol{\mu}_j$ is the mean vector of the point subset Y_j within the *j*th supervoxel, $\mathbf{n}_{\mathbf{x}'_i}$ is the normal vector of X' at $\mathbf{x}'_i$, $\mathbf{n}_{Y_j}$ is the normal vector of the surface within the *j*th supervoxel, and $\mathbf{p}$ is the current transformation parameter. $\mathbf{n}_{\mathbf{x}'_i}$ and $\mathbf{n}_{Y_j}$ are the normalized eigenvectors associated with the smallest eigenvalue of the covariance matrix of Y_j and the points in X' around $\mathbf{x}'_i$, respectively. The ratio of Δ to the Euclidean distance at each included angle between $\mathbf{n}_{\mathbf{x}'_i}$ and $\mathbf{n}_{Y_j}$ is shown in Fig. 3.7(a). When the included angle equals $\pi / 4$, Δ is equal to double the Euclidean distance, and when the included angle equals $\pi / 2$, Δ goes to infinity. Therefore, the distribution of

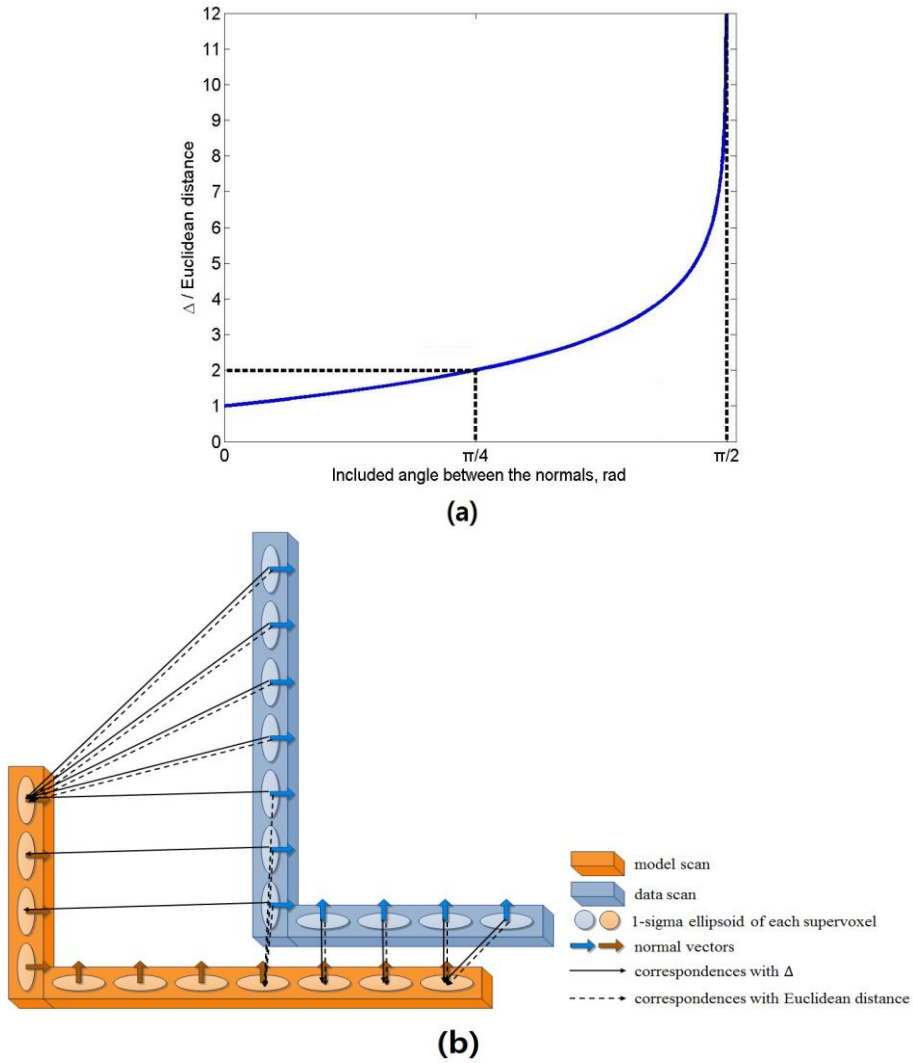

Figure 3.7: Difference between Δ and the Euclidean distance. (a) Relationship between the ratio of Δ to the Euclidean distance and the included angle between the normal vectors. (b) Comparison between correspondences generated by using Δ and the Euclidean distance.

the surface which is perpendicular to the surface around $\mathbf{x}'_i$ cannot be the corresponding distribution. A simple example of this effect is shown in Fig. 3.7 (b).

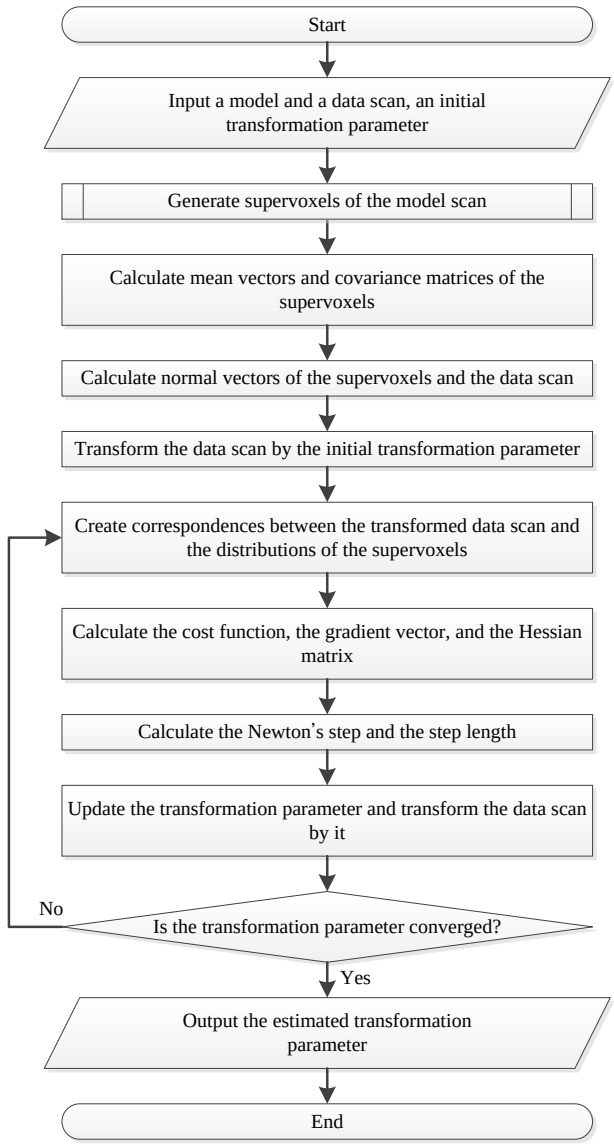

Figure 3.8: Flowchart of the SV-NDT registration algorithm

A flowchart of the SV-NDT registration algorithm is shown in Fig. 3.8. First, a supervoxel set SV is obtained by applying the supervoxel-generating algorithm to the model scan Y. Next, the mean vector and the covariance matrix of the point

subset within each supervoxel are calculated. This process is carried out using Eqs. (3.6) and (3.7) with $Y_n = \{\mathbf{y} \in Y \mid \mathbf{y} \in \bigcup_{k:v_k \in sv_n} v_k\}$ for the nth supervoxel. After that, each normal vector $\mathbf{n}_{\mathbf{x}_i}$ of X at $\mathbf{x}_i$ and each normal vector $\mathbf{n}_{Y_j}$ of Y_j is calculated. Then, the data scan is transformed by the initial transformation parameter. After the initialization phase, the iterative optimization phase begins. The first step is to calculate the indices of the corresponding normal distributions of the transformed data scan X'. The second step is to calculate the cost function f, the gradient vector $\mathbf{g}$, and the Hessian matrix $\mathbf{H}$. Next, Newton's step $\Delta\mathbf{p}$ and the step size γ are calculated, after which $\mathbf{p}$ is updated by Eqs. (3.13) and (3.14). Subsequently, each transformed point $\mathbf{x}'_i$ with its normal vector $\mathbf{n}_{\mathbf{x}'_i}$ is calculated with the current transformation parameter $\mathbf{p}$. The normal distributions of the scan used in Fig. 3.1, which is generated by the supervoxel-generating algorithm are shown in Fig. 3.9. The surface within each supervoxel is mostly flat; thus, many normal distributions are near-degenerate. We denote the eigenvalues of the covariance matrix of each normal distribution as λ_1, λ_2, and λ_3, satisfying $\lambda_1 \geq \lambda_2 \geq \lambda_3$. Then, if λ_2 is less than $\lambda_1 / 10$, λ_2 and λ_3 are replaced with $\lambda_1 / 10$. Also, if λ_2 is greater than or equal to $\lambda_1 / 10$ and λ_3 is less than $\lambda_1 / 10$, only λ_3 is replaced with $\lambda_1 / 10$.

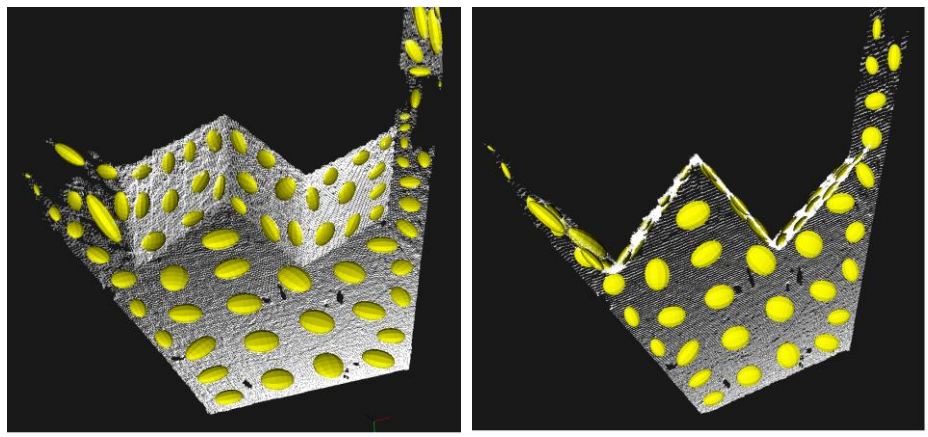

Figure 3.9: The normal distributions which employ the supervoxel-generating algorithm to transform the model scan which is used in Fig. 3.1. The normal distributions model the model scan more accurately using fewer distributions. The number of normal distributions is 87 in Fig. 3.1 but 78 in Fig. 3.9. (a) Front view. (b) Top view.

Chapter 4

Evaluation

The performance evaluation of the SV-NDT is separated into an evaluation of the supervoxel-generating algorithm and the registration process. The experiments for the performance evaluation of the supervoxel-generating algorithm under various scan conditions are presented in section 4.1. The performance of the registration process of the SV-NDT is assessed through a comparison with other widely used scan registration algorithms in section 4.2 converges.

4.1 Supervoxel-generating algorithm

The experiments to assess the speed of the supervoxel-generating algorithm and the modeling accuracy of the normal distributions as generated using the supervoxel-generating algorithm were performed with several types of model scans. In these experiments, synthetic scan data were used to control the attributes of the model scans and to simplify the analysis of the results. The synthetic scan data were

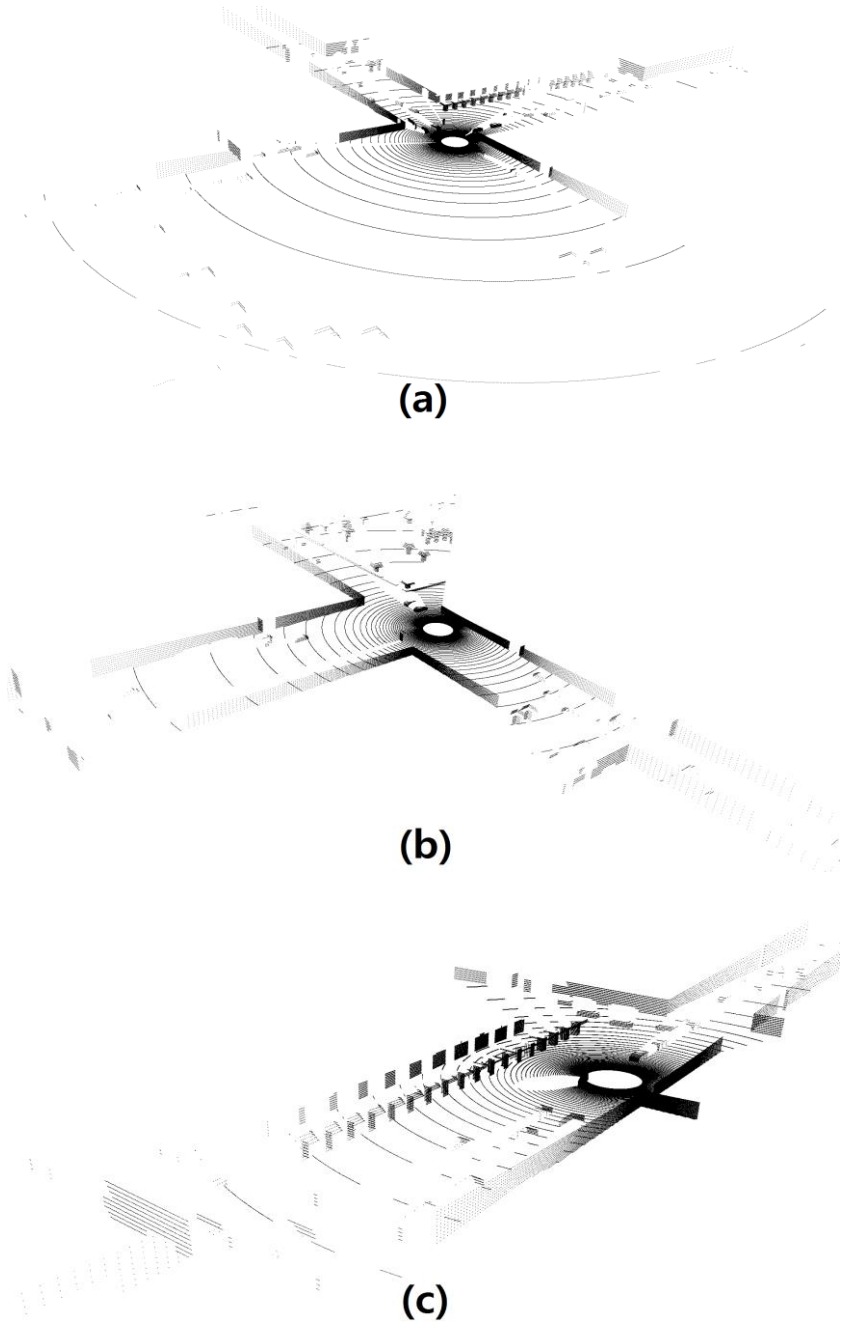

Figure 4.1: Scenes of the synthetic scan data

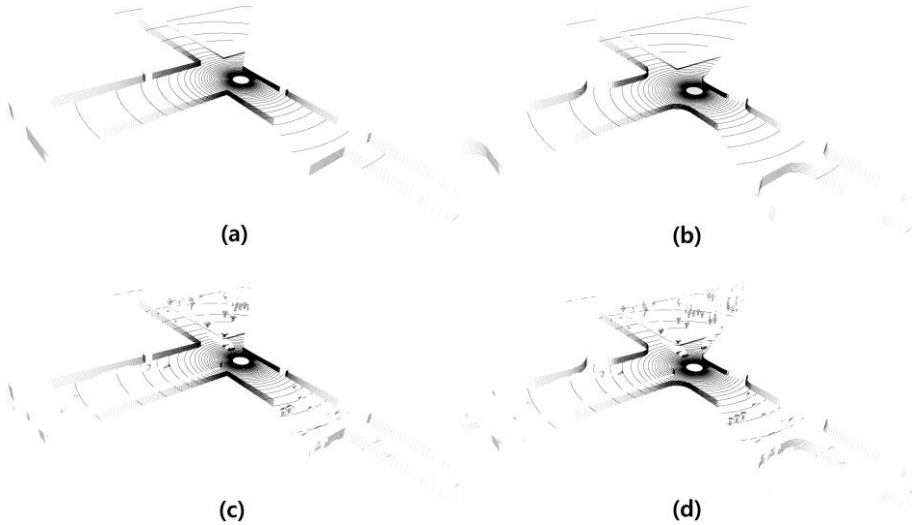

Figure 4.2: Scans created by varying the type of surfaces from the second scene. (a) large planar surface (b) large rounded surfaces (c) large planar surfaces with objects (d) large rounded surfaces with objects

generated by setting up a situation in which a robot in an urban environment collected scans using a Velodyne HDL-64E LiDAR sensor. Therefore, the specifications of the Velodyne HDL-64E were used to determine the angular resolutions and the field of view. Three scenes were constructed for the experiments, as shown in Fig. 4.1. A total of 16 scans were created by varying the three conditions. The first condition involves the existence of relatively small objects, such as trees, cars, and posts. The scans without objects are composed of large surfaces such as the ground and buildings. The second condition is the curvature of the surfaces. The scans in Fig. 4.1 have only planes. The rounded surfaces were generated by bending the surfaces around the edges. Given that there could be four types of surfaces, four scans could be generated per scene by changing these two conditions. The four scans

Table 4.1: Average runtime for the supervoxel generation using each kind of scans (ms)

Noise level (cm) / Type of surfaces	0	1	5	10
Large planar surfaces	142	264	268	352
Large rounded surfaces	160	269	328	387
Large planar surfaces with objects	148	271	297	333
Large rounded surfaces with objects	148	276	322	353

generated from the second scene are illustrated in Fig. 4.1. The final condition is the noise level. The noise is modeled as additive zero-mean Gaussian noise, and its standard deviation is used as the noise level. additive zero-mean Gaussian noise, and its standard deviation is used as the noise level. Four values, i.e., 0, 1, 5, and 10 centimeters, were used as the noise levels. The synthetic scan data contain 251,688 points on average.

4.1.1 Speed

First, an evaluation of the speed of the supervoxel-generating algorithm was performed. The average runtimes of the supervoxel-generating algorithm using each type of scan are given in Table 4.1 when the seed resolution was set to five meters. The runtime does not depend on whether the objects exist, but it increased slightly when the surfaces of the scans were rounded at the edges. However, this result depends heavily on the noise levels. The runtimes at a noise level of ten centimeters are more than doubled relative to those in noise-free cases. Nevertheless, the overall

runtime is quite small, at less than 0.4 seconds, considering the number of points in the scans.

4.1.2 Modeling accuracy

Next, the evaluation of the modeling accuracy of the normal distributions generated using the supervoxel-generating algorithm was conducted by comparing the normal distributions generated using the 3-D regular grid with the resolution equal to the seed resolution of the supervoxel-generating algorithm. The second scene was selected for the evaluation because it has the most objects, and zero and ten centimeters were used as the noise levels.

Initially, noise-free scan data were used; Figure 4.3 shows the normal distributions created by each method according to the type of surface. The columns present the results of the 3-D regular grid and the supervoxel-generating algorithm, and the rows show the results for each type of surface, which are in this case large planar surfaces, large rounded surfaces, large planar surfaces with objects, and large rounded surfaces with objects. For the large planar surfaces, the simplest type of surfaces, many of the normal distributions shown in Fig. 4.3(a) do not model local surface structures accurately because the boundaries of the cell in the 3-D regular grid do not take into account the boundaries of the local surfaces, which have different local geometries. However, the normal distributions generated with the supervoxel-generating algorithm model the model scan accurately (Fig. 4.3(b)). In Fig. 4.3(b), some supervoxels are composed of points on a curve and not a surface. It appears that the distributions from those supervoxels do not model the local

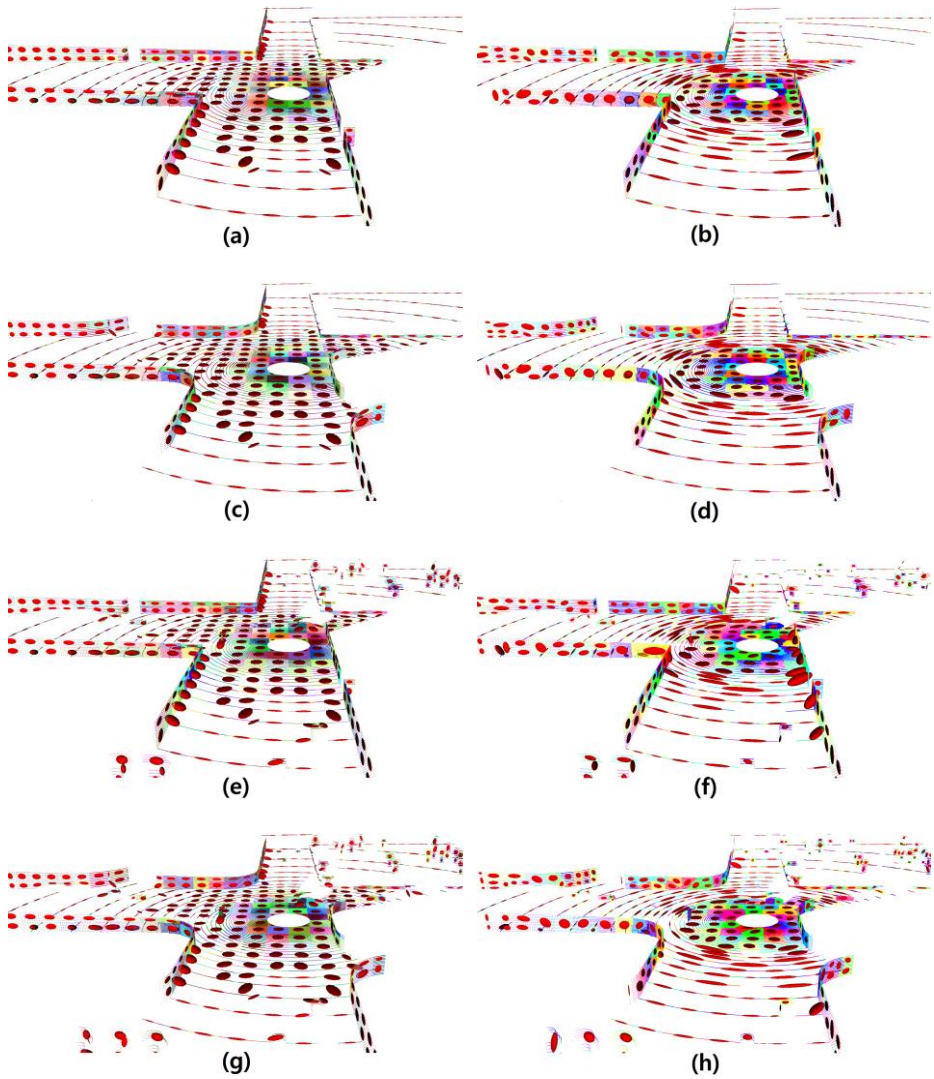

Figure 4.3: Normal distributions created by each method according to the type of surfaces with zero noise level. Each column presents the results of the 3-D regular grid and the supervoxel-generating algorithm, and each row shows the results of each type of surfaces, which are large planar surfaces, large rounded surfaces, large planar surfaces with objects, and large rounded surfaces with objects.

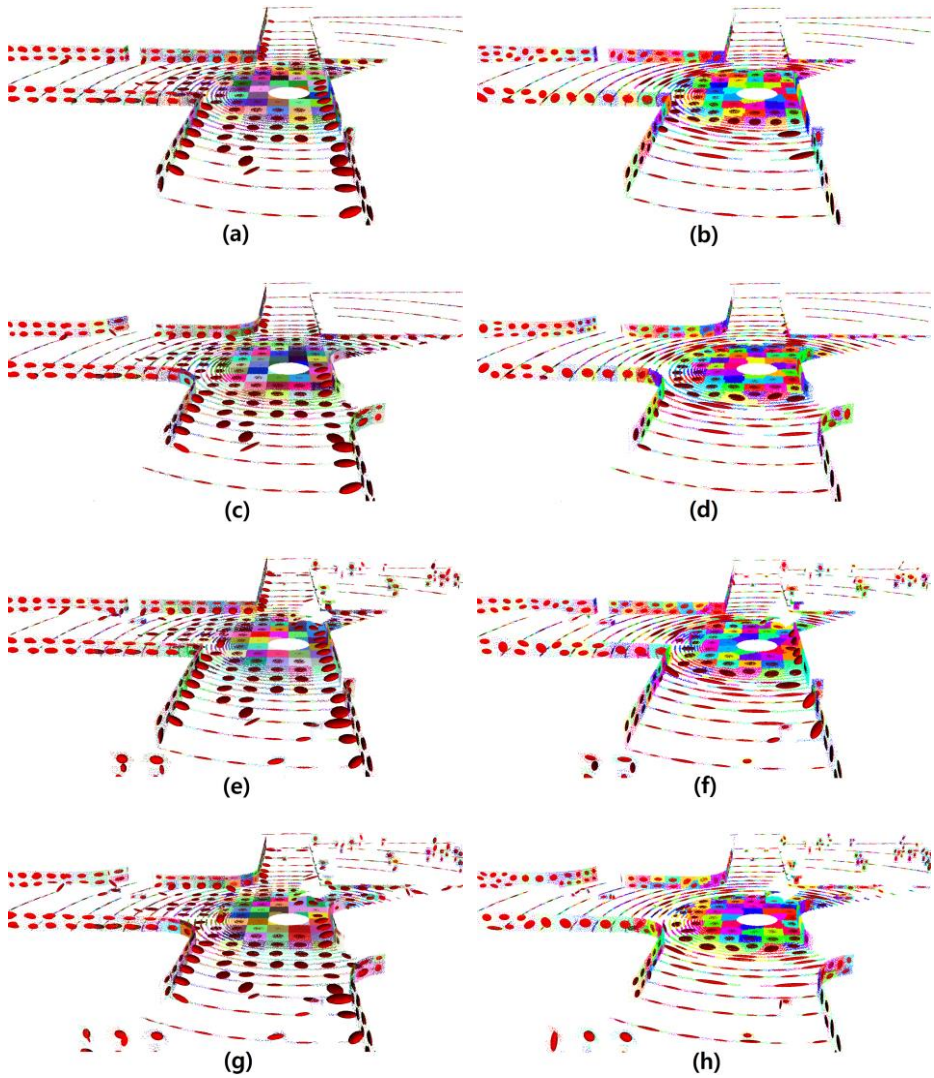

Figure 4.4: Normal distributions created by each method according to the type of surfaces with 10 centimeters noise level. Each column presents the results of the 3-D regular grid and the supervoxel-generating algorithm, and each row shows the results of each type of surfaces, which are large planar surfaces, large rounded surfaces, large planar surfaces with objects, and large rounded surfaces with objects.

surfaces accurately, but this stems from the limited angular resolution of the sensor. When the ray from the sensor to a point is nearly parallel to the surface containing that point, the points obtained around that point are locally formed, not as a surface but as a curve due to the limit of the angular resolution of the sensor, and the curves are far apart from each other. There is no reason for these curves to be considered as constituents of one local surface when no information about the angular resolution of the sensor is given. Thus, although supervoxels with ground points that are far from the sensor may contain a curve, the modeling accuracy is not affected.

To evaluate the influence of the first condition, i.e., the existence of the objects, the first and the second row in Fig. 4.3 are compared with the third and last row in Fig. 4.3, respectively. When the 3-D regular grid is used, normal distributions that could not model local surfaces were created. This mainly results from the fact that both the object and some part of the ground were modeled by one normal distribution. Because a few ground points are far from the sensor, the trees near them are modeled quite well. With the supervoxel-generating algorithm, although a few distributions deteriorated the modeling quality, the added objects were modeled accurately by adding small normal distributions.

To assess the influence of the second condition, i.e., whether the surfaces are planar or round, the first and the third row in Fig. 4.3 are compared with the second and last row in Fig. 4.3, respectively. However, the results of both methods are not affected by the second condition.

To evaluate the effect of the final condition, i.e., the noise level, the same process was repeated with a noise level of ten centimeters. These results are shown in Fig. 4.4. Because some of the points near the cell boundaries pass these boundaries,

the modeling accuracy of the 3-D regular grid decreased sharply (the left column in Fig. 4.4). However, the noise had little impact on the results of the supervoxel-generating algorithm (the right column in Fig. 4.4). The supervoxel-generating algorithm created the boundaries of the supervoxels based on the local surface structures; thus, it is robust to a high noise level.

To quantitatively evaluate that how well the normal distributions generated from the model scan model the surroundings, tests that classify samples according to whether they are from the occupied space or not were performed. These classifications can be carried out by the likelihood function. In the NDT algorithm, the score function in Eq. (3.11) is used as log-likelihood function approximately. Thus, a sample is classified as from the occupied space if the score function is larger than a certain threshold and from the unoccupied space otherwise. The accuracy, the F-measure (also F1-score), the mutual information, and the area under the receiver operating characteristic (ROC) [77] were used as the performance indices. These values were estimated by the k-fold cross-validation, and 10 was chosen as the value of k value in these experiments. Because the model scan is composed of positive examples which are the samples from the occupied space, negative examples that are the samples from the unoccupied space should be created. In order to test effectively, each negative example was generated by not choosing a totally random point but moving each point in the model scan to some extent along the line from the point to the origin of the sensor [78]. The extent was randomly selected value from a uniform distribution which ranges from 1.5 meters to 5 meters. After that, the positive examples were divided into ten point subsets, $P_1, \cdots, P_{10}$, and the negative

		True class	
		Positive	Negative
Test outcome	Positive	**True positive**	**False positive**
	Negative	**False negative**	**True negative**

Figure 4.5: Confusion matrix

examples were also split into ten point subsets, $N_1, \cdots, N_{10}$, in order that each subset of the positive examples P_i corresponds to the N_i. Next, normal distributions were generated by using the training set that was the union of the subsets of the positive examples except the P_i, then the performance indices were estimated using the test set that was the union of the P_i and the N_i. Because the NDT algorithm employs only positive examples, i.e., samples from the occupied space, to generate normal distributions, only positive examples were used in the training phases. This process was repeated ten times with changing i from one to ten. Finally, each final estimated performance index was determined as the mean value of the ten estimated values.

Test outcomes of test set can be classified into 4 cases. An outcome of a positive example is called true positive if the example is classified as positive and false negative otherwise. Also, an outcome of a negative example is called true negative if the example is classified as negative and false positive otherwise. Figure 4.5

shows a confusion matrix summarizing those cases. As basic performance indices of binary classifiers, the accuracy and the F-measure are defined as

$$accuracy = \frac{True\ positive + True\ negative}{Total\ population} \tag{4.1}$$

$$recall = \frac{True\ positive}{True\ positive + False\ negative} \tag{4.2}$$

$$precision = \frac{True\ positive}{True\ positive + False\ postive} \tag{4.3}$$

$$F\text{-}measure = \frac{recall \cdot precision}{recall + precision} \tag{4.4}$$

where $Total\ population$ is the total number of the test set. The accuracy of a binary classifier which predict examples by using a fair coin, i.e., there is no discriminatory ability, is 0.5, thus the minimum value of the accuracy is 0.5. If the accuracy of a binary classifier is less than 0.5, the accuracy can be larger than 0.5 by reversing the test outcome. When the number of the positive examples and the negative examples are a and b respectively, the random classifier, the binary classifier having no ability to predict, has $a/2$ true positives, $a/2$ false negatives, $b/2$ false positives, and $b/2$ true negatives. Thus, the minimum value of the F-measure, the F-measure of the random classifier, is $2a/(3a+b)$, and this value is 0.5 in these experiments because a equals b.

When two discrete random variables, X and Y, are given, the mutual information is defined as

$$I(X;Y) = \sum_{x,y} p_{XY}(x,y) \log \frac{p_{XY}(x,y)}{p_X(x)p_Y(y)} \tag{4.5}$$

where p_{XY} is a joint probability mass function of X and Y and p_X and p_Y are marginal probability mass function of X and Y, respectively. In these experiments, the X means the true class, and the Y means the test outcome. The value of 1 represents positive, and the value of 0 represents negative. Then, the joint distribution can be calculated as

$$p_{XY}(x,y) = \begin{cases} \dfrac{True\ negative}{Total\ population}, & x = 0\ and\ y = 0 \\ \dfrac{False\ negative}{Total\ population}, & x = 1\ and\ y = 0 \\ \dfrac{False\ positive}{Total\ population}, & x = 0\ and\ y = 1 \\ \dfrac{True\ positive}{Total\ population}, & x = 1\ and\ y = 1 \end{cases} \tag{4.6}$$

, and each marginal distribution can be calculated by summing the joint distribution over the other random variable. The minimum value of the mutual information can also be calculated by the random classifier, and it is 0.

The area under the ROC (AUC), like its name, is defined by the area under the ROC curve, and it represent the overall performance of a binary classifier about whole value of a threshold, unlike previous three indices which represent the per-

formance of a binary classifier with a certain threshold. The ROC graph is obtained by the true positive rate (TPR) and the false positive rate (FPR), and these two values are defined as

$$TPR = \frac{True\ positive}{True\ positive + False\ negative} \tag{4.7}$$

$$FPR = \frac{False\ positive}{False\ positive + True\ negative} \tag{4.8}$$

For each threshold, a (FPR, TPR) ordered pair can be obtained, and this is able to be plotted by a point on the FPR-TPR plane. If the threshold is changed from 0 to 1, then a curve on the FPR-TPR plane is created. This curve is called a ROC curve, and it starts from $(0,0)$ and ends at $(1,1)$. The ROC curve of the random classifier is a line from $(0,0)$ to $(1,1)$, and the AUC is 0.5. Thus, the minimum value of the AUC is 0.5, and if the AUC is less than 0.5, the AUC can be larger than 0.5 by reversing the test outcome.

Average values of the estimated performance indices of the supervoxel-generating algorithm and the 3-D regular grid on the model scan of each type of surfaces and each noise level are shown in Fig. 4.6. Except for the AUC which does not depend on the threshold, the value of each performance index is the maximum value of estimated values which can be obtained using thresholds from 0 to 1, and the minimum value of each plot on the y-axis is set to the minimum value of each performance index. In Fig. 4.6, all indices of the supervoxel-generating algorithm are greater than those of the 3-D regular grid, and when the noise level is increased,

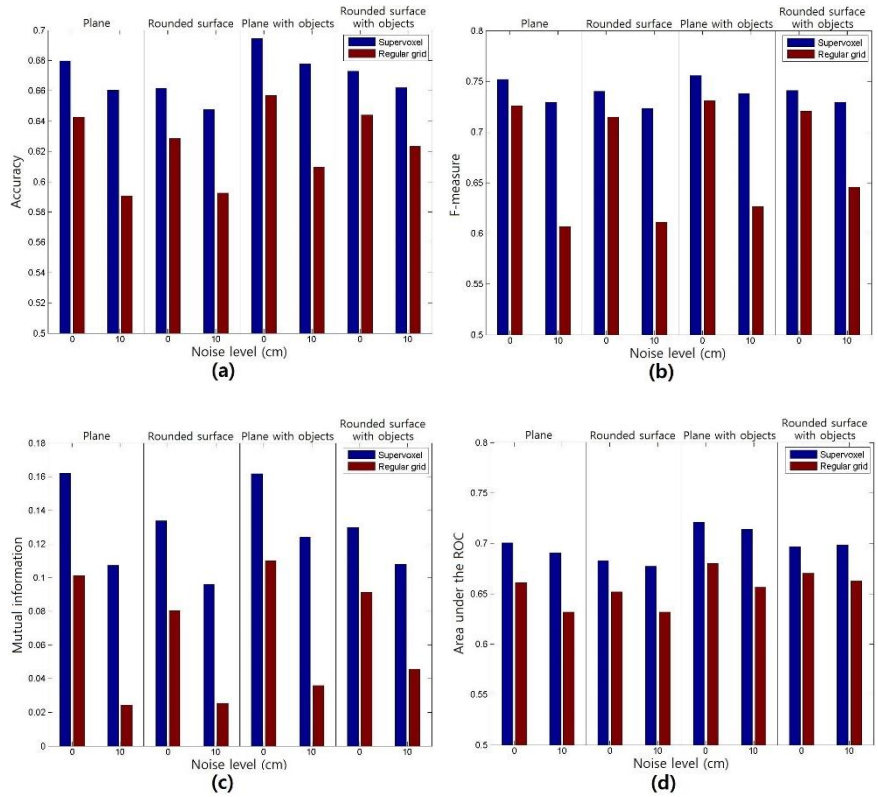

Figure 4.6: The estimated performance indices for the supervoxel-generating algorithm and the 3-D regular grid (a) accuracy (b) F-measure (c) mutual information (d) area under the ROC

the indices of the two methods are decreased on the whole. However, the decrements of the supervoxel-generating algorithm are far less than those of the 3-D regular grid. These results quantitatively demonstrate the fact, that the normal distributions generated using the supervoxel-generating algorithm model the model more accurately than those generated using the 3-D regular grid and that the supervoxel-generating algorithm is robust to the noise because it can adaptively create the boundaries of the supervoxels, which can be verified in Fig. 4.3 and 4.4.

4.2 Registration

The experiments to assess the performance of the registration process by the SV-NDT used a point cloud dataset in both synthetic and real-world datasets. Not only the 3-D NDT but also the ICP and the G-ICP, which are both widely used 3-D scan registration algorithms, were selected for the comparison of the performance. These three algorithms were implemented with a point cloud library (PCL). In addition to those algorithms, the SV-NDT which uses the Euclidean distance when creating correspondences between the points in the data scan and the normal distributions was also evaluated on the synthetic scan data in order to assess the effect of the proposed distance function.

The performance indices are the robustness, accuracy, and speed. A registration is judged as a success if the final transformation error is less than certain success criteria, and the robustness of each registration algorithm was measured by the success rate. The speed and the accuracy of each algorithm were measured using the runtime, the final translation error, and the final rotation error of the successful registration cases. If a registration result of each algorithm is best at the global minimum of the cost function of each algorithm, each performance index shows a distinct feature. The robustness represents how large the basin of attraction of the global minimum is, and the accuracy represents how close the global minimum to the ground truth. In addition, the speed represents how fast the algorithm reach the global minimum when it start in the basin of attraction of the global minimum.

4.2.1 Synthetic dataset

First, the synthetic scan data is used for the experiments. Each point in the synthetic scan data used in the previous section was paired by creating new scans whose origins are five meters away from the origins of the existing scans. Therefore, a total of 48 scan pairs were used, and the raw data were used for the experiments. The seed resolution in the SV-NDT and the voxel resolution in the 3-D NDT were set to 5 meters. Likewise, the maximum distance threshold between two corresponding points is a parameter in the ICP and the G-ICP, and it was set to 10 meters for each algorithm. These parameters were determined from among 2.5, 5, and 10 meters by experiments to determine which one provides the best performance with those scan data. The registration experiments were performed using a variety of initial transformation errors. The tested initial transformation errors were composed of the translation errors from -3 to 3 meters at intervals of 1.5 meters along the x and y axes and the rotation errors from -30 to 30 degrees at intervals of 15 degrees about the z axis. Thus, a total of 125 initial transformation errors were tested for each scan pair. To measure the performance indices, the success criteria were set to a translation error of 0.3 meters and a rotation error of 0.05 radians. The success rate of each algorithm according to the conditions, the type of surfaces and the noise level, was calculated by the success rate out of a total 375 registration results generated by applying 125 initial transformation errors to three scenes with the same conditions. For each type of surface, the results for each algorithm according to the noise level are illustrated in Figs. 4.7 to 4.10. The results in Figs 4.7 to 4.10 were obtained using large planar surfaces, large rounded surfaces, large planar

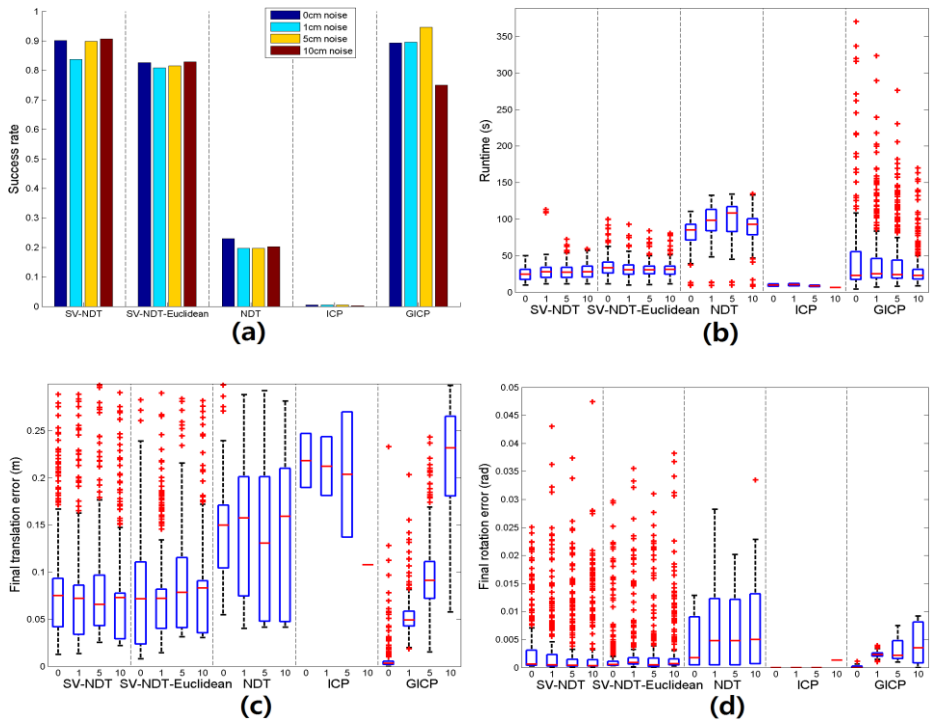

Figure 4.7: Registration results of each registration algorithm according to the noise levels using the type of large planar surfaces. (a) success rate (b) runtime (c) final translation error (d) final rotation error

surfaces with objects, and large rounded surfaces with objects. To compare each algorithm easily, every figure is divided into five sections (from left: SV-NDT, SV-NDT with the Euclidean distance (SV-NDT-E), the 3-D NDT, ICP, and G-ICP). Each section shows the results of each algorithm according to the noise level (from left: 0, 1, 5, and 10 centimeters).

The first plot of each figure shows the success rate of each algorithm using bar graphs. The second to fourth plots of each figure respectively present the runtime, final translation and rotation error of each algorithm using boxplots. In each boxplot,

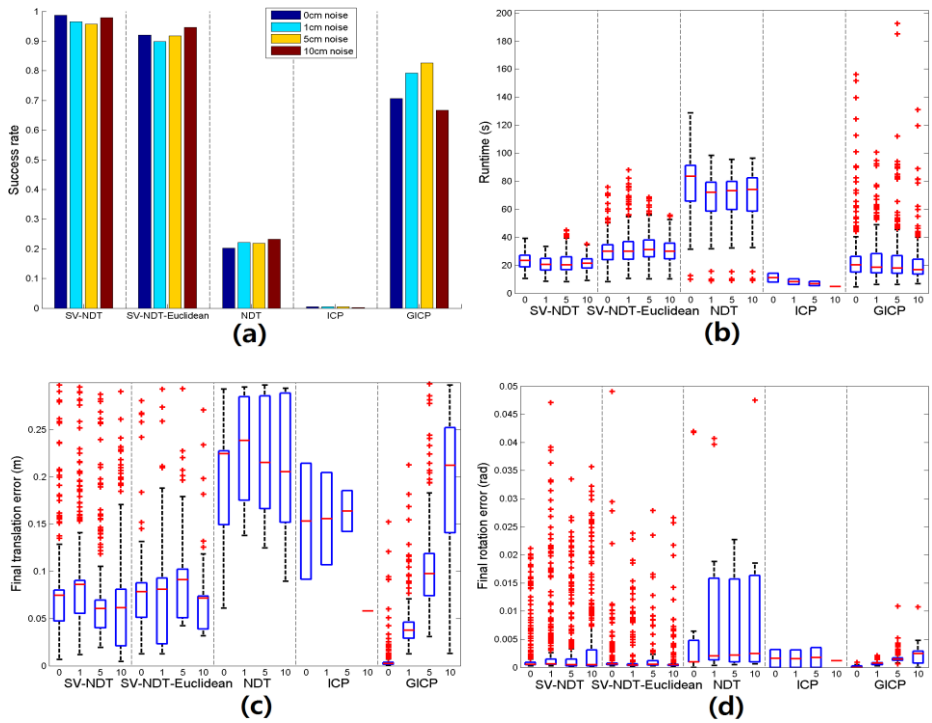

Figure 4.8: Registration results of each registration algorithm according to the noise levels using the type of large rounded surfaces. (a) success rate (b) runtime (c) final translation error (d) final rotation error

the red line shows the median, and the blue box represents the range from the 25th percentile to the 75th percentile. The length of the whiskers is 1.5 times the length between the 25th percentile and the 75th percentile, and the results outside the whiskers are considered as outliers.

First, the robustness of each algorithm was analyzed. As shown in the figures, the most robust algorithm is the SV-NDT, and its success rates are greater than 0.9 in all but two cases of conditions. Moreover, its lowest success rate is 0.8373. The G-ICP is more robust than the SV-NDT-E except for the case of the large rounded

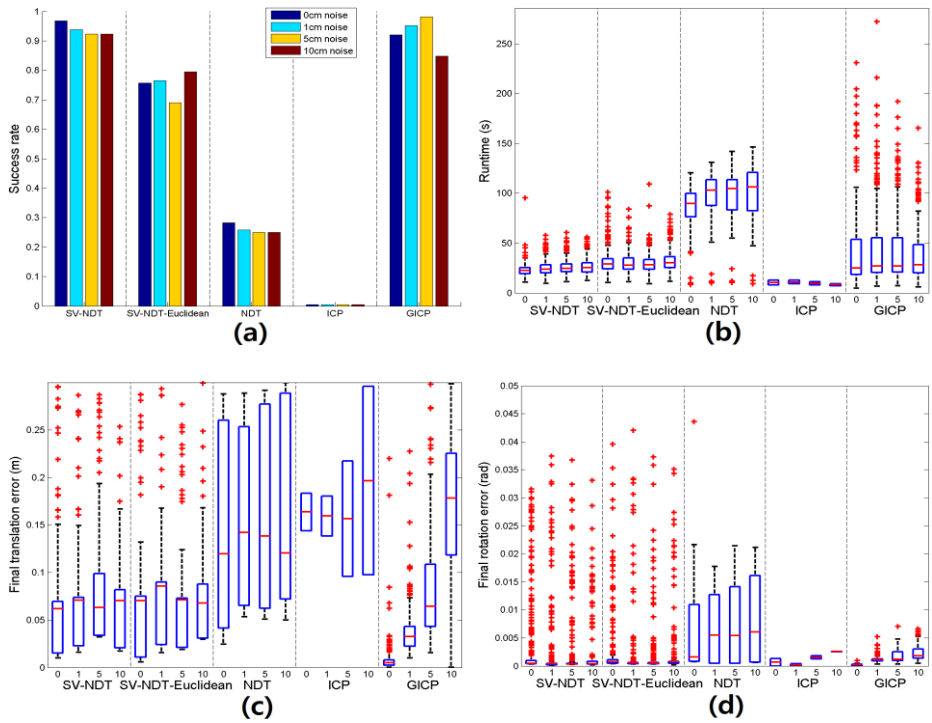

Figure 4.9: Registration results of each registration algorithm according to the noise levels using the type of large planar surfaces with objects. (a) success rate (b) runtime (c) final translation error (d) final rotation error

surfaces, and it is followed by the 3-D NDT and the ICP. The 3-D NDT is more robust than the ICP, but it cannot overcome the large initial transformation errors. For all except for the SV-NDT-E, the success rates increased when the scan data with the objects were used, as the objects provide additional information with which to estimate the relative transformation between the two scans. The success rates of the SV-NDT and the SV-NDT-E when rounded surfaces are used are larger than those when planar surfaces are used, but the G-ICP has the opposite effect. The overall robustness levels of the SV-NDT, the SV-NDT-E, and the 3-D NDT depend

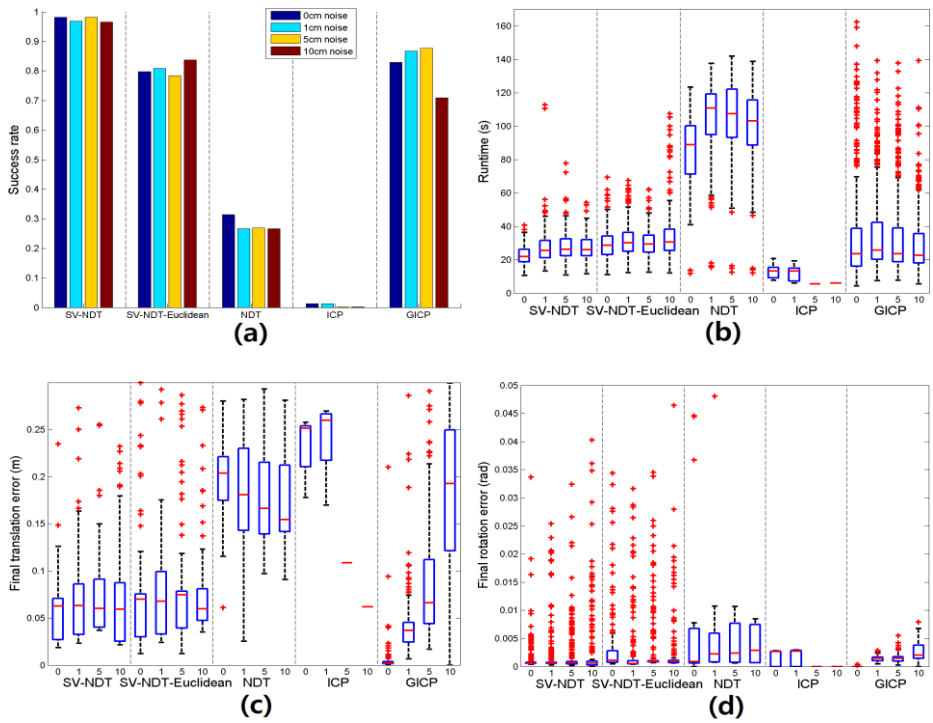

Figure 4.10: Registration results of each registration algorithm according to the noise levels using the type of large rounded surfaces with objects. (a) success rate (b) runtime (c) final translation error (d) final rotation error

little on the noise level, whereas the robustness of the G-ICP relies heavily on the noise level. When the noise level is ten centimeters, the success rate of the G-ICP falls rapidly. This outcome results from the number of points used to estimate the covariance matrices. The SV-NDT and the 3-D NDT use points in each part of the partitioned model scan; thus, many covariance matrices are reliably estimated due to the use of a sufficient number of points. However, the G-ICP uses only a prede-fined number of points around each point to estimate the covariance.

Next, the speed of each algorithm is evaluated. The ICP is the fastest, followed by the SV-NDT, the SV-NDT-E, the G-ICP, and the 3-D NDT. The ICP achieves success only when the initial transformation errors are quite small. Thus, the runtime is very small, about 8.86 seconds, compared to the other algorithms. The median runtimes of the SV-NDT and the G-ICP, about 24.26 and 23.14 seconds respectively, are similar to each other, but their corresponding 75th percentiles, about 29.43 and 40.58 seconds, differ considerably. The median runtime of the SV-NDT-E is about 30.07 seconds, which is larger than that of the G-ICP, but the 75th percentile of the SV-NDT-E, about 36.03 seconds, is smaller than that of the G-ICP.

Finally, the accuracy levels were compared. The results of the SV-NDT, the SV-NDT-E, and the 3-D NDT do not show any tendency with regard to different noise levels. In addition, they show similar accuracy levels in many cases. However, the G-ICP shows a clear tendency. In the noise-free case, the G-ICP is the most accurate, but the accuracy decreases sharply with an increase in the noise level. The reason is identical to that of the success rate: the number of points used to estimate each covariance matrix. The overall accuracy of the 3-D NDT is much lower than that of the SV-NDT because the normal distributions generated when using the 3-D regular grid cannot model the model scan accurately, as shown in section 4.1.

The SV-NDT-E is significantly superior to the 3-D NDT in all aspects but lacks the performance in terms of the robustness and speed compared to the SV-NDT. This shows that the registration performance is heavily affected by the modeling accuracy of the normal distributions as transformed from the model scan. Moreover, the proposed distance function has a positive effect on the size of the basin of attraction and the speed of convergence.

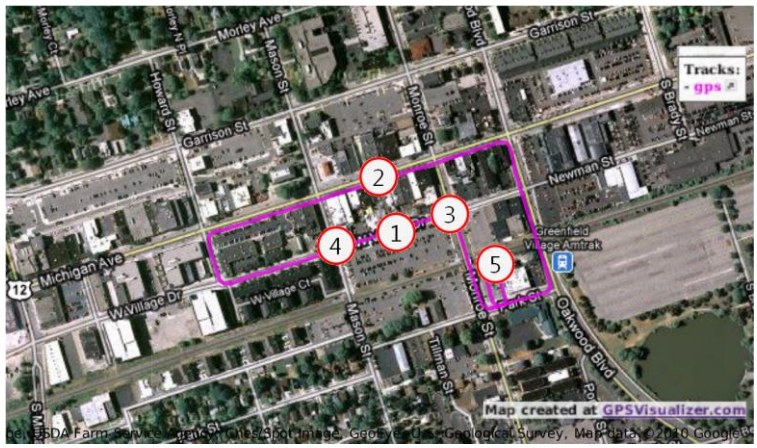

Figure 4.11: Positions of the test scan pairs with full path of the dataset.

Table 4.2: Detailed surroundings of the test scan pairs

Scan pair number	Detailed surroundings
1	A straight road with low car density
2	A straight road with high car density
3	Turning left at an intersection
4	Going straight at an intersection
5	A parking lot

4.1.2 Real-world dataset

The real-world dataset is the Ford Campus Vision and Lidar Data Set [79] gathered using a Velodyne HDL-64E LiDAR sensor. Five distinguishing scan pairs with time step differences of 10 were selected for the test. Figure 4.11 and Table 4.2 show the positions and the detailed surroundings of the test scan pairs, respectively. The raw data, which contain 78,341 points on average, were used in an intact form. The seed resolution in the SV-NDT and the voxel resolution in the 3-D NDT were set to 10 meters and 5 meters, respectively, and the maximum correspondence

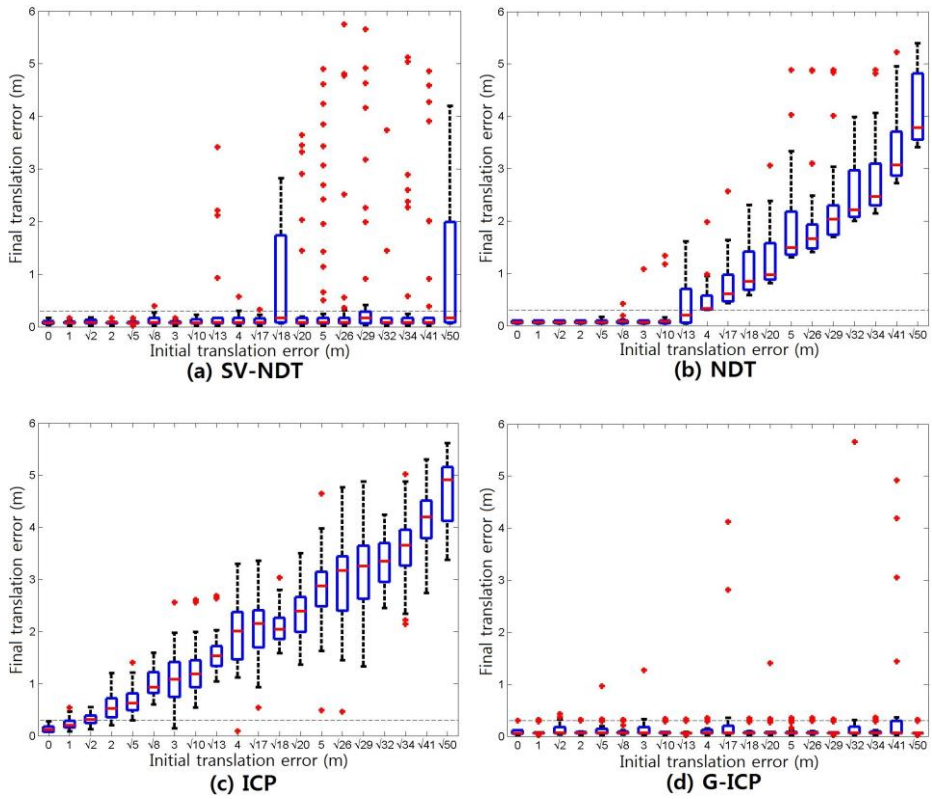

Figure 4.12: The boxplots of the final translation errors of each registration algorithm according to the initial translation errors when the initial transformation errors consist of only the initial translation errors.

distance was set to 10 meters for each algorithm. These parameters were determined from among 3, 5, 10, and 20 meters by experiments to determine which one provides the best performance with the dataset.

Initially, cases in which the initial transformation errors consist of either translation or rotation errors were evaluated so as to evaluate the robustness of each algorithm to the initial translation and initial rotation errors separately. First, experiments to test the robustness to initial translation errors were performed for

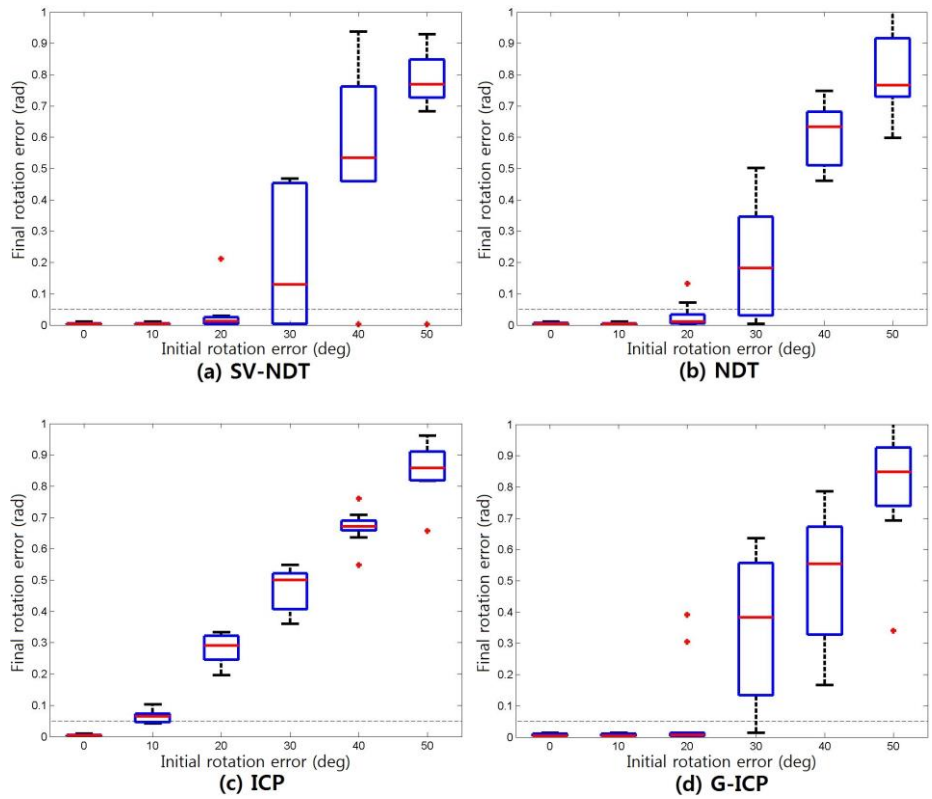

Figure 4.13: The boxplots of the final rotation errors of each registration algorithm according to the initial rotation errors when the initial transformation errors consist of only the initial rotation errors.

each algorithm. The tested initial translation errors ranged from -5 to 5 meters at intervals of one meter along the x and y axes. Figure 4.12 shows boxplots of the results of these experiments. The horizontal axis represents the magnitude of each initial translation error vector. Because the initial translation errors are two-dimensional vectors, the magnitude difference is not uniform. The dashed line represents the success criterion, which was set to 0.3 meters. The results show that the SV-NDT and the G-ICP are most robust to the initial translation error. In the ICP

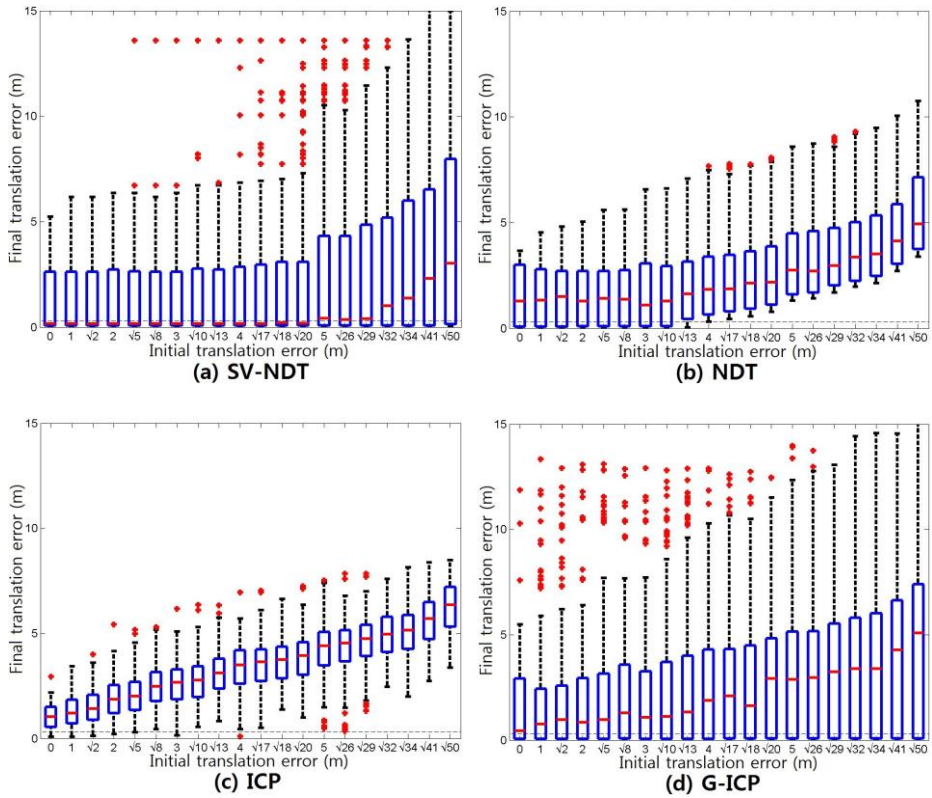

Figure 4.14: The boxplots of the final translation errors of each registration algorithm according to the initial translation errors when the initial transformation errors consist of both initial translation and initial rotation errors.

case, the final translation errors increase with greater initial translation errors. The 3-D NDT fails in registration when the initial translation errors are larger than four meters. However, all of the medians of the final translation errors of the SV-NDT and the G-ICP are under the dashed line. Nevertheless, the SV-NDT has many outliers when the initial translation errors are greater than or equal to the square root of 18. About half of these outliers occur from the second scan pair. The major characteristic of the environment of the second scan pair is that there are numerous

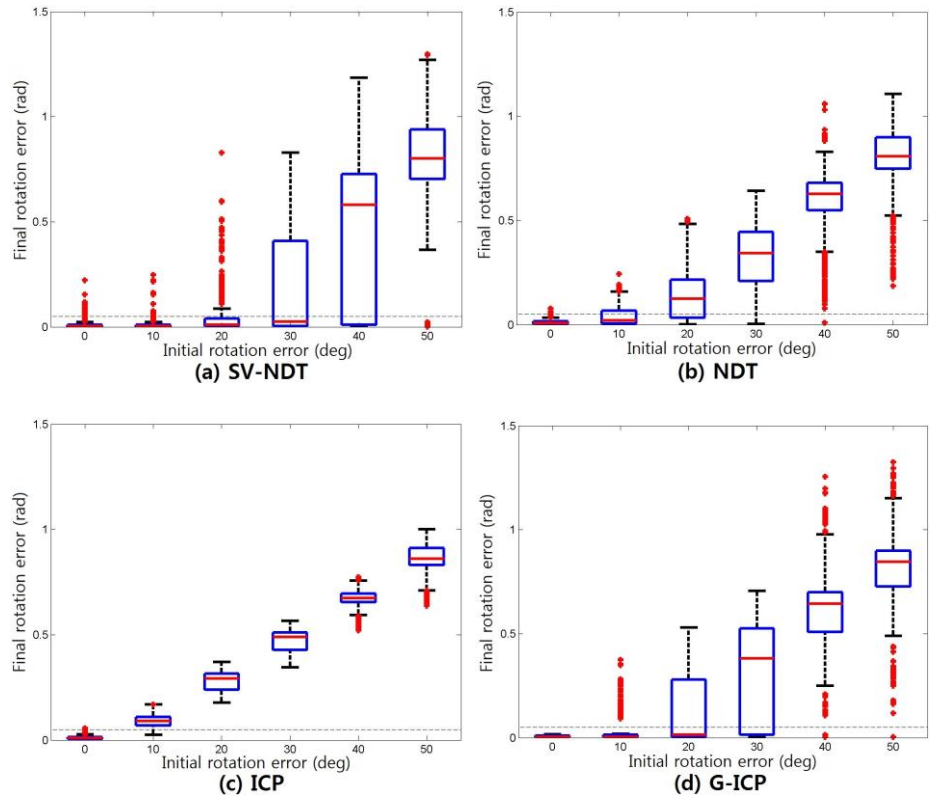

Figure 4.15: The boxplots of the final rotation errors of each registration algorithm according to the initial rotation errors when the initial transformation errors consist of both initial translation and initial rotation errors.

moving cars. Considering that most of the scanned points from cars are from the sides of the cars, the estimated local surfaces from the cars are perpendicular to the ground. The G-ICP algorithm makes correspondences using the Euclidean distance and takes the normal directions of the local surfaces into consideration by adjusting for the influence of the correspondences via a cost function according to the similarity of the normal directions. The points from moving cars in the data scan correspond to the points from the ground; thus, their influences on the cost function are

reduced sharply. Consequently, the G-ICP has few outliers in spite of the many moving cars which existed in this case. However, the SV-NDT algorithm takes the normal directions of the local surfaces into account when making correspondences. Thus, the points from moving cars in the data scan correspond to the distributions from the same cars which are, however, not in the same position or from other surfaces which are not parallel to the ground. As a result, a few final translation errors are quite large and are therefore considered as outliers, but most of the final translation errors, at least three quarters of them except for the initial translation errors, are the square roots of 18 and 50, i.e., small enough to be regarded as successful cases. Next, experiments to test the robustness to the initial rotation error were performed for each algorithm. The tested initial rotation errors ranged from -50 to 50 degrees at intervals of 10 degrees about the z axis. Boxplots of the results of these experiments are shown in Fig. 4.13. The dashed line, the success criterion, was set to 0.05 radians. The SV-NDT, the 3-D NDT, and the G-ICP carried out the scan registration successfully with an initial rotation error of up to 20 degrees, but the ICP failed with small initial rotation errors. Because the 25^{th} percentiles of the final rotation errors of the SV-NDT and the 3-D NDT are under the dashed line in the case of 30 degrees, the SV-NDT and the 3-D NDT are the most robust algorithms with regard to the initial rotation errors. Given that the initial transformation error consists of both translation and rotation error concurrently in general cases of scan registration, an assessment of those cases is also required. Thus, experiments whose initial transformation errors were composed of initial translation and initial rotation errors were performed for each algorithm. The tested initial formation errors ranged from -5 to 5 meters of the initial translation error at intervals of one meter along the x and y

axes and from -50 to 50 degrees of the initial rotation error at intervals of 10 degrees about z axis. In other words, each algorithm runs a total of 1331 registration per scan pair. Boxplots of the translation and the rotation part of the results are shown in Figs. 4.14 and 4.15, respectively. The SV-NDT outperforms the other scan registration algorithms. Because the initial translation and rotation errors were combined, the overall magnitude of the final translation or rotation errors and the numbers of outliers are increased for all algorithms. This stems from the differences in the sizes of the basins of attraction of each registration algorithm. Because the sizes of the basins of attraction of the 3-D NDT and the ICP are small, the estimated transformations are close to the initial transformation. Thus, the final transformation errors increase with the initial transformation errors, but they are narrowly distributed. As a result, the numbers of outliers of the 3-D NDT and the ICP are small. However, in that the SV-NDT and the G-ICP have large basins of attraction of the correct solution, these algorithms can overcome large initial transformation errors, but in the cases with converging local minima, the distributions of the final transformation errors are wide. This is why the SV-NDT and the G-ICP have many outliers, as shown in Figs. 4.14 and 4.15. In spite of that, the medians of the final translation errors of the SV-NDT are under the dashed line when the initial translation errors are less than five meters. Although the 25^{th} percentiles of the final translation errors of the G-ICP are as low as those of the SV-NDT, the medians increase with the initial translation error. The success rate of the 3-D NDT is greater than 25% when the initial translation error is as high as four meters, but the 3-D NDT fails in registration when the initial translation errors are larger than four meters. Because the ICP is not robust to the initial rotation errors, the robustness decreases

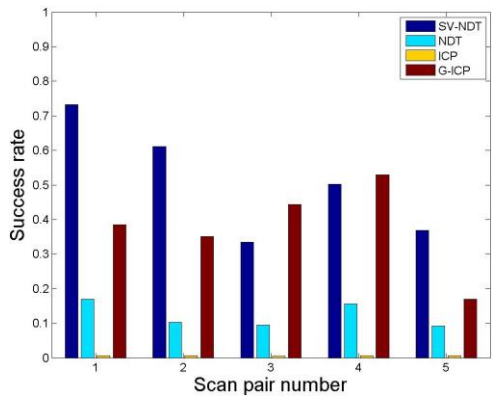

Figure 4.16: Success rate of each algorithm for each scan pair.

Table 4.3: Performances of each algorithm

	SV-NDT	**NDT**	**ICP**	**G-ICP**
Success rate average	0.5098	0.1231	0.0057	0.3757
Success rate partial average	0.7342	0.3391	0.0157	0.5921
Translation error (mm)	73.523	61.471	190.58	61.31
Rotation error (mrad)	3.017	4.183	3.502	3.50
Runtime (s)	4.025	23.835	1.862	7.88

drastically regardless of the initial translation error. According to Fig. 4.15, the SV-NDT succeeded in most of the registrations up to an initial rotation error of 20 degrees, and the 3-D NDT and G-ICP achieved success up to an initial rotation error of 10 degrees. In addition, the success rate of the SV-NDT is greater than 50% when the initial rotation error is 30 degrees.

The success rate of each algorithm is summarized in Fig. 4.16 and in the first row of 4.3. The SV-NDT far surpasses the 3-D NDT and is superior to the other algorithms on average although the G-ICP is better than the SV-NDT on scan pairs 3 and 4, which were collected at intersections. The overall success rates are

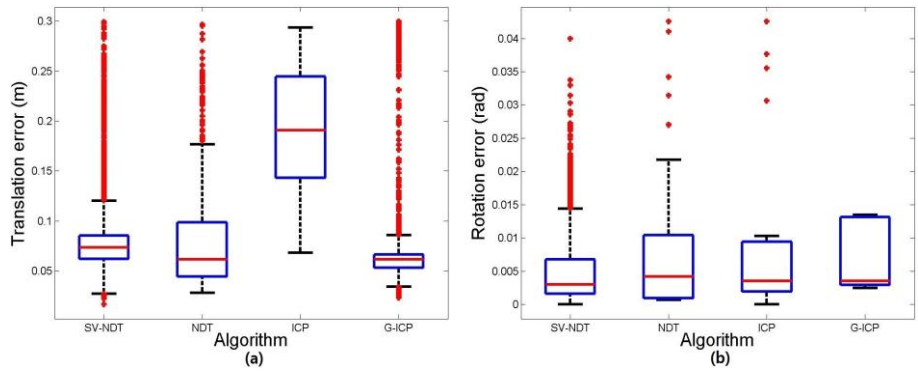

Figure 4.17: Accuracy of each registration algorithm (a) translation part (b) rotation part

somewhat low, as the initial translation errors are two-dimensional. When magnitude of the initial translation error is larger, there exist more cases of the initial translation error with the same magnitude. When the initial transformation error is larger than 5 meters for translations and larger than 30 degrees for rotations, no algorithm works well. There are 848 of these cases, representing 63.7% out of the 1331 cases in total. The second row in 4.3 shows the partial averages of the success rates, apart from those cases.

For a comparison of the accuracy levels, the final transformation errors of the successful registration cases were evaluated. Boxplots of the translation part and the rotation part of these results are shown in Fig. 4.17, and medians in the boxplots are summarized in the third and fourth row in Table 4.3. In the translation part, the 3-D NDT and the G-ICP show the best results on the basis of the median values, followed by the SV-NDT and the ICP. The difference between the median values of the SV-NDT and the NDT is approximately one centimeter, which is tiny with

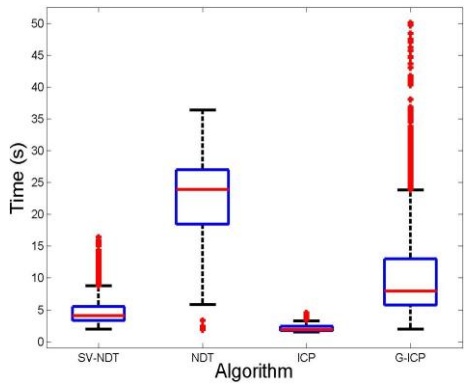

Figure 4.18: Runtime of each algorithm

Table 4.4: Runtimes of the supervoxel-generating algorithm for each scan pair

Scan pair number	1	2	3	4	5
Runtime (ms)	62	65	72	74	69

respect to the scale of the scans, about 120 meters, showing that the translation accuracy levels of the SV-NDT, the 3-D NDT, and the G-ICP are nearly identical. In the rotation part, the SV-NDT is slightly superior to the other algorithms, followed by the G-ICP, the ICP, and the 3-D NDT, but they are also about the same.

Finally, the speed of each registration algorithm were assessed. Figure 4.18 shows boxplots of the runtimes of the successful registration cases. The ICP is the fastest, followed by the SV-NDT, the G-ICP, and the 3-D NDT. The ICP succeeded in registrations with only minor initial transformation errors; thus, its runtime value is much lower than that of the others. The median values of the runtimes of each algorithm are shown in the last row of Table 4.3. The SV-NDT is about 1.96 times faster than the G-ICP and is about 5.92 times faster than the 3-D NDT. Additionally, the runtimes of the supervoxel-generating algorithm for each scan pair are presented

in Table 4.8. These runtimes are about 68.4 milliseconds on average, and they are only 1.70 percent of the average of the overall runtimes of the SV-NDT registration algorithm.

Chapter 5

Conclusion

In this thesis, a novel algorithm based on the 3-D NDT registration algorithm called the SV-NDT is proposed. It reduces the information loss of local surface structures with a supervoxel-generating algorithm when transforming the model scan into normal distributions, and it utilizes the local surface structures of the data scan through the use a newly proposed distance function which selects the corresponding distribution of each point in the data scan rather than the Euclidean distance. The results of an evaluation of the supervoxel-generating algorithm on synthetic scan data show that the algorithm greatly increases the modeling accuracy of the normal distributions. Furthermore, experiments which ran a performance evaluation of the SV-NDT on synthetic and real-world datasets demonstrated that the robustness and speed of the SV-NDT significantly exceed those of previous algorithms while maintaining comparable levels of accuracy.

In future work, performance evaluations of the SV-NDT on datasets which are obtained by various sensors or in different environments can be carried out. In addition, to improve the performance of the proposed algorithm more, a method that

generates supervoxels more consistently can be investigated to exploit supervoxels for loop detection and initial pose estimation purposes. Finally, the SV-NDT can be applied to 3-D SLAM from 3-D scan registration to the graph optimization.

Reference

[1] P. Biber and W. Straber, "The Normal Distributions Transform: A New Approach to Laser Scan Matching," in *2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, USA, 2003.

[2] 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*, vol. 32, no. 14, pp. 1627-1644, 2013.

[3] E. Einhorn and H.-M. Gross, "Generic 2D/3D SLAM with NDT Maps for Lifelong Application," in *2013 European Conference on Mobile Robots*, Barcelona, Spain, 2013.

[4] M. Magnusson, H. Andreasson, A. Nuchter and A. J. Lilienthal, "Automatic Appearance-based Loop Detection from Three-dimensional Laser Data Using the Normal Distributions Transform," *Journal of Field Robotics*, vol. 26, no. 11-12, pp. 892-914, 2009.

[5] T. Stoyanov, M. Magnusson, H. Andreasson and A. J. Lilienthal, "Path Planning in 3D Environments Using the Normal Distributions Transform," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei, Taiwan, 2010.

[6] J. W. Kim and B. H. Lee, "Robust and Fast 3-D Scan Registration Using Normal

Distributions Transform with Supervoxel Segmentation," in *Robotica*, Available on CJO doi:10.1017/S0263574714002483, 2014.

[7] S. Thrun, W. Burgard and D. Fox, Probabilistic Robotics, Cambridge, Massachusetts, USA: The MIT Press, 2005.

[8] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME - Journal of Basic Engineering,* vol. 82, no. Series D, pp. 35-45, 1960.

[9] R. C. Smith and P. Cheeseman, "On the Representation and Estimation of Spatial Uncertainty," *The International Journal of Robotics and Research,* vol. 5, no. 4, pp. 56-68, 1986.

[10] R. Smith, M. Self and P. Cheeseman, "Estimating Uncertain Spatial Relationships in Robotics," in *Autonomous Robot Vehicles*, New York, Springer, 1990, pp. 167-193.

[11] J. J. Leonard, H. F. Durrant-Whyte and I. J. Cox, "Dynamic Map Building for an Autonomous Mobile Robot," *The International Journal of Robotics Research,* vol. 11, no. 4, pp. 268-298, 1992.

[12] S. Thrun, Y. Liu, D. Koller, A. Y. Ng, Z. Ghahramani and H. Durrant-Whyte, "Simultaneous Localization and Mapping with Sparse Extended Information Filters," *The International Journal of Robotics Research,* vol. 23, no. 7-8, pp. 693-716, 2004.

[13] J. Neira and J. D. Tardos, "Data Association in Stochastic Mapping Using the Joint Compatibility Test," *IEEE Transactions on Robotics and Automation,* vol. 17, no. 6, pp. 890-897, 2001.

[14] E. Olson, M. Walter, S. Teller and J. Leonard, "Single-cluster Spectral Graph Partitioning for Robotics Applications," in *2005 Robotics: Science and System Conference*, Cambridge, Massachusetts, USA, 2005.

[15] M. S. Arulampalam, S. Maskell, N. Gordon and T. Clapp, "A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174-188, 2002.

[16] M. Montemerlo, S. Thrun, D. Koller and B. Wegbreit, "FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem," in *National Conference on Artificial Intelligence*, Edmonton, Canada, 2002.

[17] M. Montemerlo, S. Thrun, D. Koller and B. Wegbreit, "FastSLAM 2.0: An Improved Particle Filtering Algorithm for Simultaneous Localization and Mapping that Provably Converges," in *International Conference on Artificial Intelligence*, Acapulco, Mexico, 2003.

[18] A. Doucet, N. d. Freitas, K. Murphy and S. Russell, "Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks," in *UAI'00 Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, San Francisco, CA, USA, 2000.

[19] K. P. Murphy, "Bayesian Map Learning in Dynamic Environments," in *Advances in Neural Information Processing Systems*, Denver, CO, USA, 1999.

[20] G. Grisetti, C. Stachniss and W. Burgard, "Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34-46, 2007.

[21] G. Grisetti, G. D. Tipaldi, C. Stachniss, W. Burgard and D. Nardi, "Fast and Accurate SLAM with Rao-Blackwellized Particle Filters," *Robotics and Autonomous Systems*, vol. 55, no. 1, pp. 30-38, 2007.

[22] D. Hahnel, W. Burgard, D. Fox and S. Thrun, "An Efficient FastSLAM Algorithm for Generating Maps of Large-scale Syclic Environments from Raw Laser Range Measurements," in *2003 IEEE/RSJ International Conference on Intelligent Robots and*

Systems, Las Vegas, Nevada, USA, 2003.

[23] C. Stachniss, G. Grisetti, W. Burgard and N. Roy, "Analyzing Gaussian Proposal Distributions for Mapping with Rao-Blackwellized Particle Filters," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, CA, USA, 2007.

[24] 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.

[25] F. Lu and M. Evangelos, "Globally Consistent Range Scan Alignment for Environment Mapping," *Autonomous Robots*, vol. 4, no. 4, pp. 333-349, 1997.

[26] U. Frese, P. Larsson and T. Duckett, "A Multilevel Relaxation Algorithm for Simultaneous Localization and Mapping," *IEEE Transactions on Robotics*, vol. 21, no. 2, pp. 196-207, 2005.

[27] F. Dellaert and M. Kaess, "Squarse Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing," *The International Journal of Robotics Research*, vol. 25, no. 12, pp. 1181-1203, 2006.

[28] G. Grisetti, C. Stachniss, S. Grzonka and W. Burgard, "A Tree Parameterization for Efficiently Computing Maximum Likelihood Maps Using Gradient Descent," in *2007 Robotics: Science and System Conference*, Atlanta, Georgia, USA, 2007.

[29] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige and W. Burgard, "g²o: A General Framework for Graph Optimization," in *2011 IEEE International Conference on Robotics and Automation*, Shanghai, China, 2011.

[30] E. Olson, J. Leonard and S. Teller, "Fast Iterative Alignment of Pose Graphs with Poor Initial Estimates," in *2006 IEEE International Conference on Robotics and*

Automation, Orlando, Florida, USA, 2006.

[31] A. Howard, M. J. Mataric and G. Sukhatme, "Relaxation on a Mesh: A Formalism for Generalized Localization," in *2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Maui, HI, USA, 2001.

[32] H. Kretzschmar and C. Stachniss, "Imformation-theoretic Compression of Pose Graphs for Laser-based SLAM," *The International Journal of Robotics Research*, vol. 31, no. 11, pp. 1219-1230, 2012.

[33] E. Olson, J. Leonard and S. Teller, "Spatially-adaptive Learning Rates for Online Incremental SLAM," in *2007 Robotics: Science and Systems Conference*, Atlanta, Georgia, USA, 2007.

[34] G. Grisetti, D. L. Rizzini, C. Stachniss, E. Olson and W. Burgard, "Online Constraint Network Optimization for Efficient Maximum Likelihood Map Learning," in *2008 IEEE International Conference on Robotics and Automation*, Pasadena, CA, USA, 2008.

[35] M. Kaess and A. Ranganathan, "iSAM: Incremental Smoothing and Mapping," *IEEE Transaction on Robotics*, vol. 24, no. 6, pp. 1365-1378, 2008.

[36] M. Kaess, H. Johannsson, R. Roberts, V. Ila and J. J. Leonard, "iSAM2: Incremental Smoothing and Mapping Using the Bayes Tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216-235, 2011.

[37] C. Cadena, D. Galvez-Lopez and J. D. Tardos, "Robust Place Recognition with Stereo Sequences," *IEEE Transactions on Robotics*, vol. 28, no. 4, pp. 871-885, 2012.

[38] M. Cummins and P. Newman, "FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647-665, 2008.

[39] M. Cummins and P. Newman, "Appearance-only SLAM at Large Scale with FAB-MAP 2.0," *The International Journal of Robotics Research,* vol. 30, no. 9, pp. 1100-1123, 2011.

[40] E. Olson and P. Agarwal, "Inference on Networks of Mixtures for Robust Robot Mapping," *The International Journal of Robotics Research,* vol. 32, no. 7, pp. 826-840, 2013.

[41] N. Sunderhauf and P. Protzel, "Switchable Constraints for Robust Pose Graph SLAM," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vilamoura, Algarve, Portugal, 2012.

[42] G. H. Lee, F. Fraundorfer and M. Pollefeys, "Robust Pose-graph Loop-closures with Expectaion Maximization," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Tokyo, Japan, 2013.

[43] P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss and W. Burgard, "Robust Map Optimization Using Dynamic Covariance Scaling," in *2013 IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany, 2013.

[44] Y. Latif, C. Cadena and J. Neira, "Robust Loop Closing over Time for Pose Graph SLAM," *The International Journal of Robotics Research,* vol. 32, no. 14, pp. 1611-1626, 2013.

[45] P. J. Besl and N. D. McKay, "A Method for Registration of 3-D Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 14, no. 2, pp. 239-256, 1992.

[46] Y. Chen and G. Medioni, "Object Modelling by Registration of Multiple Range Images," *Image and Vision Computing,* vol. 10, no. 3, pp. 145-155, 1992.

[47] Z. Zhang, "Iterative Point Matching for Registration of Free-form Curves and

Surfaces," *International Journal of Computer Vision*, vol. 13, no. 2, pp. 119-152, 1994.

[48] A. Diosi and L. Kleeman, "Fast Laser Scan Matching Using Polar Coordinates," *The International Journal of Robotics Research*, vol. 26, no. 10, pp. 1125-1153, 2007.

[49] A. Censi and S. Carpin, "HSM3D: Feature-less Global 6DOF Scan-matching in the Hough/Radon Domain," in *2009 IEEE International Conference on Robotics and Automation*, Kobe, Japan, 2009.

[50] A. Censi, L. Iocchi and G. Grisetti, "Scan Matching in the Hough Domain," in *2005 IEEE International Conference on Robotics and Automation*, Barcelona, Spain, 2005.

[51] R. B. Rusu, N. Blodow and M. Beetz, "Fast Point Feature Histograms (FPFH) for 3D Registration," in *2009 IEEE International Conference on Robotics and Automation*, Kobe, Japan, 2009.

[52] H. Bulow and A. Birk, "Spectral Registration of Noisy Sonar Data for Underwater 3D Mapping," *Autonomous Robots*, vol. 30, no. 3, pp. 307-331, 2011.

[53] J. L. Bentley, "Multidimensional Binary Search Trees Used for Associative Searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509-517, 1975.

[54] M. Greenspan and M. Yurick, "Approximate K-D Tree Search for Efficient ICP," in *Fourth International Conference on 3-D Digital Imaging and Modeling*, Banff, Canada, 2003.

[55] F. Lu and M. Evangelos, "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.

[56] L. Armesto, J. Minguez and L. Montesano, "A Generalization of the Metric-Based Iterative Closest Point Technique for 3D Scan Matching," in *2010 IEEE International Conference on Robotics and Automation*, Anchorage, USA, 2010.

[57] J. Minguez, F. Lamiraux and L. Montesano, "Metric-Based Scan Matching Algorithms for Mobile Robot Displacement Estimation," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, Barcelona, Spain, 2005.

[58] A. V. Segal, D. Haehnel and S. Thrun, "Generalized-ICP," in *2009 Robotics: Science and System Conference*, Seattle, USA, 2009.

[59] M. Magnusson, R. Elsrud, L.-E. Skagerlund and T. Duckett, "3D Modelling for Underground Mining Vehicles," in *Conference on Modeling and Simulation for Public Safety*, Linkping, Sweden, 2005.

[60] 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 *2009 IEEE International Conference on Robotics and Automation*, Kobe, Japan, 2009.

[61] A. Das, J. Servos and S. L. Waslander, "3D Scan Registration Using the Normal Distributions Transform with Ground Segmentation and Point Cloud Clustering," in *2013 IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany, 2013.

[62] 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*, vol. 31, no. 12, pp. 1377-1393, 2012.

[63] E. Takeuchi and T. Tsubouchi, "A 3-D Scan Matching Using Improved 3-D Normal Distributions Transform for Mobile Robotic Mapping," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 2006.

[64] C. Ulas and H. Temeltas, "3D Multi-layered Normal Distribution Transform for Fast and Long Range Scan Matching," *Journal of Intelligent & Robotic Systems*, vol. 71,

no. 1, pp. 85-108, 2013.

[65] B. Douillard, J. Underwood, N. Kuntz, V. Vlaskine, A. Quadros, P. Morton and A. Frenkel, "On the Segmentation of 3D LIDAR Point Clouds," in *2011 IEEE International Conference on Robotics and Automation*, Shanghai, China, 2011.

[66] C. Tongtong, D. Bin, L. Daxue, Z. Bo and L. Qixu, "3D LIDAR-based Ground Segmentation," in *2011 First Asian Conference on Pattern Recognition*, Beijing, China, 2011.

[67] F. Moosmann, O. Pink and C. Stiller, "Segmentation of 3D Lidar Data in Non-flat Urban Environments Using a Local Convexity Criterion," in *2009 IEEE Intelligent Vehicles Symposium*, Xi'an, China, 2009.

[68] J. Lam, K. Kusevic, P. Mrstik, R. Harrap and M. Greenspan, "Urban Scene Extraction from Mobile Ground Based LIDAR Data," in *5th International Symposium on 3D Data Processing, Visualization and Transmission*, Paris, France, 2010.

[69] Y. Zhou, Y. Yu, G. Lu and S. Du, "Super-Segments Based Classification of 3D Urban Street Scenes," *International Journal of Advanced Robotics Systems*, vol. 9, no. 248, pp. 1-8, 2012.

[70] J. Papon, A. Abramov and M. Schoeler, "Voxel Cloud Connectivity Segmentation - Supervoxels for Point Clouds," in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, Oregon, Portland, 2013.

[71] E. H. Lim and D. Suter, "3D Terrestrial LIDAR Classifications with Super-voxels and Multi-scale Conditional Random Fields," *Computer-Aided Design*, vol. 41, no. 10, pp. 701-710, 2009.

[72] M. Magnusson, "The Three-Dimensional Normal-Distributions Transform - an Efficient Representation for Registration, Surface Analysis, and Loop Detection,"

Ph.D. dissertation, Orebro University, Dec. 2009.

[73] P. Biber, S. Fleck and W. Strasser, "A Probabilistic Framework for Robust and Accurate Matching of Point Clouds," in *26th Pattern Recognition Symposium*, Tübingen, Germany, 2004.

[74] J. Sturm, N. Engelhard, F. Endres, W. Burgard and D. Cremers, "A Benchmark for the Evaluation of RGB-D SLAM Systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vilamoura, Portugal, 2012.

[75] B. Douillard, A. Quadros, P. Morton, J. P. Underwood, M. De Deuge, S. Hugosson, M. Hallstrom and T. Bailey, "Scan Segments Matching for Pairwise 3D Alignment," in *2012 IEEE International Conference on Robotics and Automation*, Saint Paul, USA, 2012.

[76] S.-W. Yang, C.-C. Wang and C.-H. Chang, "RANSAC Matching: Simultaneous Registration and Segmentation," in *2010 IEEE International Conference on Robotics and Automation*, Anchorage, USA, 2010.

[77] T. Fawcett, "An Introduction to ROC Analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861-874, 2006.

[78] T. Stoyanov, M. Magnusson and A. J. Lilienthal, "Comparative Evaluation of the Consistency of Three-dimensional Spatial Representations Used in Autonomous Robot Navigation," *Journal of Field Robotics*, vol. 30, no. 2, pp. 216-236, 2013.

[79] G. Pandey, J. R. McBride and R. M. Eustice, "Ford Campus Vision and Lidar Data Set," *The International Journal of Robotics Research*, vol. 30, no. 13, pp. 1543-1552, 2011.

초 록

모바일(mobile) 로봇이 탐사, 구조, 감시, 군사 등 다양한 응용분야에 사용되기 위해서는, 가장 기본적으로 자율 주행 능력이 필요하다. 자율 주행을 위해서 가장 중요한 문제 중 하나가 로봇 스스로 주위 환경에 대한 지도를 작성하고, 그 지도안에서 자신의 위치를 파악하는 것, 즉 simultaneous localization and mapping (SLAM)이다. SLAM의 알고리즘에서 이용하는 주위 환경에 대한 정보는 희소한(sparse) 특징 점(feature)과 조밀한(dense) 점 군(point cloud), 두 가지가 있는 데, 로봇이 자신이 가진 지도를 이용해서, 구체적인 경로 계획(path planning)이나 충돌 회피(collision avoidance)를 수행하기 위해서는 조밀한 점 군을 이용한 지도가 필요하다. 그리고 다양한 형태의 로봇이 폭 넓은 임무를 수행하기 위해서는 반드시 지도가 3차원이어야 한다. 따라서 3차원 조밀한 점 군을 이용한 SLAM 알고리즘이 필요한데, 그것의 성능을 보장하기 위해서는 반드시 강력한 3차원 스캔 정합(scan registration) 알고리즘이 요구된다.

본 논문에서는 3차원 normal distributions transform(NDT)의 성능을 크게 향상시킨 새로운 3차원 정합 알고리즘인 supervoxel-NDT (SV-NDT)를 제안하였다. 3차원 NDT는 모델 스캔(model scan)을 분할할 때, 3차원 정규격자를 이용한다. 하지만 3차원 정규격자는 모델 스캔의 국소적(local) 표면 구조와 관련된 정보를 전혀 이용하지 않기 때문에, 3차원 정규격자를 이용한 정규분포 생성은 큰 정보손실을 야기한다. 하나의 정규분포로 가장 잘 모델링 할 수 있는 면(스캔의 구성요소)의 형태는 평면이다. SV-NDT는 분할단계에서 슈퍼복셀(supervoxel)

생성 알고리즘을 사용함으로써 정보손실을 줄였다. 추가적으로, 데이터 스캔(data scan)의 각 점을 자신에 대응되는(corresponding) 정규분포에 연결시킬 때, 기존의 유클리드 거리(Euclidean distance)대신 유클리드 거리와 데이터 스캔의 국소적 구조를 함께 사용한 함수로 바꿈으로써 데이터 스캔의 국소적 표면의 구조 정보도 함께 이용하도록 하였다.

슈퍼복셀 생성 알고리즘을 이용하여 생성한 정규분포들의 모델링 정확도(modeling accuracy)가 향상되고, 제안한 3차원 정합 알고리즘이 기존의 3차원 NDT와 그 외 널리 사용되는 다른 알고리즘들과 비교하여 강인성(robustness)와 속도 측면에서 더 우수하였음을 합성 데이터 집합(synthetic dataset)과 실제 데이터 집합(real-world dataset)을 이용한 실험을 통해 입증하였다. 추가적으로 대응관계 형성 시 사용되는 함수를 바꾼 효과가 정합 능력에 긍정적인 효과를 끼쳤다는 것도 확인하였다.

주요어: 정규분포변환(Normal distributions transform), 스캔 정합(registration), 슈퍼복셀 분할(supervoxel segmentation), SLAM, 이동 로봇

학 번: 2013-20777