2018 IEEE Intelligent Vehicles Symposium (IV)
Changshu, Suzhou, China, June 26-30, 2018

A Compact Map Representation for Large-scale Environments and Localization Method based on Similarity Measure

Kohei Matsuzaki[1] and Hiromasa Yanagihara[1]

***Abstract*— Targeted at autonomous driving, compact map representation for localization is needed to deal with limited disk space or communication bandwidth. State-of-the-art compact map representations distinguish partial-objects such as lane markings from whole-object data. However, they require that objects are correctly detected in both map generation and localization. Therefore, the localization may fail if either of these conditions is violated due to occlusion, poor road texture, or other complications. In this paper, we propose a novel map generation and localization method that uses whole-object data without object detection. The novel map so produced has compactness comparable to state-of-the-art methods involving object detection because it is described as blocks of quantized representation in a voxel grid. In the localization step, we formulate localization as a similarity search between the map data and differently sampled sensor data. The proposed localization method performs the search at a sub-voxel level while alleviating voxel discretization error. This enables accurate localization even with low-resolution voxels. Experiments show the proposed method achieves sufficient localization accuracy and real-time processing with a map having a data size of 32 kB per km. The storage efficiency is at least 161 times greater than state-of-the-art maps covering the same area.**

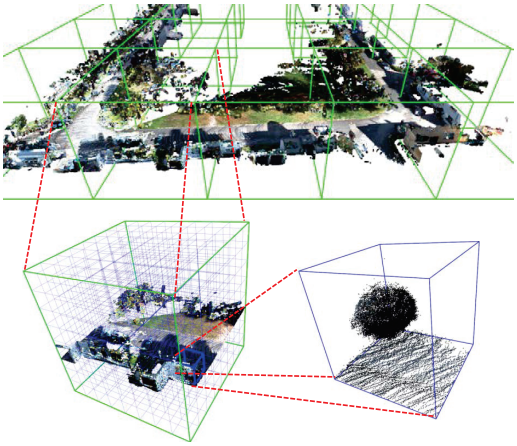


Fig. 1: Illustration of the proposed block map. The green cube and blue cube represent the *block* and the voxel respectively. Top: The 3D point-cloud of the road environment as discretized by block. For clarity, the point-cloud is colored here. Bottom left: The block consists of a set of aligned voxels. Bottom right: An example of voxel data.

I. Introduction

Localization within an environment is one of the basic requirements for autonomous driving. Commercial global navigation satellite system-based localization provides insufficient accuracy for lane-level navigation. Therefore, many methods for accurate localization have been studied that use sensors and pre-generated environmental maps.

The solutions to the localization task must be accurate and efficient on a driving platform. A position error of less than 1 m and an orientation error of less than 1° are necessary in order to identify the self-position at lane-level of accuracy [1]. Furthermore, the real-time performance should be considered for continuous localization during driving.

Most methods are based on the use of images captured by cameras [2]–[7]. However, image-based localization methods typically include an implicit assumption that map generation and localization are performed in a similar illumination environment. Therefore, their performance may be greatly degraded by severe illumination differences (e.g., from daytime to nighttime) [8], [9].

As commercial laser range sensors drop in price, in recent years, point-cloud-based localization methods have attracted increasing attention due to their use of three-dimensional (3D) light detection and ranging (LiDAR) scanners. LiDAR has the advantage of being independent of ambient illumination because it uses emitted radiation. Therefore, using a LiDAR scanner is a reasonable way to achieve robust localization unaffected by illumination changes.

Most localization methods involve a map generation step because they model environmental map data according to their proprietary algorithm. The map data size is one of the most important factors in practical localization. When using the map data in the vehicle, there are two possible scenarios.

In the first scenario, large-scale (e.g., country-, continent-, and world-scale) map data is stored on the disk in advance. It is necessary to reduce the data size per unit to hold all data on the disk. In the second scenario, the map data is streamed to the vehicle using a communication module. This may not work well if the network bandwidth is insufficient due to the streaming of large amounts of data to all vehicles on the road simultaneously. Therefore, in both scenarios, compact map data is required.

Many methods that achieve compact map representation are based on a map consisting only of partial-objects (e.g., lane markings) [10]–[17]. These methods involve an object detection process for both map generation and localization. However, involving the object detection process complicates the tasks because it requires dealing with problems of non-detection and misdetection. These methods assume that objects are correctly detected in both the map generation and

[1]Authors are with Media Recognition Laboratory, Media ICT Department, KDDI Research, Inc., Saitama, Japan {ko-matsuzaki, yanap}@kddi-research.jp

the localization. Therefore, localization may fail if either of these assumptions is violated. Although there are automatic map generation techniques [18], [19], some manual processing is necessary to generate the required quality of map. Therefore, the map generation cost is high in these methods.

In this paper, we undertake a localization that achieves sufficient performance for autonomous driving, as well as a map generation to reduce the required data size. We emphasize the minimization of the map data size while guaranteeing the desired localization accuracy and computational efficiency.

We propose a novel map generation and localization method based on similarity measure using quantized representations. Our map generation method models point-cloud data using a voxel grid, then reduces map data size by quantizing the model representation in each voxel. In order to further reduce the data size, we build a *block* map as shown in Fig. 1. Our localization method matches map data and differently sampled sensor data based on the similarity of quantized representations. It alleviates any voxel discretization error by shifting the reference point of the voxel grid in increments smaller than the grid size. This allows accurate localization even with low-resolution voxels.

Our map generation and localization method is completely LiDAR-based, so we can benefit from the advantages of LiDAR, such as its immunity to illumination changes. Furthermore, it does not suffer from the difficulty of object detection because our map contains all objects in the sensor measurement, making it easy to generate and update the map.

The contributions of this paper are summarized as follows:

1) We formulate localization as a similarity search between a map data and differently sampled sensor data.
2) The proposed method provides accurate localization even with low-resolution voxels by alleviating voxel discretization error.
3) We generate highly compact map data from a detailed point-cloud, based on a voxel grid, quantized representation, and the block.

II. Related Work

Many methods have been proposed for map generation and localization based on LiDAR measurements. We distinguish here between a partial-object map class that involves an object detection process and a whole-object map class that maps the entire measurement space.

Partial-object map. There are many methods that use only specific objects such as lane markings for map generation and localization. These methods minimize the map data size by storing only the minimum components required for accurate localization.

Schindler *et al.* [10] proposed to detect lane markings, road markings, and landmarks from a point-cloud measured by laser scanner and to generate a map consisting of those objects. In the localization step, a self-position is calculated by fitting objects detected from sensor data and objects stored in the map [1]. Schreiber *et al.* [11] developed a localization system using a map composed of lane markings and curbs. Soheilian *et al.* [12] constructed a map consisting only of road markings. However, these methods use a camera for the localization, so they may be adversely affected by severe illumination changes.

There are also methods that use a laser scanner for both map generation and localization. Wiest *et al.* [14] initially build a two-dimensional (2D) occupancy grid map from a point-cloud measured by a laser scanner. Then, they extract local features from the 2D map based on MSER algorithm [32] and perform localization using these features. Im *et al.* [15], similarly to Wiest *et al.*, build a 2D occupancy grid map and then extract the vertical corner features representing corners of buildings. Levinson *et al.* [16] detect the ground plane from a point-cloud that includes reflectivity, and generate a map composed only of the road area. In their localization step, objects such as lane markings represented by reflectivity provide a clue to accurate localization. In contrast to Levinson *et al.*, Li *et al.* [17] discard the road area after detecting the ground plane from a point-cloud. After that, they generate a 2D occupancy grid map weighted by the distance from the road center to roadside objects.

Common to partial-object map class is an assumption that objects are correctly detected in both map generation and localization. Localization may fail if either of these assumptions is violated due to occlusion, poor road texture, or other factors. Furthermore, these methods cannot be applied to areas where there is no specific object to serve as a clue.

Whole-object map. There are methods that use all data measured by a laser range sensor for map generation and localization without detecting specific objects.

The most popular method is the iterative closest point (ICP) algorithm [20] and its variants [21], [22]. In these methods, the measured point-cloud is treated as map data. Jian *et al.* [23] initially suggested representing point-cloud in a gaussian mixture model. An occupancy grid map is often used for a monte carlo localization algorithm [24]. OctoMap [25] is a popular implementation that enables the efficient storage of an occupancy grid map. Magnusson *et al.* [26] developed the 3D normal distributions transform (NDT). This method first divides the data using a voxel grid and then represents the individual data as a multivariate normal distribution. As a result, the map data size is greatly reduced compared with the original point-cloud. NDT-D2D [27] solves the registration task as a minimization of the L_2 distance between NDT models by representing not only a fixed point-cloud but also a moving point-cloud with the NDT. It is also shown that this method is more accurate and faster than the standard NDT. More recently, Wolcott *et al.* [28], [29] proposed gaussian mixture maps (GMM). This method divides the point-cloud using a 2D grid across the xy plane and representing individual data with a one-dimensional (1D) gaussian mixture model in the z direction.

All of these methods deal with a whole 3D point-cloud for map generation and localization. Therefore, these methods can avoid the difficulties associated with object detection. As a result, there is an advantage that the map generation cost is low. However, these methods share the problem that the map data size is very large as compared with partial-object

map class.

Our map belongs to the whole-object map class, but its data size is small comparable to the partial-object map class.

III. Block Map

In this section, we present our map generation method. Our goal is to minimize the map data size while achieving the localization accuracy and computational time required for autonomous driving. The upper part of Fig. 2 shows the pipeline of the proposed map generation method. The proposed method assumes that there is a point-cloud build by a simultaneous localization and mapping (SLAM) technique or the use of a global pose sensor externally.

We propose to discretize the 3D space with a voxel grid and model the data in each voxel as a quantized representation. This can significantly reduce the data size as compared with the method of storing data represented in floating point format. Furthermore, our method stores data as a block that arranges multiple voxels as one unit, as shown in Fig. 1. We reduce the data size with respect to spatial information by representing the spatial position of each voxel as its relative position inside the block.

A. Data modeling

Given a point-cloud, we divide it into blocks and divide each block into voxels. Then we model data points in each voxel as the following representation:

$$\boldsymbol{p}_i = \frac{1}{M_i} \sum_{j=0}^{M_i-1} (\boldsymbol{d}_{i,j} - \boldsymbol{c}_i), \tag{1}$$

where $\boldsymbol{d}_{i,j}$ with $\forall j \in \{0, ..., M_i - 1\}$ are data points in the i-th voxel and $\boldsymbol{c}_i$ is the corner point where all components are minimum for the i-th voxel. That is, $\boldsymbol{p}_i$ is the mean position of the data in the i-th voxel relative to $\boldsymbol{c}_i$.

Then we quantize $\boldsymbol{p}_i = (x_i, y_i, z_i)$ into an integer index using the linear quantizer q:

$$q(\boldsymbol{p}_i) = \lfloor \frac{x_i w_x}{l} \rfloor + \lfloor \frac{y_i w_y}{l} \rfloor w_x + \lfloor \frac{z_i w_z}{l} \rfloor w_x w_y, \tag{2}$$

where $\lfloor \cdot \rfloor$ is a floor function, l is the voxel grid size, and w_x, w_y and w_z are the number of divisions of one side of the voxel in the x, y, and z directions. The possible range of $q(\boldsymbol{p}_i)$ is $[0, w_x w_y w_z - 1]$. Fig. 3 illustrates an example of $\boldsymbol{p}_i$ and $q(\boldsymbol{p}_i)$, where $w_x = 3$, $w_y = 3$, $w_z = 3$, $g(x_i w_x / l) = 1$, $g(y_i w_y / l) = 0$, $g(z_i w_z / l) = 2$, and $q(\boldsymbol{p}_i) = 19$.

B. Block generation

We arrange a finite number of voxels into one unit which we call the block. The block is a set of N_x, N_y, and N_z voxels in the x, y, and z directions. Fig. 1 shows an overview of the block map. Each block is generated by calculating the three types of information: a global block center position, a binary string describing the relative position of non-empty voxels, and a binary string describing the quantized representations of non-empty voxels.

In order to reduce the map data size, we store a relative position index within the block instead of the global position

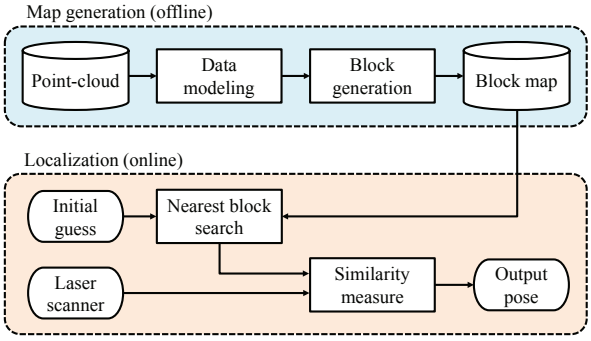


Fig. 2: The pipeline of the proposed system.

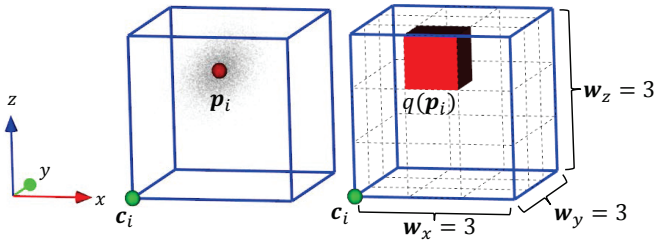


Fig. 3: Illustration of the data representation and its quantization. The green point, blue cube, red sphere, and red cube respectively represent the corner point where all components are minimum, the voxel, the data representation, and the quantized representation.

for each non-empty voxel. Therefore, we assign the following index to the n_x-th, n_y-th, and n_z-th, as counted from zero, voxels in the x, y, and z directions:

$$k = n_x + n_y N_x + n_z N_x N_y. \tag{3}$$

Then we create an $N_x N_y N_z$ dimensional binary string where the k-th bit is '0' if the k-th voxel is empty, otherwise '1'.

Regarding the quantized representation, we convert the integer index in Eq. 2 for $i = k$ to the shortest binary string b_k that can cover the range $[0, w_x w_y w_z - 1]$. Then we create a binary string in which b_k are arranged in ascending order of the corresponding k. Note that the b_k corresponding to empty voxels are not included in this binary string.

IV. Map-based Localization

We propose a localization method based on the similarity measure. As shown in the lower part of Fig. 2, the input of the localization method is an initial guess of pose $\boldsymbol{\xi}_0^m = [\boldsymbol{R}^m | \boldsymbol{t}^m]$ with a rotation matrix $\boldsymbol{R}^m$ and translation vector $\boldsymbol{t}^m$ in the map coordinate system, the block map described in Section III, and sensor data from a laser scanner. Given environmental map data and sensor data, localization can be regarded as a coordinate transformation from the sensor coordinate system to the map coordinate system.

We first determine the nearest block from map data based on an initial guess. Then we generate blocks from different samples of sensor data. We search for the block in sensor data that most closely resembles the block from the map, using a similarity measure. In this way, we can find sampling parameters corresponding to that block.

A. Problem Formulation

We formulate the localization as a similarity search between the map data and differently sampled sensor data. We first search for the block with its center position $\boldsymbol{C}^m$ nearest to the translation parameters of initial guess $\boldsymbol{t}^m$ from the block map. We obtain a block with parameters $\Theta^m = \{\boldsymbol{C}^m, N_x, N_y, N_z\}$, i.e. $N = N_x N_y N_z$ voxels centered on $\boldsymbol{C}^m$ that is generated from map point-cloud $\mathcal{P}^m$:

$$B^m(\mathcal{P}^m, \Theta^m) = \{b_0^m, ..., b_{N-1}^m\}, \tag{4}$$

where b_k^m with $\forall k \in \{0, ..., N-1\}$ are the binary strings describing the quantized representations of the k-th voxel presented in Section III-B, which can include not available (N/A), corresponding to an empty voxel.

Then we calculate the position $\boldsymbol{C}^s = \boldsymbol{C}^m - \boldsymbol{t}^m$, which corresponds $\boldsymbol{C}^m$ in the sensor coordinate system. If the initial guess does not include errors, $\boldsymbol{C}^s$ and $\boldsymbol{C}^m$ represent the same position in the real world. However, this will not be the case if the initial guess generally has errors.

In order to correct the errors in the initial guess, we apply various pose transformations to the sensor point-cloud $\mathcal{P}^s$ using the parameter $\boldsymbol{v} = (t_x, t_y, t_z, r_x, r_y, r_z) \in SE(3)$. Then we generate blocks with parameters $\Theta^s = \{\boldsymbol{C}^s, N_x, N_y, N_z\}$ in the same manner described in Section III:

$$B^s(T(\mathcal{P}^s, \boldsymbol{v}), \Theta^s) = \{b_0^s, ..., b_{N-1}^s\}, \tag{5}$$

where $T(\mathcal{P}, \boldsymbol{v})$ is a parameterized transformation function. Note that the number of voxels (i.e., N_x, N_y, and N_z) in B^s is always set to equal those in B^m.

We propose to give a similarity score to a pair of voxels with the same relative position in the block (i.e., they have the same k in Eq. 3) and the same representation:

$$f(b_k^m, b_k^s) = \begin{cases} 1 & \text{if} \quad b_k^m = b_k^s, b_k^m \neq \text{N/A}, b_k^s \neq \text{N/A} \\ 0 & \text{otherwise} \end{cases}. \tag{6}$$

We formulate the localization as the maximization of an objective function based on the similarity measure:

$$\underset{\boldsymbol{v}}{\text{argmax}} \ s(\boldsymbol{v}) = \sum_{k=0}^{N-1} f(b_k^m, b_k^s). \tag{7}$$

B. Similarity Measure between Blocks

We sample pose transformation parameters to correct the errors of the initial guess. First, we generate blocks from sensor point-clouds with various pose transformations. Then we search for the block that most closely resembles the block of the map. Finally, we localize the self-position from the transformation parameters corresponding to that block.

1) Rotation transformation: First of all, we rotate the sensor point-cloud $\mathcal{P}^s$ using an initial guess of the rotation matrix $\boldsymbol{R}^m$ in order to transform it from the sensor coordinate system to the map coordinate system with respect only to rotation. Then we further rotate $\mathcal{P}^s$ using rotation offsets o_r to correct the rotation error in the initial guess. The o_r are set to a specified number of offsets equally spaced within the rotation search range $[-\theta, \theta]$. The subsequent processing is performed independently for each of the transformed sensor point-clouds.

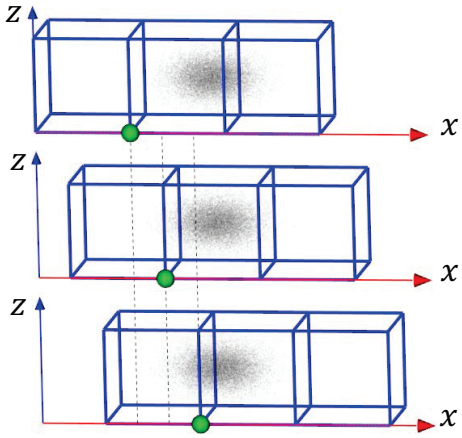


Fig. 4: Illustration of the voxel discretization error. The green point and blue cube represent the reference point and voxel respectively. The coordinates of the point-cloud are fixed and those of the reference points are shifted. It is demonstrated that different reference points generate voxels that discretize different spaces.

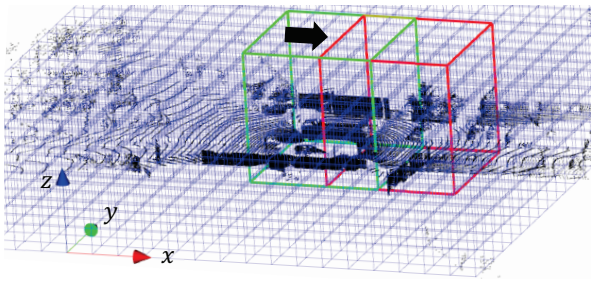


Fig. 5: Sliding-based block generation. The green and red cubes represent blocks before and after sliding.

2) Phase shift data modeling: Next, we model the sensor point-cloud in the same manner as in Section III-A. However, the discretization of spaces using a voxel grid between different coordinate systems can produce the problem of voxel discretization error. As shown in Fig. 4, the mismatch of reference points forming the voxel grid causes a voxel phase error between the map coordinate system and the sensor coordinate system, resulting in modeling errors. Typically, this problem can be addressed by using high-resolution voxels. This is because modeling errors can be avoided in most voxels with a resolution that is sufficiently finer than the scale of the target objects. However, the use of high-resolution voxels increases the map data size.

In order to address this problem without increasing the map data size, we propose to alleviate the voxel discretization error by discretizing the space while shifting the reference point. That is, we model the same data using a voxel grid with a different phase. Specifically, we first select an arbitrary reference point μ_0. Then we shift the reference point in the range $[\mu_0, \mu_0 + l)$ using a step size λ that is smaller than the voxel grid size l. Here, phase shift offsets o_p are defined as λ

multiplied by the number of siftings. Finally, we model the sensor point-cloud independently for each reference point.

3) Sliding-based block generation: We generate blocks consisting of $N_x N_y N_z$ voxels from all modeled data. As shown in Fig. 5, we calculate the block by sliding the 3D window in, voxel by voxel, in order to efficiently compute the translation transformation. That is, we just assign the index k in Eq. 3 which represents the relative position within the block of each voxel. Therefore, a different k is assigned in different blocks even if to the same voxel.

4) Initial guess correcting: Among all blocks generated from the sensor point-cloud, we search for a block that maximizes the similarity measure in Eq. 7. Then we calculate the transformation parameters corresponding to the block $\hat{v}$ from its rotation offset, phase shift offset, and sliding amount mentioned above. Finally, we obtain self-position by correcting the initial guess with $\hat{v}$.

C. Coarse-to-Fine Strategy

For efficient search, we employ a coarse-to-fine strategy. We perform multi-stage processing, which hierarchically searches with progressively finer offsets. Note that this method uses only a single map and does not require any additional data. Let the search range be $\mathcal{S}$.

In the first stage, we search $\mathcal{S}$ exhaustively, using the coarsest offsets $o_{r,1}$ and $o_{p,1}$. As a result, we obtain the estimated pose $\boldsymbol{\xi}_1$. In the next stage, we only search narrower subspaces using finer offsets $o_{r,2}$ and $o_{p,2}$. Here, we consider $\boldsymbol{\xi}_1$ as an initial guess.

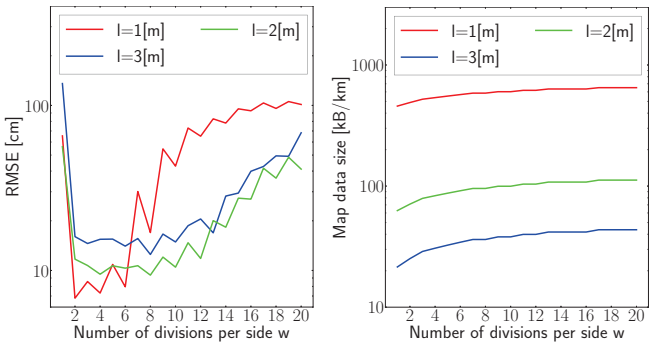
In the subsequent stages, we continue to search a narrower subspace while progressively decreasing the offsets.

V. Experiments

In this section, we evaluate the performance of the proposed map generation and localization method. We demonstrate that our map is much more compact than state-of-the-art methods, and our localization method provides adequate performance for autonomous driving. Unlike other localization methods using a high-performance GPU [17], [29], this proposed method achieves real-time processing simply by using a CPU.

A. Experimental Setup

In these experiments, we used the KITTI Vision Benchmark Suite [30], which is a publicly available dataset. This includes data measured by traveling in cities using a vehicle platform equipped with a 3D laser scanner and a GPS/IMU system. A point-cloud map was constructed from the laser scan data using the ground-truth pose given by the output of the GPS/IMU system. Next, the block map was generated using the proposed method. In the map generation step, block volume was set to 24 m $\times$ 24 m $\times$ 24 m based on our preliminary experimental results. In the localization step, the nearest block and its surrounding eight blocks were integrated and recalculated as one block with a volume of 72 m $\times$ 72 m $\times$ 24 m. The larger volume will widen the range of available scan data, but it will also increase computation



(a) Localization error (b) Map data size

Fig. 6: The accuracy of the localization and map data size with different discretization level.

time. The step size of reference point λ was set to 10 cm to achieve decimeter-level accuracy. All experiments were performed on a 3.4 GHz Intel Core i7-6800K with 32 GB of RAM. The proposed method was implemented using a CPU-based parallel programming technique.

B. Effect of Discretization Level

We verified the performance dependence of the two parameters that determine the discretization level. They are the voxel grid size l and the number of divisions per side of the voxel w in Eq. 2. We assumed the usability of the same value in x, y, and z directions for w.

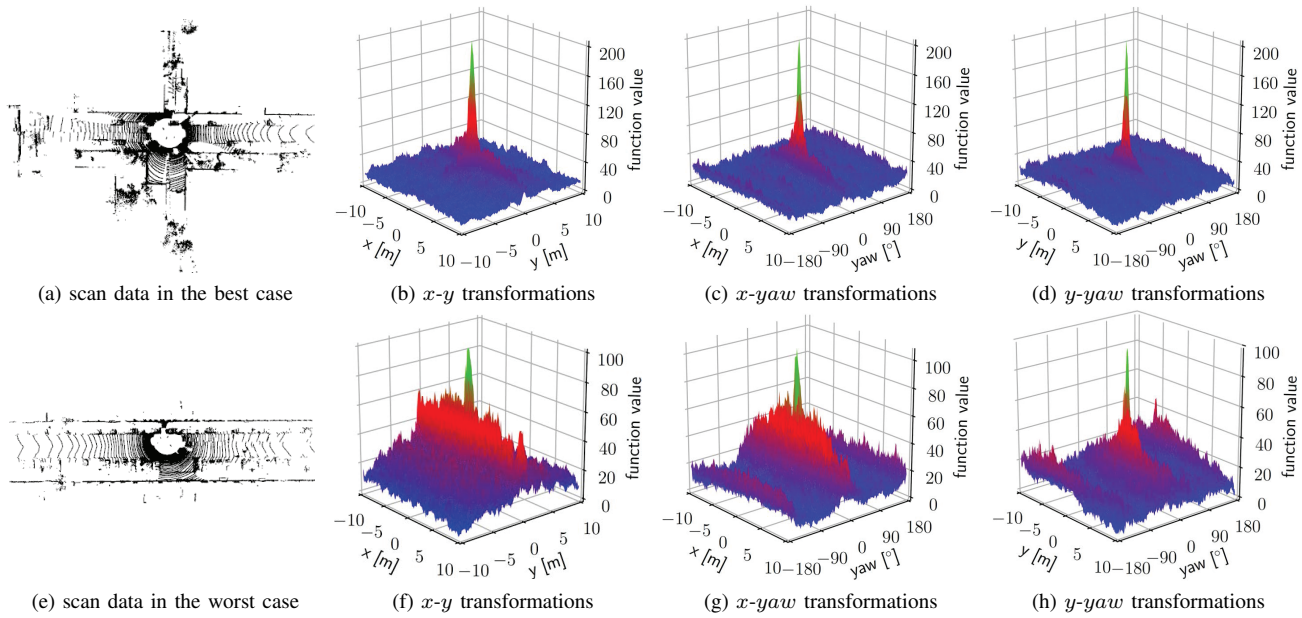
For this experiment, we selected some laser scans, which were transformed away from the ground-truth pose and then localized to the map. We set random translations in the range $[-10$ m, 10 m$]$ in x and y directions, $[-1$ m, 1 m$]$ in the z direction, and random rotations in the range $[-10°, 10°]$ around the yaw-axis. We tried 100 times for each parameter setting and calculated the root mean square error (RMSE) of the position. We also measured the map data size.

We test the perimeters $l \in \{1, 2, 3\}$ in meters and $w \in \{1, 2, ..., 20\}$. For each different l, the number of voxels was set so that the volumes of the block would be common. Fig. 6 shows all results. As shown in Fig. 6a, the accuracy of localization tends to decrease as l becomes coarser. This is because the discrimination of the map representation decreases. A too-large w leads to large errors because the repeatability of the quantization in Eq. 2 decreases. Fig. 6b shows that the map data size strongly depends on l while it is largely independent of w. This is due to the number of voxels in a block is inversely proportional to l^3, and each voxel has a binary string of length $ceil(log_2(w^3))$.

In order to achieve both accurate localization and compact map data size, we use the parameters $\{l, w\} = \{2, 4\}$ in all subsequent experiments.

C. Shape of the Objective Function

The proposed method was assessed for sensitivity to translational and rotational errors. For this experiment, two scan data values that obtained the best and the worst accuracy in Section V-B were selected. Values of the objective function



(a) scan data in the best case (b) x-y transformations (c) x-yaw transformations (d) y-yaw transformations

(e) scan data in the worst case (f) x-y transformations (g) x-yaw transformations (h) y-yaw transformations

Fig. 7: Shape of the objective function. (a)-(g) and (e)-(h) correspond to the best and the worst case in Sec. V-B, respectively.

TABLE I: Comparison of map data size

Map class		Method	Data size [MB/km]		Map representation
			Non compressed	Lossless compressed	
partial-object	sparse	Im *et al.* [15]	0.014	-	vertical corner feature to model wall of building
		Wiest *et al.* [14]	0.01	-	MSER feature to model landmark
		Gwon *et al.* [13]	**0.0015**	-	polynomial curve to model lane marking
	dense	Levinson *et al.* [16]	-	6.21	road surface reflectivity
		Road DNA [17]	1.5	-	occupancy weighted by distance from road center
whole-object	dense	ICP [20]	1477.13	1306.77	point-cloud
		GMM [28]	30	-	1D normal distribution of grid
		OctoMap [25]	-	22.54	occupancy of grid
		NDT [26], [27]	9.54	5.16	3D normal distribution of grid
		Proposal	**0.082**	**0.032**	block composed of quantized mean of grid

$s(\boldsymbol{v})$ in Eq. 7 were calculated for a range of (x, y), (x, yaw), and (y, yaw) values around the ground-truth. Fig. 7 shows the two scan data and the shapes of the objective function represented as 3D plots. Figs. 7a-7d and Figs. 7e-7h correspond to scan data in the best case and the worst case respectively.

In the best case, the shape of the objective function is a sharp convex. The function value drops significantly, even for translations smaller than the voxel grid size or rotations less than 1°. In the worst case, the sharpness decreases compared to the best case. This is due to the poor shape characteristics of the environment as shown in Fig. 7e. However, the objective function is still sensitive to translational and rotational errors so that the required accuracy may be achieved.

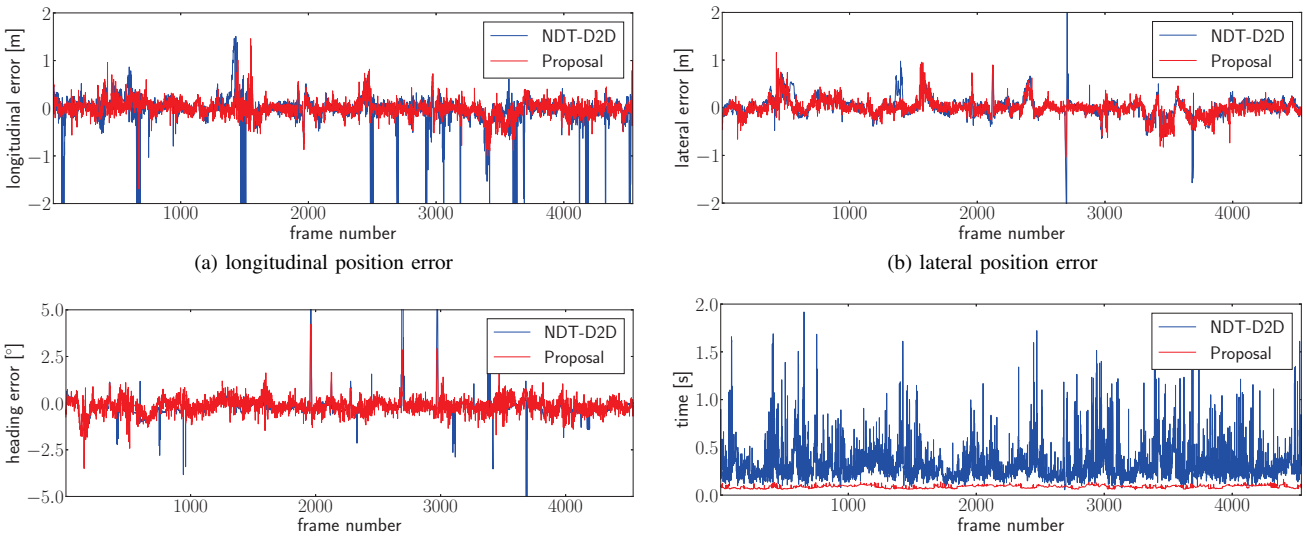
D. Comparison of Map Data Size

We compare the map data size of the proposed method with the state-of-the-art methods. We distinguished between the partial-object maps and the whole-object maps described in Section II. All reports are summarized in Table I.

For ICP [20], NDT[1] [26], [27], OctoMap[2] [25], and the proposed method, we report the data size measured from the KITTI dataset using the open-source softwares. The grid size for NDT and OctoMap were set to 1.6 m and 0.2 m respectively, which yielded the best accuracy in the research of Stoyanov *et al.* [31]. For ICP, NDT, and the proposed method, we also measured the lossless compressed file size in zip format. For other methods, we referred to the data size reported in the literature.

Im *et al.* extracted 271 corners from 2 km of a road and generated a corner map of 28 kB [15]. Wiest *et al.*'s map consists of 1997 MSER features [32] extracted from 4.6 km of a road that was covered with approximately 10 kB per km [14]. Gwon *et al.* modeled 55 line segments in 3.7 km of a road with 715 floating points [13]. Levinson *et al.* generated losslessly compressed images that required approximately 10

[1]https://github.com/OrebroUniversity/perception_oru-release
[2]https://github.com/OctoMap/octomap



(a) longitudinal position error

(b) lateral position error

(c) heading orientation error

(d) computational time

Fig. 8: Continuous localization results for NDT-D2D and this proposal.

MB per mile [16]. Li *et al.* obtained a map (Road DNA) of 30 MB from 20 km of measurements on an university campus [17]. Since the former three maps are composed only of sparse objects, their data size is very small. However, localization fails if not enough objects can be detected in these methods. Since the latter two maps are composed of dense objects, the data size is relatively large in the partial-object map method.

For ICP, we assumed the point-cloud to be a map, but too large for large-scale measurement. Wolcott *et al.* reported that they can store 1 km of their Gaussian Mixture Maps (GMM) using less than 30 MB of disk space [28]. The OctoMap became large, as high-resolution voxels were required for accurate localization. Although NDT realized efficient data modeling, it used many parameters in each voxel. In comparison, the proposed method generates a map with a much smaller data size.

One of the reasons is that the amount of data for each voxel is small. Our binary strings describing relative positions and quantized representations of non-empty voxels provide much better storage efficiency than other methods that store floating points. Another reason is that it uses a low-resolution grid compared with other methods. This is because our method achieves accurate localization even with coarse voxels by performing a similarity search at a sub-voxel level while alleviating voxel discretization error. As a result, our map has a data size comparable to the partial-object map, even though it is a whole-object map.

Furthermore, the proposed map has a higher lossless compression ratio than others. It has high redundancy because most of the data consists of quantized representations. Therefore, it can be compressed more efficiently than other maps composed of real-valued representations. As a result, the proposed map size reaches 32 kB per km, which represents a storage efficiency that is at least 161 times greater than state-of-the-art maps covering the same area.

TABLE II: Comparison of average absolute values

	Lon. [cm]	Lat. [cm]	Head. [°]	Time [ms]
NDT-D2D [27]	26.55	13.79	0.39	321.64
Proposal	**12.28**	**12.09**	**0.35**	**78.79**

E. Continuous Localization

We evaluated the performance of the proposed method for continuous localization on the sequence 00 included in the KITTI dataset, which is a 3.72 km route in an urban area. The proposed method was compared with NDT-D2D[1] [27], which achieves the most compact map representation among conventional whole-object maps and faster localization than the standard NDT [26]. As an indicator of localization performance, the longitudinal position error (Lon.), lateral position error (Lat.), heading orientation error (Head.), and the computational time (Time) were used. The given pose from GPS/IMU was assumed to be the ground-truth. A constant acceleration model was used to calculate an initial guess from the previous localization results and this was then used as input for the algorithm. The localization was considered to be a failure if the position error relative to the ground-truth increased to 5 m, which is the distance that the algorithms will not be able to recover. In that case, the localization was restarted from the next frame with a GPS/IMU data value as an initial guess.

All results are shown in Fig. 8, and the average absolute values are summarized in Table II. The number of failures for NDT-D2D and the proposed method is 36 and 0 respectively, over 4541 frames.

As can be seen, the proposed method achieves accuracy comparable to NDT-D2D. The proposed method results in a longitudinal position error, lateral position error and heading orientation error of 12.28 cm, 12.09 cm, and 0.35° respectively. This performance is sufficient for autonomous driving

since it enables identification of self-positioning with lane-level accuracy. Both methods sometimes return large position errors when encountering an environment with poor shape characteristics since there are few clues for localization. NDT-D2D often failed due to the well-known local minima problem if a poor initial guess was given. On the other hand, the proposed method has little dependence on the initial guess because it finds a global optimal solution within a search range. Although the proposed method occasionally returns a large heading error when turning a corner, an accurate solution is obtained in a subsequent frame for the same reason.

The average time for NDT-D2D and the proposed method is 321.64 ms and 78.79 ms respectively. NDT-D2D is often slow because it includes an iterative optimization process; i.e., computational complexity increases when convergence is difficult. In contrast, the proposed method is always fast since it is based on a non-iterative similarity search, resulting in a speed of less than 100 ms even on a CPU. It has sufficient efficiency for real-time processing because its computational frequency is comparable to the repetition rate of modern LiDAR scanners (e.g., 10 Hz).

VI. Conclusions

In this paper, we proposed a method to achieve fast and accurate localization for autonomous driving using a highly compact map representation. We formulated the localization as a similarity search between the map data and differently sampled sensor data. The proposed map generation method can avoid the difficulties associated with object detection since it maps entire measurement space.

Experiments on a dataset with a wide range of measurements showed that the proposed localization method provides adequate performance for autonomous driving tasks. Our map data size is smaller by at least two orders of magnitude than similar maps. The proposed method provides accurate localization even with low-resolution voxels by alleviating voxel discretization error and providing sufficient computational efficiency even when using the CPU.

The next task is to apply the proposed map representation to a global localization method, which does not rely on the initial guess. Because it does not depend on specific objects like lane markings, we also anticipate that the proposed method can be successfully applied to other situations such as drone flights.

References

[1] A. Schindler, "Vehicle Self-Localization with High-Precision Digital Maps," In *Proc. of IV*, 2013, pp. 141–146.

[2] M. A. Brubaker, A. Geiger, and R. Urtasun, "Map-based Probabilistic Visual Self-Localization," *IEEE Trans. on PAMI*, vol. 38, no. 4, pp.652–665, 2016.

[3] H. Lategahn and C. Stiller, "Vision-only Localization," *IEEE Trans. on ITS*, vol. 15, no. 3, pp.1246–1257, 2014.

[4] J. Courbon, Y. Mezouar, and P. Martinet, "Autonomous Navigation of Vehicles from a Visual Memory using a Generic Camera Model," *IEEE Trans. on ITS*, vol. 10, no. 3, pp. 392–402, 2011.

[5] G. Cao, F. Damerow, B. Flade, M. Helmling, and J. Eggert, "Camera to Map Alignment for Accurate Low-Cost Lane-Level Scene Interpretation," In *Proc. of ITSC*, 2016, pp. 498–504.

[6] T. Wu and A. Ranganathan, "Vehicle Localization using Road Markings," In *Proc. of IV*, 2013, pp. 1185–1190.

[7] R. Spangenberg, D. Goehring, and R. Rojas, "Pole-based Localization for Autonomous Vehicles in Urban Scenarios," In *Proc. of IROS*, 2016, pp. 2161–2166.

[8] C. Linegar, W. Churchill, and P. Newman, "Made to Measure: Bespoke Landmarks for 24-hour, All-Weather Localisation with a Camera," In *Proc. of ICRA*, 2016, pp. 787–794.

[9] G. Pascoe, W. Maddern, M. Tanner, P. Pinies, and P. Newman, "NID-SLAM: Robust Monocular SLAM using Normalised Information Distance," In *Proc. of CVPR*, 2017, pp. 1435–1444.

[10] A. Schindler, G. Maier, and Florian Janda, "Generation of High Precision Digital Maps using Circular Arc Splines," In *Proc. of IV*, 2012, pp. 246–251.

[11] M. Schreiber, C. Knöppel and U. Franke, "LaneLoc: Lane Marking based Localization using Highly Accurate Maps," In *Proc. of IV*, 2013, pp. 449–454.

[12] B. Soheilian, X. Qu, M. Brédif, "Landmark based Localization: LBA Refinement using MCMC-Optimized Projections of RJMCMC-Extracted Road Marks," In *Proc. of IV*, 2016, pp. 940–947.

[13] G. P. Gwon, W. S. Hur, S. W. Kim, and S. W. Seo, "Generation of a Precise and Efficient Lane-Level Road Map for Intelligent Vehicle Systems," *IEEE Trans. on VT*, vol. 66, no. 6, pp. 4517–4533, 2017.

[14] J. Wiest, H. Deusch, D. Nuss, S. Reuter, M. Fritzsche, and K. Dietmayer, "Localization based on Region Descriptors in Grid Maps," In *Proc. of IV*, 2014, pp. 793–799.

[15] J. H. Im, S. H. Im, and G. I. Jee, "Vertical Corner Feature based Precise Vehicle Localization using 3D LIDAR in Urban Area," *IEEE Sensors Journal*, vol. 16, no. 8, article no. 1268, 2016.

[16] J. Levinson, M. Michael, and T. Sebastian, "Map-Based Precision Vehicle Localization in Urban Environments," In *Proc. of RSS*, 2007.

[17] L. Li, M. Yang, C. Wang, and B. Wang, "Road DNA based Localization for Autonomous Vehicles," In *Proc. of IV*, 2016, pp. 883–888.

[18] C. Guo, K. Kidono, J. Meguro, Y. Kojima, M. Ogawa, and T. Naito, "A Low-Cost Solution for Automatic Lane-Level Map Generation using Conventional In-Car Sensors," *IEEE Trans. on ITS*, vol. 17, no. 8, pp. 2355–2366, 2016.

[19] J. Meguro, H. Ishida, C. Guo, K. Yamaguchi, and Y. Kojima, "Automatic Orthographic Road Image Generation Using in Vehicle Sensors for ADAS," *IJAE*, vol. 8, no. AVEC14, pp. 30–36, 2017.

[20] P. J. Besl and N. D. McKay, "A Method for Registration of 3-D Shapes," *IEEE Trans. on PAMI*, vol. 14, no. 2, pp. 239–256, 1992.

[21] S. Rusinkiewicz and M. Levoy, "Efficient Variants of the ICP Algorithm," In *Proc. of 3DIM*, 2001, pp. 145–152.

[22] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," In *Proc. of RSS*, 2009.

[23] B. Jian and B. C. Vemuri, "A Robust Algorithm for Point Set Registration using Mixture of Gaussians," In *Proc. of ICCV*, 2005, pp. 1246–1251.

[24] J. Ryde and H. Hu, "3D Mapping with Multi-Resolution Occupied Voxel Lists," *Autonomous Robots*, vol. 28, no. 2, pp. 169–185, 2010.

[25] A. Hornung, K. M. Wurm, M., Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An Efficient Probabilistic 3D Mapping Framework based on Octrees," *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013.

[26] M. Magnusson, A. Lilienthal, and T. Duckett, "Scan Registration for Autonomous Mining Vehicles using 3D-NDT," *Journal of Field Robotics*, vol. 24, no. 10, pp. 803–827, 2007.

[27] T. Stoyanov, M. Martin, and A. J. Lilienthal, "Point Set Registration Through Minimization of the L2 Distance Between 3D-NDT Models," In *Proc. of ICRA*, 2012, pp.5196–5201.

[28] R. W. Wolcott and R. M. Eustice, "Fast LIDAR Localization using Multiresolution Gaussian Mixture Maps," In *Proc. of ICRA*, 2015, pp. 2814–2821.

[29] R. W. Wolcott and R. M. Eustice, "Robust LIDAR Localization using Multiresolution Gaussian Mixture Maps for Autonomous Driving," *IJRR*, vol. 36, no. 3, pp. 292–319, 2017.

[30] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite," In *Proc. of CVPR*, 2012, pp. 3354–3361.

[31] T. Stoyanov, M. Magnusson, H. Almqvist, and A. J. Lilienthal, "On the Accuracy of the 3D Normal Distributions Transform as a Tool for Spatial Representation," In *Proc. of ICRA*, 2011, pp. 4080–4085.

[32] P. E. Forssén and D. G. Lowe, "Shape Descriptors for Maximally Stable Extremal Regions," In *Proc. of ICCV*, 2007, pp. 1–8.