2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Real-Time Graph-Based SLAM with Occupancy Normal Distributions Transforms

Cornelia Schulz and Andreas Zell

***Abstract*— Simultaneous Localization and Mapping (SLAM) is one of the basic problems in mobile robotics. While most approaches are based on occupancy grid maps, Normal Distributions Transforms (NDT) and mixtures like Occupancy Normal Distribution Transforms (ONDT) have been shown to represent sensor measurements more accurately. In this work, we slightly re-formulate the (O)NDT matching function such that it becomes a least squares problem that can be solved with various robust numerical and analytical non-linear optimizers. Further, we propose a novel global (O)NDT scan matcher for loop closure. In our evaluation, our NDT and ONDT methods are able to outperform the occupancy grid map based ones we adopted from Google's Cartographer implementation.**

I. Introduction

The problem of *Simultaneous Localization and Mapping* (SLAM) is a chicken-and-egg question. A correct global pose is required to build a consistent map, and a correct map is essential for precise self-localization. This dependent problem is typically solved via scan matching. To efficiently incorporate *loop closure* information, when a robot re-enters a previously visited location, *graph-based* methods can be applied. Thereby, nodes represent robot poses and edges describe constraints between these poses and are constructed both from frame-to-frame matching and loop closure [1].

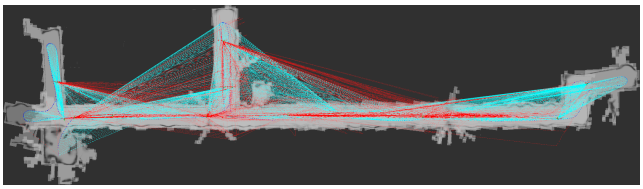
Nowadays, SLAM is mostly considered solved, especially regarding the 2D indoor case. However, depending on the robot setup and the utilized sensor, it still remains difficult.

The most promiment SLAM system to date is Google's *Cartographer* [1], which is based on occupancy grid maps and supports both 2D and 3D space. While it is without doubt a very sophisticated framework, we still observed difficulties when applying it to data from our own robots.

Besides occupancy grid maps, *Normal Distribution Transforms* (NDT) [2] and combinations of NDT with occupancy information, for instance NDT-OM [3], its enhanced version NDT-OMFG [4], and ONDT [5], have been introduced. Since other work's results [6], [7] indicate that NDT-based matching algorithms can be faster and provide better results than *Iterative Closest Point* (ICP), they might be superior to occupancy grid map based solutions for SLAM.

However, although such map types have been utilized multiple times in a SLAM context [2], [8], [9], [10], most approaches use inefficient hand-crafted matchers and no quantitative comparison with state-of-the-art occupancy grid map based SLAM methods has been done.

C. Schulz and A. Zell are with the Cognitive Systems Group at the Computer Science Department, University of Tübingen, Sand 1, 72076 Tübingen, Germany.
Contact: cornelia.schulz@uni-tuebingen.de



(a)

(b)

(c)

Fig. 1. (a) Optimized pose graph and ONDT submaps of a dataset generated with a TurtleBot2. Odometry nodes are rendered in blue, local matching constraints in cyan and loop closure constraints in red. (b) Our TurtleBot2 robots and (c) one of our Summit XL outdoor robots.

In this work, we investigate the applicability of (O)NDT in comparison to occupancy grid maps in the context of graph-based SLAM in order to address these issues.

The contributions of this work are the following:

- Inspired by Google's Cartographer, we modify the NDT matching function from [2] to become more robust in featureless environments and extend it to ONDT. In contrast to occupancy grid maps, no interpolation between map cells is required, since NDT-based maps are spatially continuous, hence the resulting matcher is faster than its occupancy grid map based counterpart.
- We solve the matching function in a derivation-free manner using fast open source solvers. In contrast to hand-crafted Jacobian- and Hessian-based methods, our approach is easy to implement and very fast.
- We propose a global scan matcher for loop closure with (O)NDT maps, similar to the correlative scan matcher [11] used in Google's Cartographer. However, our version does not require a stack of precomputed grids and hence it is more memory efficient.
- Our experiments show that our ONDT and NDT versions both outperform the occupancy grid map based methods on Rawseeds [12], [13] datasets and on two datasets recorded with our own robots.

For a fair comparison of the different methods, we implemented the different methods in a common SLAM framework that is indifferent to the utilized map type [14].

978-1-7281-7395-5/20/$31.00 ©2020 IEEE

II. Related Work

While occupancy grid maps only consist of occupancy probabilities, NDT maps are originally also designed as a regular grid, but instead of an occupancy value, the cells encode the measurement points falling into them as a multivariate sample distribution [2]. Consequently, a coarser grid resolution is sufficient. NDT-OM(FG) [3], [4] and ONDT [5] are combinations of these two possibilities and the involved grid cells contain both information types.

Both pure NDT-based and ONDT maps have already been used for Monte Carlo Localization (MCL) [15], [5]. Thereby, the probability density function of the sample distributions was stretched and then used directly as a likelihood-field-like sensor model to weight the pose hypotheses.

Scan matching was already discussed in the original publication of NDT by Biber and Strasser [2]. The authors maximized a correlation score between scan and map by applying Newton's algorithm with manually derived Hessians and Jacobians. Stoyanov et al. [16] applied the same method in 3D and further introduced distribution-to-distribution matching, which can either be used for scan-to-map matching by converting the scan to a local map, as done by Stoyanov et al. [9], or for map-to-map matching in a SLAM context. The latter was done by Einhorn et al. [8], but instead of performing frame-to-frame matching, wheel odometry was used to accumulate multiple scans into local submaps.

Ulas and Temeltas [17] on the other hand used multiple layers of NDT maps with different resolutions for scan matching via iterative refinement. The downside of this method is that, besides the additional memory requirements, inserting points into multiple maps also costs further runtime. The authors also used Jacobian and Hessian based solvers.

In principle however, this is similar to the global correlative scan matching method [11] that is used in Google's Cartographer [1]. In contrast to [17], it is a brute-force-like method and operates over a larger search window to avoid the problem of local optima.

In a graph-based SLAM context, Jelínek [10] combined the global correlative scan matching method [11], applied to NDT-based maps, with the local distribution-to-distribution matching method from [9]. However, the global matching step was performed with only two layers of NDT maps, and no quantitative evaluation was done.

In general, none of the presented works on NDT SLAM include a comparison with occupancy grid map based methods. Merely some NDT-based matchers [6], [7] were compared with ICP, a standard method for frame-to-frame matching that is not necessarily used in state-of-the-art SLAM implementations anymore. Google's Cartographer [1], for instance, uses a correlation maximization method similar to NDT matchers for frame-to-frame matching.

In this work, we use a common SLAM framework supporting different map types [14] to compare (O)NDT- and occupancy grid map based versions. Similar to Cartographer, we apply derivation-free solvers from Ceres [18] for local matching, and we use a slightly adapted version of the correlative scan matcher [11] for global matching.

III. Preliminaries

For clarity, we briefly summarize the basics about our ONDT and NDT maps [5] as well as the original NDT matching function and solution [2]. Further, we shortly introduce the basic SLAM structure, the utilized components and the relevant parameters [14].

A. Occupancy Normal Distribution Transforms (ONDT)

As proposed in the original publication by Biber and Strasser [2], our (O)NDT maps consist of multiple submaps of the same resolution, shifted by half the resolution along the different axes. This results in an effective grid with effective resolution r being half the resolution of the submaps and is required to reduce discretization problems, see [5].

In each submap, the likelihood of measuring a point q is derived from the non-normalized probability density function of sample distribution $\mathcal{N}_j(\mu_j, \Sigma_j)$ of cell j, multiplied by the occupancy probability $p_{j,\text{occ}}$ of cell j, where j is the cell q falls into. The complete likelihood for q is obtained by building the mean over all n_s submaps:

$$p(q) = \frac{1}{n_s} \cdot \sum_{j \in \{j_1,\dots,j_{n_s}\}} p_{j,\text{occ}} \cdot \exp\left[-\frac{(q-\mu_j)^t \Sigma_j^{-1} (q-\mu_j)}{2}\right]. \quad (1)$$

If NDT maps are used instead of ONDT, the probabilities $p_{j,\text{occ}}$ do not exist and can be set to 1 in this equation. In the 2D case it is $n_s = 4$, in the 3D case it is $n_s = 8$.

B. Original Approach to NDT Matching

The original approach of Biber and Strasser [2] to scan matching with NDT maps was to maximize the score of the desired transformation $\mathcal{T}$, which is the sum of the likelihood values of all scan points q, transformed with $\mathcal{T}$:

$$\text{score}(\mathcal{T}) := \sum_q p(\mathcal{T}(q)) \quad (2)$$

In practice, Newton's algorithm was used to iteratively update transformation $\mathcal{T}$ by adding a $\Delta\mathcal{T}$ that minimizes the function $f := -\text{score}(\mathcal{T})$. $\Delta\mathcal{T}$ is obtained by solving

$$\mathbf{H} \cdot \Delta\mathcal{T} = -\mathbf{g}, \quad (3)$$

with $\mathbf{H}$ and $\mathbf{g}$ being the Hessian and the gradient of f. For $\mathbf{g}$, the Jacobian needs to be derived.

C. Graph-Based SLAM Framework

The graph-based SLAM framework we use was developed by Kuhlmann [14] and is closely related to and inspired by Google's Cartographer [1]. As Cartographer, it builds several submaps with configurable split policy, for instance starting a new submap after each number n_{pc} of inserted data chunks.

A subset of scan poses is inserted into the pose graph, for instance after each Δt, Δd or $\Delta\theta$ (time, translational or rotational difference). From *local* and *global matching*, local and global constraints are generated between the matched node and the node corresponding to the start of the map.

After all n_n nodes, these constraints are used to optimize the global pose graph via *Sparse Pose Adjustment* (SPA) [19]. The main advantage over Cartographer is that all components are build as plugins and are therefore exchangeable, and that any map type can be utilized.

IV. Local Matcher

The briefly discussed NDT matching approaches are all based on the Jacobian and Hessian of a non-linear function. Apparently, the involved Newton or Levenberg-Marquardt algorithms are implemented by hand. Re-implementations of these methods did not yield promising results and were mostly slow. In our tests, we achieved the best results with derivation-free solvers and the following cost functions.

A. (O)NDT Cost Function

In principle, we want to maximize the score defined in (2). However, since most solver implementations minimize least squares problems, simply using the negative score as done by Biber and Strasser [2] is not sufficient, because the squared negative residuals would effectively minimize the score.

Equal to the occupancy grid map cost function used in Cartographer [1] we therefore minimize the least squares problem over residuals $(1 - p(\mathcal{T}(q_i)))$. To get a score independent of the number n_q of measurement points q_i involved, we formulate our map cost function as

$$C_{\text{map}}(\mathcal{T}) = \sum_{i=1}^{n_q} \left(\frac{1 - p(\mathcal{T}(q_i))}{n_q} \right)^2. \tag{4}$$

Unfortunately, this is not sufficient for successful matching in featureless environments, for instance if only straight walls are observable, since the driven distance would be underestimated, as illustrated and discussed in Fig. 2.

B. Translational and Rotational Cost Functions

In the Cartographer implementation, this problem is tackled by two modifications. Firstly, unallocated grid cells are treated like cells with a low occupancy probability of 0.1 to enforce the allocation of new cells by provoquing non-zero correlation scores for scan points with unallocated cells, which is required if the robot is moving forward into previously unvisited areas. For (O)NDT maps, this is not useful, since the higher grid cell size and the overlapping area causes a squared area with side length of about the resolution r around the distributions being allocated implicitly as free.

Secondly, the Cartographer implementation uses additional translational and rotational cost functions to penalize deviations of the final solution $\mathcal{T}$ from an initial guess that is usually obtained by extrapolation based on wheel odometry.

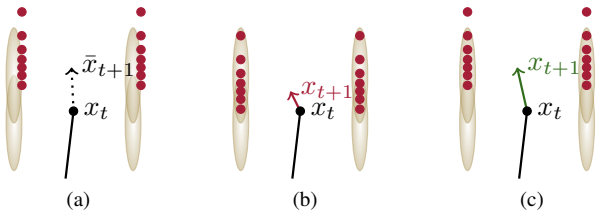


Fig. 2. Motivation of (7), using an exemplary featureless environment like a hallway with straight walls on both sides, illustrated as yellow-ish overlapping distributions according to the (O)NDT data structure. Scan points are shown in red. (a) Extrapolating from the last pose x_t with wheel odometry data yields an initial guess $\bar{x}_{t+1}$ for the next pose. (b) Simple scan matching without penalizing deviations from $\bar{x}_{t+1}$ would align the scan with the map distributions and the travelled distance is underestimated. (c) With (7), the scan is mostly aligned with the distributions, but x_{t+1} is closer to $\bar{x}_{t+1}$ and new map sectors are more likely being allocated.

This principle and the resulting effects on the matching result are illustrated in Fig. 2. On the other hand, this means that in featureless environments, our matchers (and probably also the occupancy grid map version) will fail if no odometry or other source for a good initial guess is available.

The according cost functions are implemented as in the Cartographer implementations as

$$C_{\text{trans}}(\mathcal{T}) = \sum_i \left(\text{trans}_i(\mathcal{T})\right)^2 = \|\text{trans}(\mathcal{T})\|_2^2, \tag{5}$$

with trans_i being a function extracting the i-th translational component of a transformation, i.e. x, y, z, and

$$C_{\text{rot}}(\mathcal{T}) = \sum_i \left(\text{rot}_i(\mathcal{T})\right)^2 = \|\text{rot}(\mathcal{T})\|_2^2, \tag{6}$$

with rot_i being a function extracting the i-th rotational component of a transformation, i.e. φ, θ, ψ (roll,pitch,yaw) or in case of quaternions the components x, y, z, w.

Depending on the utilized representation of the rotation, the solution space needs to be configured accordingly. Using Ceres [18], for instance the provided *QuaternionParameterization* can be applied. It projects a point back to the valid solution space. If φ, θ, ψ are used, this would for instance mean to restrict the angles to the interval $[-\pi, \pi)$.

The total cost function is then the weighted sum of the individual cost functions:

$$\begin{aligned} C(\mathcal{T}) = w_{\text{map}} &\cdot C_{\text{map}}(\mathcal{T}) + \\ w_{\text{trans}} &\cdot C_{\text{trans}}(\mathcal{T}) + \\ w_{\text{rot}} &\cdot C_{\text{rot}}(\mathcal{T}) \end{aligned} \tag{7}$$

Note that the transformation $\mathcal{T}$ is defined as the correction applied to the initial guess and that it is therefore initialized as identity function.

C. Solvers

To find $\mathcal{T}$, any derivation-free non-linear solver can be applied to solve the least squares minimization problem

$$\mathcal{T} = \arg\min_{\mathcal{T}'} C(\mathcal{T}'). \tag{8}$$

This can be solved either numerically, for instance with the Levenberg-Marquardt approach available in *ALGLIB* [20] or different methods provided by *NLopt* [21], from which we found *NEWUOA* [22] best, or analytically using automatic derivation and appropriate solvers, e.g. with *Ceres* [18].

In our tests, the Ceres version provided the best results at reasonable runtimes. Since this is also the solver used in Cartographer, this is the one we use for evaluation. The main difference between our matcher and the Cartographer matcher for occupancy grid maps is, that the occupancy grid map cells need to be interpolated and the (O)NDT cells not, since they are per definition piecewise spatially continuous.

Cartographer solves this via bicubic interpolation [1], which requires in the 2D case 16 map accesses. The fitted cubic hermite splines are then differentiated automatically, which yields the required Jacobian. With our (O)NDT maps, the correlation functions (1) can be differentiated directly, which is presumably much faster, depending on the costs of a single map access and the interpolation step.

More information about automatic derivatives and their efficiency can be found on the Ceres web page [18].

V. Global Matcher

The global scan matcher we use for loop closure is similar to the global correlative scan matcher [11] implemented in Cartographer [1], but it does not require the precomputation of a stack of grid maps with different resolutions.

A. Loop Closure Candidates

In our framework, poses according to newly inserted graph nodes are globally only matched against maps they were not matched locally with. The selection of loop closure candidates is not necessarily restricted further. However, it is for instance useful to only match nodes within a maximum distance of $d_{\max}$ and only a small ratio ρ of all graph nodes.

B. Brute Force Approach

As discussed in [1], Cartographer's global correlative scan matcher in principle performs an exhaustive brute force search in a configurable search window with linear window size of s_d and angular window size of s_θ. For (O)NDT maps, the correlation function to be maximized is (2), normalized with the number of points n_q, i.e.

$$\frac{1}{n_q} \cdot \sum_q p\left(\mathcal{T}\left(q\right)\right). \tag{9}$$

The candidate solutions are generated according to this window and linear and angular step sizes δ_d and δ_θ, respectively. As in Cartographer, δ_θ is estimated from the scan to match such that points with maximum distance to the sensor origin differ by at most δ_d between two consecutive steps [1]. However, instead of setting δ_d to r, we use δ_d as a parameter.

C. Depth First Search (DFS)

To reduce the exhaustive runtime, we iteratively refine our search candidates in k steps, similar to DFS. First, we sample initial candidate transformations τ with a linear step size of $2^{k-1} \cdot \delta_d$. The angular step size δ_θ is kept constant over all k steps and the according rotated measurement points can be precomputed as in the Cartographer implementation, see [1].

These candidate transformations are then scored based on (9) and only transformations with a score f_τ not smaller than a minimum score of $f_{g,\min}$ are investigated further. Around these τ, new candidates are spawned according to the linear step size of the next iteration. Finally, the best τ is taken as matching result. In pseudo code, this is illustrated in Alg. 1.

Algorithm 1 Global 2D (O)NDT Matcher

Input: initial guess $\mathcal{T}_0$, measurement points $\{q\}$, map
Output: either matching result $\mathcal{T}$ and score, or **NULL**

$\delta_{d,k} \leftarrow 2^{k-1} \cdot \delta_d$ ▷ initial linear step size
$\mathcal{X} \leftarrow \{\langle \tau, f_\tau \rangle\}$
  ▷ initial candidates τ around $\mathcal{T}_0$ according to $\delta_{d,k}$ and δ_θ
  ▷ scores f_τ of these candidates according to (9)
for each $\chi = \langle \tau_\chi, f_{\tau_\chi} \rangle \in \mathcal{X}$ **with** $f_{\tau_\chi} \geq f_{g,\min}$ **do**
  $\mathcal{X} \leftarrow \mathcal{X} \cup$ recursively sampled $\langle \tau_{\chi+\Delta_i}, f_{\tau_{\chi+\Delta_i}} \rangle$
    ▷ recursively sample around τ according to $\delta_{d,k-1}, \ldots, \delta_{d,1}$
    ▷ stop recursing as soon as a score $< f_{g,\min}$ occurs
end for
$\chi_{\text{best}} \leftarrow \arg\max_{f_{\tau_\chi}} \left(\chi = \langle \tau_\chi, f_{\tau_\chi} \rangle \in \mathcal{X}, f_{\tau_{\chi_{\text{best}}}} \geq f_{g,\min}\right)$
return χ_{best} **or NULL** ▷ no result if $f_{\tau_{\chi_{\text{best}}}} < f_{g,\min}$

In Cartographer, this is done similarly, but the score of the candidates is estimated based on a stack of k grid maps with different resolutions. The initial candidates are scored using a grid map with a resolution of $2^k \cdot r$ and the last ones with the original grid map. These maps are precomputed and each cell is set to the maximum of the corresponding cells of the map with the next finer resolution to implement a real DFS.

However, this can not directly be applied to (O)NDT, since the map cells can not be combined in such manner, that the maximum score of all possible subsampled transformations could be estimated by simple lookup. Consequently, we use the original map in each step, which could result in overlooking local optima, depending on the sampling resolution δ_d.

VI. Evaluation

In the following, we present the experiments we conducted to compare our (O)NDT matchers to the according occupancy grid map based versions, which are re-implementations functionally equivalent to the Cartograher methods.

A. Datasets and Metrics

To investigate a good mixture of different robotic platforms, we utilize datasets *25a* and *27a* of the *Rawseeds Bicocca* dataset compilation [12], [13], and also two self-made datasets, recorded with a TurtleBot2 and a Summit XL, respectively. The ground truth trajectories of our robots were optimized by iteratively applying the Cartographer offline tool with extreme parameters and are shown in Fig. 3.

The robot utilized for the Rawseeds datasets was equipped with two SICK laser scanners with $80\,\mathrm{m}$ range, our TurtleBot2 merely with a Hokuyo URG-04LX-UG01 laser scanner with a very short range of about $5.6\,\mathrm{m}$, and our Summit XL with a SICK TiM561 laser scanner with about $10\,\mathrm{m}$ range. The other sensors are irrelevant for this evaluation.

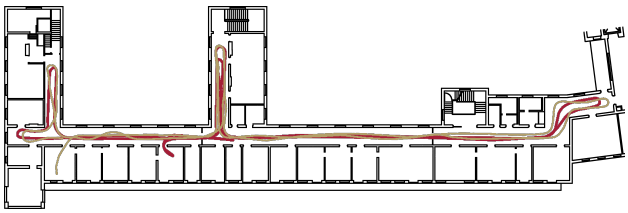


Fig. 3. Ground truth trajectories of the TurtleBot2 (yellow-ish) and the Summit dataset (red), aligned coarsely with the floor plan of our department.

The TurtleBot2 dataset is challenging, because the low range of the utilized Hokuyo laser scanner results in mostly featureless scans, similar to the problem shown in Fig. 2. The Summit XL dataset is also difficult to handle, because the robot moved very fast, especially during rotations, and therefore, the translational and especially rotational distance between pairs of consecutive scans is relative large.

For evaluation, we measured the mean runtimes of the different components: insertion of a scan or point cloud into a map, local and global matching. The framework overhead should be approximately the same regardless of the utilized map type and is therefore ignored. Further, we estimated the achieved localization accuracy in terms of *absolute trajectory error* (ATE) [23] and *relative pose error* (RPE) [24], metrics provided by the Rawseeds benchmarking suite [12], [13].

B. Parameters

As expected, our SLAM system is very sensitive to parameter changes. Therefore, we performed an exhaustive manual parameter search in order to find good common parameters for all map types, such that each of the final results (with default value of k) was close to the best one achieved with the according map type. Since the map types behave differently, we optimized the respective values of $f_{\{l,g\},\min}$ separately.

For a better understanding, all relevant parameters are listed in Tab. I with default values for occupancy grid maps ('Occ') and (O)NDT maps. The indicator '–' means that the associated feature was by default not used. Deviations of the default parameters will be listed in the results table.

Further, we optionally applied voxel filters with resolutions of v_l and v_g before local and global matching, respectively, in order to reduce the data size and therefore the runtime.

TABLE I
PARAMETERS

	Name	Description	Default Value Occ	(O)NDT
Mapping	r	(effective) map resolution	5 cm	50 cm
	n_{pc}	num. inserted data per submap	500	
Pose Graph	Δt	min. time difference (between 2 nodes)	2.5 s	
	Δd	min. translational difference	0.25 m	
	$\Delta\theta$	min. rotational difference	0.2 rad	
Global Optim.	n_n	num. of nodes in between 2 SPA steps	100	
	$d_{\max}$	max. distance between matched nodes	50 m	
	ρ	ratio of matched graph nodes	0.3	
Local Matching	v_l	voxel filter resolution	–	
	w_{map}	weight of map cost function	2	
	w_{trans}	weight of translational cost function	18	2
	w_{rot}	weight of rotational cost function	9	1
	$f_{l,\min}$	local minimum score	–	–
Global Matching	v_g	voxel filter resolution	–	
	s_d	linear search window size	7.5 m	
	s_θ	angular search window size	45°	
	δ_d	linear sampling resolution	r	
	k	stack size or iteration count	5	1
	$f_{g,\min}$	global minimum score	–	–

Empirically, we noticed that the map score of a successfully matched scan is for occupancy grid maps about two to three times as high as for (O)NDT maps. Therefore, we increased the weights w_{trans} and w_{rot} by factor $3^2 = 9$ for occupancy grid maps in order to obtain a comparable ratio between the different cost functions.

We repeated each of the experiments with default values for k and δ_d at least three times and made sure that the resulting runtimes and accuracy values did not significantly vary (i.e. that they stayed in $\mu \pm \sigma$). The results with the Bicocca datasets, however, were not always reproducible.

C. Results

The results of the best individual runs with NDT, ONDT and occupancy grid map based approaches can be found in Tab. II. Thereby, we evaluated three different global matching configurations with different $k = \{1, 3, 5\}$. Depending on k, we used values $\delta_d = \{50\,\text{cm}, 15\,\text{cm}, 5\,\text{cm}\}$ for the (O)NDT variants to reduce the runtime for small k such that most runs were concluded in real-time, unless marked with a '*'.

Additionally to the overall results, we further investigated the development of the ATE over time. It is plotted for the best run per map type for the different datasets in Fig. 4. In these plots, a low local slope, i.e. a flat curve, indicates that the local matcher is inducing only a small drift.

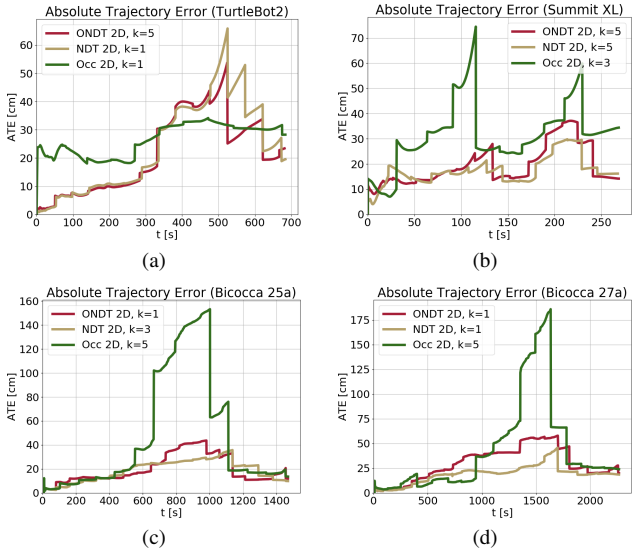


Fig. 4. Comparison of the best runs with the different map types (ONDT in red, NDT in yellow and Occ in green) regarding ATE over time on the 4 datasets: (a) TurtleBot2, (b) Summit XL, (c) Bicocca 25a, (d) Bicocca 27a.

D. Discussion

Because of the frequent direction changes in our self-made datasets, it was advantageous to reduce n_{pc}, especially for the dataset recorded with the very fast Summit XL. For the same reason, we set w_{trans} to zero, because the Summit XL's wheel odometry introduced additional noise and because it almost always slightly underestimated the real driven distance.

For the Summit XL dataset, it also helped to deactivate the local voxel filter to get a more precise local scan matching result. This was, on the other hand, contra-productive for the TurtleBot2 dataset, presumably due to the high sensor noise of the Hokuyo laser scanner, so we applied $v_l = 10\,\text{cm}$.

With the Bicocca datasets, we experienced high runtimes due to a high number of submaps and therefore global scan matching operations, and due to the very high laser scanner range that resulted in high amounts of data points and large submaps. Therefore, we had to adapt the parameters of pose graph generation, global optimization and global matching to reduce amount and runtime of the global matching steps.

In general, we observed empirically that our local (O)NDT scan matchers were less sensitive to changes of the local voxel filter resolution v_l, and that they also worked with high values of v_l. Furthermore, as can be derived from the locally mostly less steep curves in Fig. 4 and the lower RPE values for most datasets, they produced less local drift than their occupancy grid map based counterparts.

The vertical jumps in Fig. 4 result from global optimization steps. Thereby, jumps resulting in an increased ATE were most likely caused by false positive loop closure constraints. This especially occured for the occupancy grid map based variants in the Summit XL and Bicocca datasets.

TABLE II

PERFORMANCE COMPARISON OF THE DIFFERENT 2D MAP AND MATCHER TYPES ON DIFFERENT DATASETS
(SYSTEM SPECIFICATIONS: INTEL XEON E3-1270 V3 CPU @ 3.5 GHZ WITH 16 GB OF RAM)

Dataset	Parameters			Mean Runtimes [ms]			Localization Accuracy		
	Map Type	k	others	Insertion	Local M.	Global M.	ATE [cm]	T-RPE [cm]	R-RPE [°]
TurtleBot2	ONDT	1	$n_{pc} = 500$ $v_l = 10\,\text{cm}$ $v_g = 50\,\text{cm}$	$\mathbf{0.07 \pm 0.04}$	$\mathbf{1.03} \pm 0.43$	$\mathbf{3.83 \pm 2.30}$	24.83 ± 15.84	0.17 ± 0.44	0.51 ± 1.15
		3		$\mathbf{0.07} \pm 0.08$	1.12 ± 0.65	8.61 ± 6.71	25.25 ± 11.15	0.17 ± 0.44	0.51 ± 1.15
		5		$\mathbf{0.07 \pm 0.04}$	$\mathbf{1.03 \pm 0.33}$	11.57 ± 7.08	$\mathbf{23.42} \pm 12.06$	0.17 ± 0.44	0.51 ± 1.15
	NDT	1	only (O)NDT: $f_{l,\min} = -$ $f_{g,\min} = 0.1$	0.05 ± 0.04	1.03 ± 0.34	$\mathbf{3.78 \pm 2.25}$	$\mathbf{19.57 \pm 11.83}$	0.18 ± 0.60	0.52 ± 1.20
		3		0.05 ± 0.04	1.04 ± 0.34	7.53 ± 4.12	35.37 ± 16.54	0.18 ± 0.60	0.52 ± 1.20
		5		0.05 ± 0.04	$\mathbf{1.02 \pm 0.31}$	11.21 ± 6.25	32.27 ± 16.51	0.18 ± 0.60	0.52 ± 1.20
	Occ	1	only Occ: $f_{l,\min} = -$ $f_{g,\min} = 0.85$	0.27 ± 1.09	0.92 ± 0.47	266.19 ± 141.15	$\mathbf{28.29 \pm 12.75}$	0.38 ± 0.72	0.54 ± 1.13
		3		$\mathbf{0.26 \pm 1.07}$	$\mathbf{0.86 \pm 0.26}$	16.53 ± 8.86	28.51 ± 12.54	0.38 ± 0.72	0.54 ± 1.13
		5		$\mathbf{0.26} \pm 1.08$	$\mathbf{0.86 \pm 0.26}$	$\mathbf{1.53 \pm 0.75}$	30.49 ± 15.28	0.38 ± 0.72	0.54 ± 1.13
Summit XL	ONDT	1	$n_{pc} = 250$ $w_{\text{trans}} = 0$ $v_g = 50\,\text{cm}$	$\mathbf{0.10} \pm 0.12$	$\mathbf{3.77 \pm 0.95}$	$\mathbf{26.38 \pm 14.53}$	24.46 ± 11.70	1.01 ± 3.53	1.27 ± 2.63
		3		$\mathbf{0.10 \pm 0.10}$	3.97 ± 1.60	63.36 ± 36.16	17.47 ± 9.60	1.01 ± 3.53	1.27 ± 2.63
		5		$\mathbf{0.10} \pm 0.16$	4.15 ± 2.04	98.69 ± 56.13	$\mathbf{14.17 \pm 8.22}$	1.01 ± 3.53	1.27 ± 2.63
	NDT	1	only (O)NDT: $f_{l,\min} = 0.4$ $f_{g,\min} = 0.2$	$\mathbf{0.07} \pm 0.09$	$\mathbf{3.76 \pm 1.10}$	$\mathbf{24.81 \pm 13.77}$	28.79 ± 13.73	0.98 ± 2.56	1.29 ± 2.66
		3		$\mathbf{0.07} \pm 0.13$	3.91 ± 1.69	60.06 ± 40.84	24.96 ± 12.52	0.98 ± 2.56	1.29 ± 2.66
		5		$\mathbf{0.07 \pm 0.08}$	4.17 ± 2.39	95.60 ± 59.83	$\mathbf{16.17 \pm 9.37}$	0.98 ± 2.56	1.29 ± 2.66
	Occ	1	only Occ: $f_{l,\min} = 0.75$ $f_{g,\min} = 0.6$	0.92 ± 4.21	11.92 ± 20.24	2837.16 ± 1120.97	125.44 ± 88.11	0.47 ± 1.10	1.15 ± 2.42 *
		3		0.47 ± 1.99	5.13 ± 3.89	152.94 ± 80.43	$\mathbf{34.41} \pm 15.65$	0.47 ± 1.10	1.15 ± 2.42
		5		$\mathbf{0.43 \pm 1.84}$	$\mathbf{4.46 \pm 2.50}$	$\mathbf{24.12 \pm 19.50}$	37.42 ± 15.28	0.47 ± 1.10	1.15 ± 2.42
Bicocca 25a	ONDT	1	$n_{pc} = 2500$ $\Delta t = 10\,\text{s}$ $\Delta d = 1\,\text{m}$ $\Delta\theta = 0.6\,\text{rad}$ $\rho = 0.4$ $v_l = 10\,\text{cm}$ $v_g = 1\,\text{m}$ $s_d = 7\,\text{m}$ $s_\theta = 30°$	$\mathbf{0.17 \pm 0.27}$	$\mathbf{1.27 \pm 0.95}$	$\mathbf{68.43 \pm 66.77}$	$\mathbf{11.87 \pm 5.31}$	$\mathbf{1.44 \pm 3.90}$	1.17 ± 2.28
		3		0.19 ± 0.45	1.44 ± 1.47	177.70 ± 188.99	15.27 ± 9.35	1.45 ± 4.23	1.17 ± 2.32 *
		5		0.20 ± 0.53	1.61 ± 2.07	360.11 ± 367.74	13.34 ± 7.06	1.46 ± 4.25	$\mathbf{1.14 \pm 2.25}$ *
	NDT	1		$\mathbf{0.05 \pm 0.13}$	$\mathbf{1.31 \pm 1.02}$	$\mathbf{55.36 \pm 49.76}$	11.45 ± 4.96	1.45 ± 4.12	$\mathbf{1.15} \pm 2.23$
		3		0.06 ± 0.22	1.45 ± 1.42	144.84 ± 139.55	$\mathbf{9.83} \pm 5.87$	$\mathbf{1.42 \pm 3.38}$	1.17 ± 2.26
		5		0.07 ± 0.29	1.59 ± 1.77	289.41 ± 269.05	41.08 ± 24.34	1.45 ± 4.16	$\mathbf{1.15 \pm 2.21}$ *
	Occ	1		0.56 ± 2.69	2.99 ± 3.25	6877.78 ± 4668.58	402.92 ± 280.92	1.48 ± 4.43	1.28 ± 2.35 *
		3		0.32 ± 1.70	1.64 ± 1.96	496.32 ± 377.45	20.38 ± 11.42	$\mathbf{1.56} \pm 4.42$	$\mathbf{1.26 \pm 2.26}$ *
		5	only ONDT: $f_{l,\min} = -$ $f_{g,\min} = 0.2$ only NDT: $f_{l,\min} = -$ $f_{g,\min} = 0.4$ only Occ: $f_{l,\min} = -$ $f_{g,\min} = 0.7$	$\mathbf{0.21 \pm 1.04}$	$\mathbf{1.10 \pm 0.73}$	$\mathbf{44.28 \pm 34.73}$	$\mathbf{13.59 \pm 8.38}$	1.58 ± 4.41	1.29 ± 2.41
Bicocca 27a	ONDT	1		$\mathbf{0.17 \pm 0.32}$	$\mathbf{1.32 \pm 1.08}$	$\mathbf{72.64 \pm 73.27}$	$\mathbf{19.83} \pm 14.23$	1.97 ± 6.48	1.50 ± 4.38
		3		0.19 ± 0.42	1.45 ± 1.42	172.21 ± 170.28	20.82 ± 13.30	2.02 ± 6.78	$\mathbf{1.48 \pm 4.05}$ *
		5		0.23 ± 0.62	1.72 ± 2.02	364.04 ± 346.11	20.46 ± 15.94	$\mathbf{1.94 \pm 6.31}$	1.49 ± 4.38 *
	NDT	1		$\mathbf{0.05 \pm 0.17}$	$\mathbf{1.33 \pm 1.15}$	$\mathbf{59.51 \pm 55.27}$	$\mathbf{18.52 \pm 11.74}$	1.98 ± 6.79	$\mathbf{1.44} \pm 4.01$
		3		0.06 ± 0.22	1.42 ± 1.42	140.82 ± 124.70	23.48 ± 12.81	$\mathbf{1.94 \pm 6.55}$	1.49 ± 4.57
		5		0.07 ± 0.33	1.73 ± 1.99	314.97 ± 275.45	30.65 ± 21.40	2.07 ± 6.91	1.48 ± 3.88 *
	Occ	1		0.58 ± 2.66	3.11 ± 3.35	6960.77 ± 4802.04	1035.76 ± 425.63	2.15 ± 6.73	$\mathbf{1.44} \pm 3.15$ *
		3		0.39 ± 2.06	2.02 ± 2.30	515.96 ± 378.65	67.94 ± 63.56	$\mathbf{1.99 \pm 6.46}$	1.52 ± 4.19 *
		5		$\mathbf{0.27 \pm 1.20}$	$\mathbf{1.46 \pm 1.20}$	$\mathbf{57.70 \pm 61.24}$	$\mathbf{23.93 \pm 17.87}$	2.11 ± 7.29	1.56 ± 4.48

experiments marked with * were not concluded in real-time (i.e. the last scan matching operation terminated significantly after the end of the datastream)

It indicates that the (O)NDT map score in general may be less ambiguous and therefore better suited for loop closure.

With (O)NDT maps, this effect only occured in the TurtleBot2 dataset. Presumably, this is caused by the featureless low-range scans in combination with the smoothing character of distribution-based maps. Nevertheless, during the first part the ATE is much lower than with occupancy grid maps.

Overall, the best results (gray rows in Tab. II) were mostly achieved with NDT maps, but the ONDT results were more robust against changes of k. On all datasets, the occupancy grid map based version was outperformed by both of them.

The significantly worst ATE results occured in runs with occupancy grid maps that were not concluded in real-time. In these runs, due to the immense runtime of global matching, most global matching constraints were inserted into the pose graph after all local ones. Since global optimization is only executed as long as poses are inserted into the pose graph, according to parameter n_n, the corresponding loop closure information had no impact on the optimized trajectory.

Interestingly, good results were already achieved with high δ_d values for global (O)NDT matching. Apparently, the local constraints compensate for inaccurate global constraints.

VII. CONCLUSION

In this work, we presented components for local and global scan matching in a SLAM context that are based on NDT and ONDT maps and which are simple and easy to implement. We experimentally evaluated various ONDT and NDT SLAM configurations compared to according occupancy grid map based ones, which are implemented using components functionally close to Google's Cartographer implementation.

On all investigated datasets, recorded with strongly differing robots, our (O)NDT based configurations were able to outperform their occupancy gridmap based counterparts both regarding localization accuracy and error over time. In general, we experienced that our NDT and ONDT based components were much more robust against different challenging influences and parameter changes.

REFERENCES

[1] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-Time Loop Closure in 2D LIDAR SLAM," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1271–1278.

[2] P. Biber and W. Strasser, "The Normal Distributions Transform: A New Approach to Laser Scan Matching," in *IEEE/RSJ International Conf. on Intelligent Robots and Systems (IROS)*, 2003, pp. 2743–2748.

[3] J. Saarinen, H. Andreasson, T. Stoyanov, J. Ala-Luhtala, and A. J. Lilienthal, "Normal Distributions Transform Occupancy Maps: Application to Large-Scale Online 3D Mapping," in *IEEE International Conf. on Robotics and Automation (ICRA)*, 2013, pp. 2233–2238.

[4] J. Saarinen, T. Stoyanov, H. Andreasson, and A. J. Lilienthal, "Fast 3D Mapping in Highly Dynamic Environments using Normal Distributions Transform Occupancy Maps," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 4694–4701.

[5] C. Schulz, R. Hanten, and A. Zell, "Efficient Map Representations for Multi-Dimensional Normal Distributions Transforms," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 2679–2686.

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

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

[8] E. Einhorn and H.-M. Gross, "Generic NDT mapping in dynamic environments and its application for lifelong SLAM," *Robotics and Autonomous Systems*, vol. 69, 2015, pp. 28–39.

[9] T. Stoyanov, J. Saarinen, H. Andreasson, and A. J. Lilienthal, "Normal Distributions Transform Occupancy Map Fusion: Simultaneous Mapping and Tracking in Large Scale Dynamic Environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 4702–4708.

[10] L. Jelínek, "Graph-based SLAM on Normal Distributions Transform Occupancy Map," Bachelor's Thesis, Charles University, Prague, 2016.

[11] E. B. Olson, "Real-Time Correlative Scan Matching," in *IEEE Intern. Conf. on Robotics and Automation (ICRA)*, 2009, pp. 4387–4393.

[12] A. Bonarini, W. Burgard, G. Fontana, M. Matteucci, D. G. Sorrenti, and J. D. Tardos, "RAWSEEDS: Robotics Advancement through Web-publishing of Sensorial and Elaborated Extensive Data Sets," in *Proc. of IROS'06 Workshop on Benchmarks in Robotics Research*, 2006.

[13] S. Ceriani, G. Fontana, A. Giusti, D. Marzorati, M. Matteucci, D. Migliore, D. Rizzi, D. G. Sorrenti, and P. Taddei, "Rawseeds Ground Truth Collection Systems for Indoor Self-localization and Mapping," *Autonomous Robots*, vol. 27, no. 4, 2009, pp. 353–371.

[14] P. Kuhlmann, "Multi-Modal SLAM for Outdoor Robots," Master's thesis, University of Tübingen, 2019.

[15] J. Saarinen, H. Andreasson, T. Stoyanov, and A. J. Lilienthal, "Normal Distributions Transform Monte-Carlo Localization (NDT-MCL)," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 382–389.

[16] 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," *International Journal of Robotics Research (IJRR)*, vol. 31, no. 12, 2012, pp. 1377–1393.

[17] C. Ulas and H. Temeltas, "A 3D Scan Matching Method Based On Multi-Layered Normal Distribution Transform," *IFAC Proceedings Volumes*, vol. 44, no. 1, 2011, pp. 11 602 – 11 607.

[18] S. Agarwal and K. M. et al., "Ceres Solver," http://ceres-solver.org.

[19] K. Konolige, G. Grisetti, R. Kümmerle, W. Burgard, B. Limketkai, and R. Vincent, "Efficient Sparse Pose Adjustment for 2D Mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010, pp. 22–29.

[20] S. Bochkanov, "ALGLIB," www.alglib.net.

[21] S. G. Johnson, "The NLopt nonlinear-optimization package," http://github.com/stevengj/nlopt.

[22] M. J. D. Powell, "The NEWUOA software for unconstrained optimization without derivatives," in *40th Workshop on Large Scale Nonlinear Optimization*. Springer, 2006, pp. 255–297.

[23] RAWSEEDS. Absolute Trajectory Error (ATE). http://www.rawseeds.org/rs/methods/view/9.

[24] ——. Relative Pose Error (RPE). http://www.rawseeds.org/rs/methods/view/11.