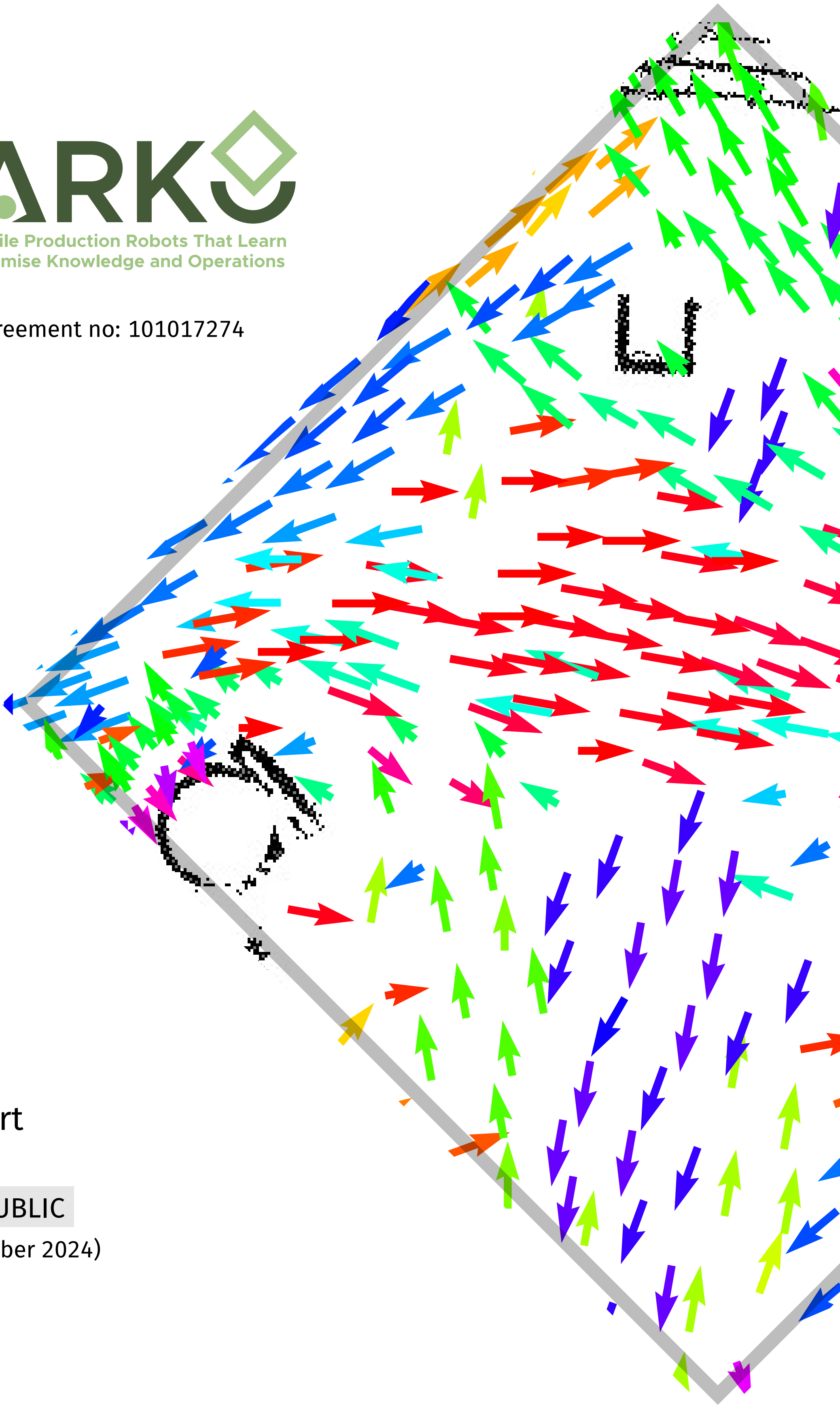
DARKO

Dynamic Agile Production Robots That Learn and Optimise Knowledge and Operations

H2020-ICT-2020-2 Grant agreement no: 101017274



DELIVERABLE 3.4
Final mapping report

Dissemination Level: PUBLIC

Due date: month 48 (December 2024)
Deliverable type: Report
Lead beneficiary: ORU

Contents

1 Introduction

This deliverable reports on the final version of the DARKO mapping system, including algorithms and software that have been developed, validated, and integrated on the DARKO robot platforms as of Month 48 (December 2024), as well as an outlook towards ongoing research in the final months of the project.

The overall objective of WP3 is to work towards *hands-off, failure-aware construction of rich map representations beyond mere geometry*. This goal maps to DARKO Objective 3 (efficient deployment) and Objective 2 (human-robot co-production).

The mapping-related methods reported in this deliverable comprise novel lidar-based and RGB/D-based methods for constructing and localizing in 3D navigation maps (T3.1), baseline software for merging lidar data with floor-plan line drawings (T3.2), creating and using several map of dynamics representations (T3.3), as well as methods for constructing "reliability-aware maps ", including reference-free map quality assessment and localization risk maps for anticipating localization inaccuracies (T3.4).

2 Efficient mapping (T3.1)

2.1 Baseline lidar mapping and localization

The baseline mapping and localization stack used by the DARKO robot uses a traditional graph-based simultaneous localisation and mapping (SLAM) method [1, 2] based on normal distributions transform (NDT) occupancy map (NDT-OM) sub-maps [3]. For localizing in the NDT-OM map graph, we use a graph-aware version of NDT Monte Carlo localisation (NDT-MCL) [4]. The output of the baseline mapping system is a 3D NDT-OM map (for localization), a 3D point cloud map (mainly for visualization), and a 2D grid map that can be easily integrated with the WP6 motion planners.

This software has been used for essential mapping and localization in the integrated system that has been demonstrated in Milestones 1–3. The example output can be seen in Figure 1.

2.2 Efficient Lidar Mapping

In contrast to the traditional *explicit* map representations described in Section 2.1, the past few years have seen an enormous interest in neural and *implicit* map representations, which allow for fully continuous representations. However, achieving accurate, detailed surface reconstruction at a low memory cost is difficult, especially for large-scale scenes.

As part of DARKO's WP3, we have devised a neural 3D surface reconstruction method called 3QFP [5]. We propose a sparse data structure called, *Tri-Quadtrees*, which represents the environment using learnable features stored in three planar quadtree projections. The learned features are then decoded into signed distance values through a small multi-layer perception. Compared to existing methods, we demonstrate that this approach facilitates smoother reconstruction with a higher completion ratio with fewer holes.

The 3QFP method learns a continuous signed distance function (SDF) representation of the environment, given lidar scans and known poses. Specifically, the world coordinate $\boldsymbol{p}_i \in \mathbb{R}^3$ is mapped into an SDF value $s_i \in \mathbb{R}$. As shown in Figure 2, our neural implicit representation is composed of two components: the learnable features stored in the quadtree nodes and a globally shared MLP to predict the SDF value. The features and the network parameters are learned during test time by using direct lidar measurement to supervise network predictions.

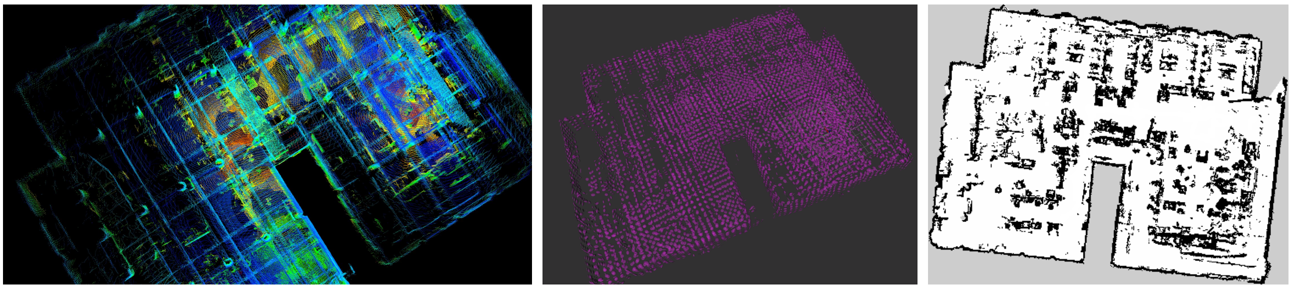


Figure 1: Geometric 2D and 3D maps from Milestone 3 at the Deutsches Museum in Munich, created with the DARKO prototype mapping system from T3.1. Left to right: 3D point cloud map (for visualization), 3D NDT-OM map (for localization), 2D occupancy grid (for motion planning).

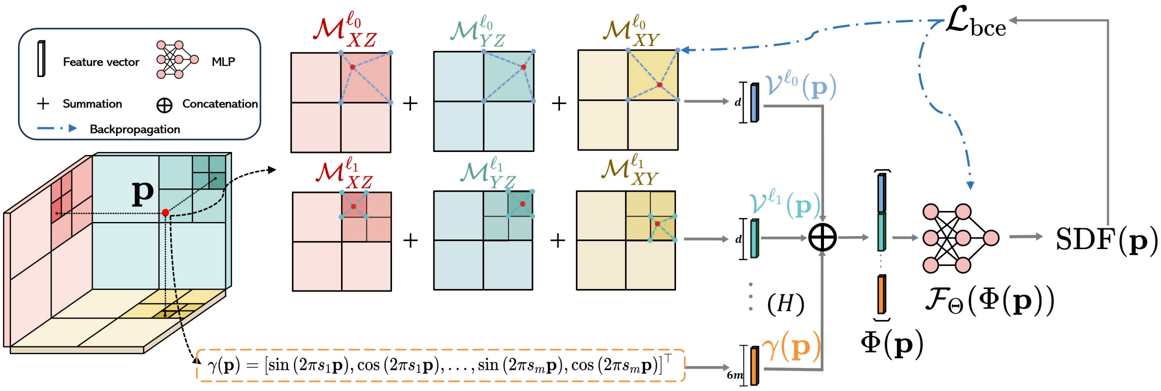

Feature vector
MLP
Summation
Concatenation
Backpropagation
$\mathcal{M}^{\ell_0}_{XZ}$
$\mathcal{M}^{\ell_0}_{YZ}$
$\mathcal{M}^{\ell_0}_{XY}$
$\mathcal{M}^{\ell_1}_{XZ}$
$\mathcal{M}^{\ell_1}_{YZ}$
$\mathcal{M}^{\ell_1}_{XY}$
$\mathbf{p}$
$\mathcal{V}^{\ell_0}(\mathbf{p})$
$\mathcal{V}^{\ell_1}(\mathbf{p})$
d
(H)
$\gamma(\mathbf{p})$
$6m$
$\gamma(\mathbf{p}) = [\sin(2\pi s_1 \mathbf{p}), \cos(2\pi s_1 \mathbf{p}), \ldots, \sin(2\pi s_m \mathbf{p}), \cos(2\pi s_m \mathbf{p})]^\top$
$\Phi(\mathbf{p})$
$\mathcal{F}_\Theta(\Phi(\mathbf{p}))$
$\mathrm{SDF}(\mathbf{p})$
$\mathcal{L}_{\mathrm{bce}}$

Figure 2: Overview of *3QFP* [5]. We represent the scene with three planar quadtrees $\mathscr{M}^{\ell}_{i}$, $i \in \{XZ, YZ, XY\}$, where ℓ represents the quadtree depth. We store features in the deepest H levels of quadtree resolution. When querying for a point $\boldsymbol{p}$, we project it onto planar quadtrees to identify the node containing $\boldsymbol{p}$ at level ℓ. The feature of $\boldsymbol{p}$ is then calculated by bilinear interpolation based on the queried location and vertex features. We add features at the same level and concatenate among different levels. Concatenated with the positional encoding $\gamma(\boldsymbol{p})$, $\boldsymbol{p}$'s feature ($\Phi(\boldsymbol{p})$) is fed into a small MLP ($\mathscr{F}_\Theta$) to predict the SDF value. The learnable features stored in the quadtree nodes and the network parameters are learned by test-time optimization using the loss function $\mathscr{L}_{\mathrm{bce}}$.

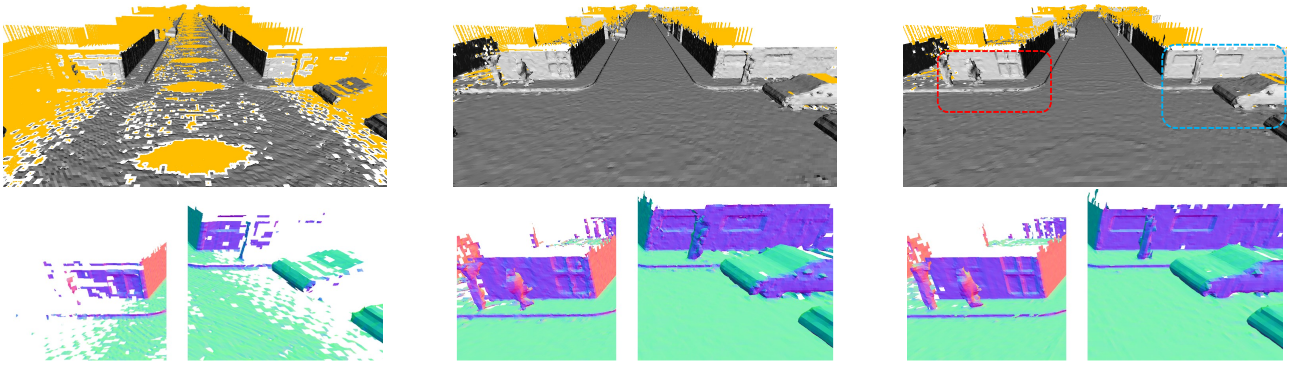

(a) *VDBFusion* (b) *SHINE-Mapping* (c) Ours

Figure 3: Qualitative visualization of the map quality on the MaiCity dataset using every 6th frame. The first row depicts the difference between the dense ground truth point cloud and the reconstructed mesh; the ground truth points with an error of more than 0.1 m are highlighted in orange. The second row shows zoomed-in images of the dashed areas (indicated in the top-right image). When inputs are sparse (e.g., every 6th frame in this case), our method obtains visibly smoother results.

As shown in the quantitative experiments in Table 1, our method is more memory efficient than the previous state-of-the-art implicit representation [6], while still achieving a higher completion ratio than explicit representation methods such as VDBFusion [7]. The qualitative results shown in Figure 3 demonstrate that our method is capable of producing smoother reconstructions and achieves good hole filling when inputs are sparse.

Table 1: Quantitative evaluation of the reconstruction quality on the MaiCity and NewerCollege datasets with *dense* inputs. We report the *Completion* (Comp.), *Accuracy* (Acc.), *Completion Ratio* (Comp.Ratio) and *Accuracy Ratio* (Acc.Ratio) with a threshold of 0.1 m for MaiCity and 0.2 m for NewerCollege. We also report the number of learnable parameters for neural implicit representation methods. Bold fonts represent the best results. Our method achieves a significantly higher completion ratio than *VDBFusion* with fewer parameters than *SHINE-Mapping*. (↓: lower better; ↑: higher better.)

Dataset	Method	#Param ↓	Comp.[cm] ↓	Acc.[cm] ↓	Comp.Ratio[%]↑	Acc.Ratio[%]↑
MaiCity	*VDBFusion* [7]	\	27.33	**1.36**	78.12	**99.13**
	SHINE-Mapping [6]	4.53×10^6	3.34	1.66	95.43	97.09
	Ours	$\mathbf{1.27 \times 10^6}$	**2.68**	1.52	**97.27**	97.60
NewerCollege	*VDBFusion*	\	13.20	**5.50**	91.51	**98.10**
	SHINE-Mapping	1.14×10^7	**9.55**	7.60	**94.58**	91.37
	Ours	$\mathbf{1.60 \times 10^6}$	9.68	6.72	94.10	93.69

2.3 High-Fidelity SLAM with 3DGS

In addition to accurate *geometric* reconstruction as described above, we have also explored how to reconstruct the scene with high fidelity *appearance*, aiming to enhance localization and enable additional applications based on rendering novel RGB views from unseen viewpoints [8].

In this work [8], we use 3D Gaussians as scene representation primitives to provide metrically accurate pose tracking and visually realistic reconstruction. Our two main contributions are (1) a Gaussian densification strategy based on the rendering loss to

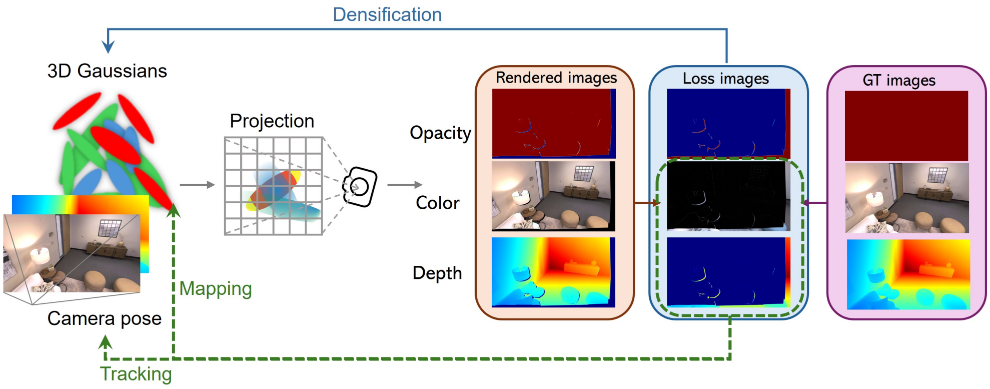


Figure 4: Overview of the mapping and tracking method with 3D Gaussian splats, developed in WP3. Our method takes RGBD frames as inputs. During mapping, when given a posed RGBD frame, we first render the opacity image, color image, and depth image. Then, we compare the rendered images with the given ("ground truth") input frames to densify the existing map. During tracking, we minimize the color and depth re-rendering loss to optimize the camera pose.

map unobserved areas and refine reobserved areas and (2) regularization parameters to alleviate the "forgetting" problem that otherwise happens during continuous mapping – where parameters tend to overfit the latest frame and result in decreasing rendering quality for previous frames (illustrated in Figure 5). Both mapping and tracking are performed with Gaussian parameters to minimize re-rendering loss in a differentiable way.

Compared to recent neural and concurrently developed Gaussian splitting RGBD SLAM baselines, our method achieves state-of-the-art benchmark results on the synthetic dataset `Replica` and competitive results on the real-world dataset TUM.

Figure 4 shows an overview of our method. Given RGBD frames and estimated camera poses, we update the map by comparing the rendered images and the ground truth to identify unobserved regions and areas requiring refinement. Regularization terms are incorporated into the optimization process to mitigate the issue of forgetting during mapping. For tracking, we track the camera pose in the Gaussian map by minimizing color and depth re-rendering loss.

In the quantitative experiment Table 2, we show that our method achieves better reconstruction results than current state-of-the-art neural implicit SLAM methods and the concurrent work using Gaussian splatting. Further results are available in the published paper, Sun et al. [8].

3 Heterogeneous map merging (T3.2)

Task T3.2 was intended to study methods for exploiting rough prior maps such as floor plans for assisted SLAM, mutual map improvement, and information transfer between map representations. We have ported a previous implementation for heterogeneous map merging called auto-complete graph (ACG) [15] and deployed it on the DARKO robot platform in particular for Milestone 2.

Figure 6 shows examples of using ACG. Alternatively to the "uninformed" SLAM system from T3.1, when using ACG for SLAM, the robot is localized with NDT-MCL using a 2D NDT map extracted from the current shape of the prior map, while a sensor-based NDT submap is being created. The corners and walls extracted from the sensor-based map

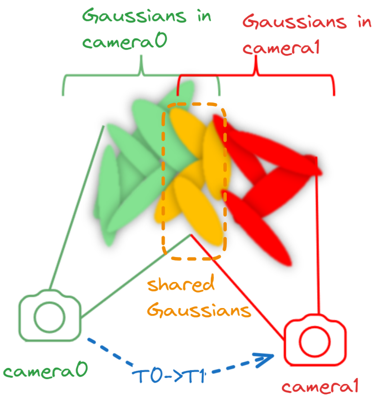
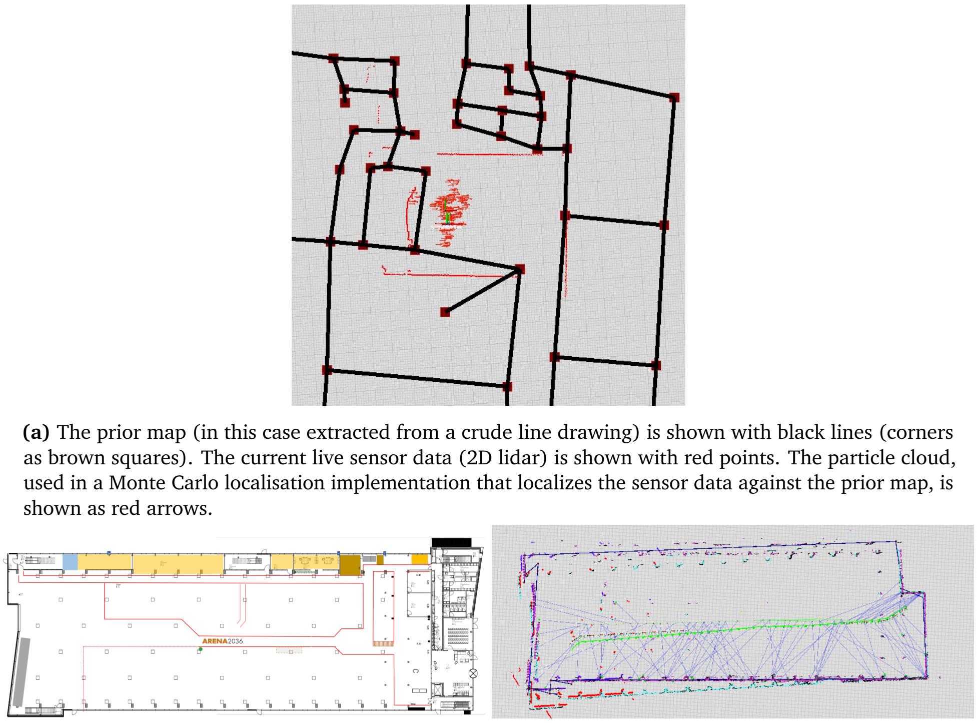


Figure 5: Illustration of the *forgetting* problem in the context of continual mapping based on Gaussians. The Gaussians colored yellow are shared by camera0 and camera1. However, these Gaussians tend to be optimized to overfit the latest frame camera1, resulting in a drop in reconstruction quality for previous frames.

Table 2: Map rendering performance with Gaussian splatting on the `Replica` [9] dataset. The best results are highlighted by **first** , second , and third . ↑ means larger is better while ↓ means smaller is better. Our method [8] achieves the best results in most metrics.

Method	**Primitives**	**Metric**	`Room0`	`Room1`	`Room2`	`office0`	`office1`	`office2`	`office3`	`office4`	**Avg.**
NICE-SLAM [10]	Neural + Voxels	PSNR [dB] ↑	22.12	22.47	24.52	29.07	30.34	19.66	22.23	24.94	24.42
		SSIM ↑	0.69	0.76	0.81	0.87	0.89	0.80	0.80	0.86	0.81
		LPIPS ↓	0.33	0.27	0.21	0.23	0.18	0.23	0.21	0.20	0.23
		ATE RMSE [cm] ↓	0.97	1.31	1.07	0.88	1.00	1.06	1.10	1.13	10.6
		Depth L1 [cm] ↓	1.81	1.44	2.04	1.39	1.76	8.33	4.99	2.01	2.97
ESLAM [11]	Neural + Feature Plane	PSNR [dB] ↑	25.25	25.31	28.09	30.33	27.04	27.99	29.27	29.15	27.80
		SSIM ↑	0.87	0.25	0.93	0.93	0.91	0.94	0.95	0.95	0.92
		LPIPS ↓	0.32	0.30	0.25	0.21	0.25	0.24	0.19	0.21	0.25
		ATE RMSE [cm] ↓	0.71	0.70	0.52	0.57	0.55	0.58	0.72	0.63	0.63
		Depth L1 [cm] ↓	0.97	1.07	1.28	0.86	1.26	1.71	1.43	1.06	1.18
Point-SLAM [12]	Neural + Point Cloud	PSNR [dB] ↑	32.40	34.08	35.50	38.26	39.16	**33.99**	**33.48**	33.49	35.17
		SSIM ↑	0.97	**0.98**	0.98	0.98	**0.99**	0.96	0.96	**0.98**	0.97
		LPIPS ↓	0.11	0.12	0.11	0.10	0.12	0.16	0.13	0.14	0.12
		ATE RMSE [cm] ↓	0.61	0.41	0.37	0.38	0.48	0.54	0.69	0.72	0.52
		Depth L1 [cm] ↓	0.53	**0.22**	0.46	0.30	0.57	**0.49**	**0.51**	**0.46**	**0.44**
GS-SLAM [13]	Parameterized Gaussians	PSNR [dB] ↑	31.56	32.86	32.59	38.70	**41.17**	32.36	32.03	32.92	34.27
		SSIM ↑	0.96	0.97	0.97	0.98	**0.99**	0.97	**0.97**	0.96	0.97
		LPIPS ↓	0.09	0.07	0.09	0.05	**0.03**	0.09	0.11	0.11	0.08
		ATE RMSE [cm] ↓	0.48	0.53	0.33	0.52	0.41	0.59	0.46	0.70	0.50
		Depth L1 [cm]	1.31	0.82	1.26	0.81	0.96	1.41	1.53	1.08	1.16
SplaTAM [14]	Parameterized Gaussians	PSNR [dB] ↑	32.86	33.89	35.25	38.26	39.17	31.97	29.70	31.81	34.11
		SSIM ↑	**0.98**	0.97	0.98	0.98	0.98	0.97	0.95	0.95	0.97
		LPIPS ↓	0.07	0.10	0.08	0.09	0.09	0.10	0.12	0.15	0.10
		ATE RMSE [cm] ↓	0.31	0.40	0.29	0.47	0.27	0.29	0.32	0.55	0.36
		Depth L1 [cm]	–	–	–	–	–	–	–	–	–
Ours	Parameterized Gaussians	PSNR [dB] ↑	**33.06**	**35.74**	**37.21**	**41.12**	41.11	33.56	33.21	**34.48**	**36.19**
		SSIM ↑	**0.98**	**0.98**	**0.99**	**0.99**	**0.99**	**0.98**	**0.97**	**0.98**	**0.98**
		LPIPS ↓	**0.05**	**0.05**	**0.04**	**0.03**	**0.03**	**0.07**	**0.08**	**0.08**	**0.05**
		ATE RMSE [cm] ↓	**0.19**	**0.34**	**0.16**	**0.21**	**0.26**	**0.23**	**0.21**	**0.38**	**0.25**
		Depth L1 [cm] ↓	**0.39**	0.34	**0.33**	**0.29**	**0.26**	0.67	0.93	0.97	0.52



(a) The prior map (in this case extracted from a crude line drawing) is shown with black lines (corners as brown squares). The current live sensor data (2D lidar) is shown with red points. The particle cloud, used in a Monte Carlo localisation implementation that localizes the sensor data against the prior map, is shown as red arrows.

(b) Running ACG for heterogeneous map merging at Milestone 2 at ARENA2036. Left: floor plan used as input. Right: visualization of the mapping. The wall outline extracted from the left image is shown with black lines. Cyan points show a 2D rendering of the lidar-based map as it would look like without merging with the floor plan. Purple points show the corrected map. Blue lines denote edges in the factor graph that connect robot poses to detected corners.

Figure 6: The auto-complete graph (ACG) as integrated in the DARKO system.

are associated with the corners and walls prior to the edges in the ACG, and the graph is repeatedly optimized with a set of two robust back-ends in tandem (a Huber kernel followed by dynamic covariance scaling [16]), in order to deal with large numbers of false corner associations. One problem with the current implementation of the system, which is also evident in Figure 6, is that the line extractor is rather crude and the prior (black) looks distorted, which makes the mutual map correction difficult.

Ultimately, the attention of WP3 focused more on the remaining tasks T3.1, T3.3, T3.4.

4 Maps of dynamics (T3.3)

Maps of dynamics (MoDs) are representations of motion patterns learned from prior observations. Within DARKO we have developed novel MoD representations and, in particular, studied how MoDs can be exploited for downstream tasks such as human-aware robot navigation and long-term human motion prediction.

4.1 The CLiFF-map

The back bone of our work on MoDs has been the CLiFF-map representation [17]. CLiFF-maps represent speed and direction jointly as velocity $\mathbf{V} = [\theta, \rho]^T$ using direction θ and speed ρ, where $\rho \in \mathbb{R}^+$, $\theta \in [0, 2\pi)$. For each of a set of discrete locations in a map, CLiFF

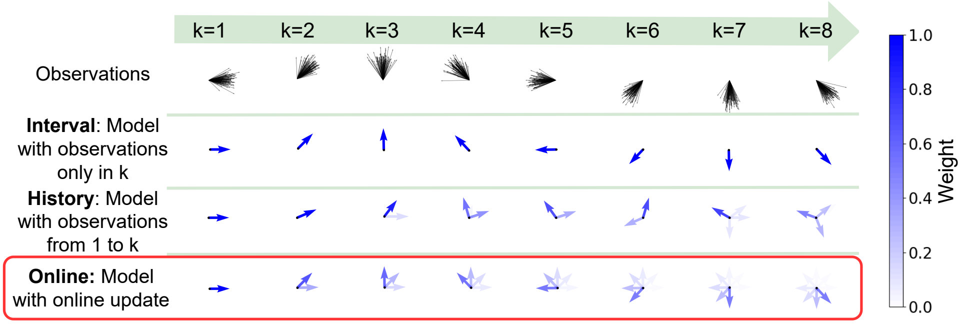

k=1
k=2
k=3
k=4
k=5
k=6
k=7
k=8
Observations
Interval: Model with observations only in k
History: Model with observations from 1 to k
Online: Model with online update
1.0
0.8
0.6
0.4
0.2
0.0
Weight

Figure 7: Online update results for a representative toy example; compared with using only new observations (*Interval*) and using all observations (*History*) to build the model. The **top row** shows raw observations for each of the eight directions $(0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°)$, provided in each iteration k. Blue arrows depict the mean vectors of a CLiFF Gaussian mixture model, with transparency indicating the component weights. Three modeling approaches are compared: the **second row** shows models built using only observations from the current iteration k; the **third row** shows models built with cumulative observations from iteration 1 to k, which are over generalized and fail to prioritize recent observations; the **fourth row** shows the proposed online-update models, which incorporate new data while retaining relevant historical patterns, offering a dynamic representation of the motion pattern over time.

fits a semi-wrapped Gaussian mixture model (GMM) (i. e., a cylindrical distribution) to motion observations within a certain radius. The GMM is estimated using the mean shift algorithm and expecation maximization, and as such is typically computed offline given a batch of measurements. Therefore it is most well suited to learning maps of flow in environments where the flow is stationary, i.e., not assumed to change over time.

4.2 Online updates of CLiFF-map

In DARKO, we have proposed a method to update a CLiFF map of dynamics in a lifelong operating robot, by using a variation of the stochastic expectation maximization algorithm [18]. As new observations are collected, our goal is to update the existing representation to effectively and accurately integrate the new information. At the same time, the robot should not immediately dismiss the previously learned patterns without the need to store the entire historical dataset.

Our proposed online update method maintains the probabilistic representation in each observed location, updating parameters by continuously tracking sufficient statistics. More details are available in our paper, Zhu et al. [19].

As shown in Figure 7, our method not only ensures that the model remains adaptively accurate in reflecting the most recent human motion but also maintains consistency with historical data, thereby preserving a comprehensive understanding of the environment over time. In experiments on both a synthetic dataset and the real-world ATC [20] dataset, we show that our method is able to quickly recognize changes in environments with sparse and dense motion flows, while being significantly faster than baseline methods, as shown in Figure 8.

4.3 CLiFF-LHMP

Long-term human motion prediction (LHMP) is important for mobile service robots and intelligent vehicles to operate safely and smoothly around people. The more accurate

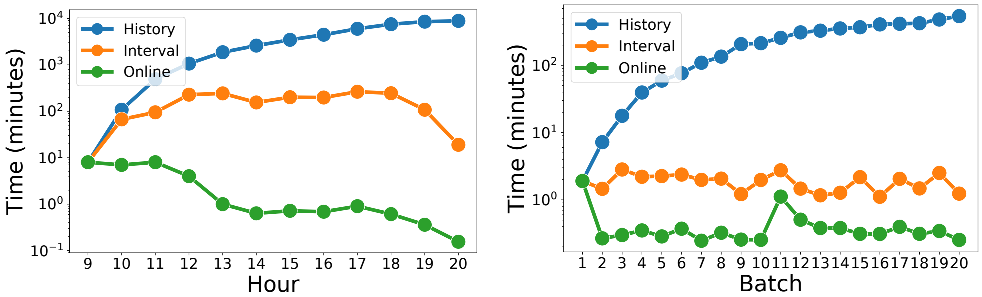
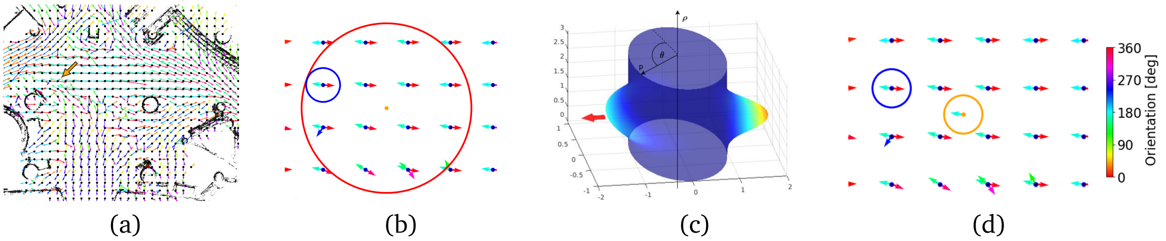


Figure 8: Running time of each iteration when building MoDs from the ATC dataset (**left**) and *den520d* dataset (**right**). In both datasets, the online model shows significantly reduced runtime compared with history and interval models.



(a) (b) (c) (d)

Figure 9: In CLiFF-LHMP, sampling a direction from the CLiFF-map has four steps. **(a)** CLiFF-map. The location to sample from is marked with an orange arrow. **(b)** Selection of SWGMMs in the CLiFF-map: The red circle contains all SWGMMs within a set distance from the sampling location. From these SWGMMs, the SWGMM with the highest motion ratio is selected (marked with a blue circle). **(c)** The SWGMM distribution in the selected location wrapped on a unit cylinder. The speed is represented by the position along the ρ axis and the direction is θ. The probability is represented by the distance from the surface of the cylinder. A velocity vector (marked with a red arrow) is sampled from this SWGMM. **(d)** The direction value of the sampled velocity is shown in the sampled direction and marked with an orange circle.

predictions are, particularly over extended periods of time, the better a system can, e.g., assess collision risks and plan ahead. However, accurate prediction of human trajectories is challenging due to complex factors, including, for example, social norms and environmental conditions. The influence of such factors can be captured through MoDs, which encode spatial motion patterns learned from (possibly scattered and partial) past observations of motion in the environment and which can be used for data-efficient, interpretable motion prediction. We propose to exploit maps of dynamics for long-term human motion prediction (LHMP).

In DARKO, we have proposed CLiFF-LHMP [21]. The motion patterns represented in a CLiFF-map implicitly avoid collisions with static obstacles and follow the topological structure of the environment, e.g., capturing the dynamic flow through a hall into a corridor (see Figure 10). We bias a constant velocity prediction with samples from the CLiFF-map to generate multi-modal trajectory predictions. The sampling process is described in Figure 9. The algorithm of CLiFF-LHMP is presented in Algorithm 1.

Here we report evaluation of the predictive performance using two real-world datasets: ATC [20] and THÖR [22]. The baseline prediction approaches include IS-MDP [23] and the constant velocity predictor [24, 25]. Figure 11 shows that CLiFF-LHMP is 45 % more accurate than the baseline at 50 s, with average displacement error (ADE) below 5 m up to 50 s. In contrast to prior art in long-term environment-aware motion prediction

Algorithm 1: CLiFF-LHMP

```
Input: $\mathscr{H}$, $x_{t_0}$, $y_{t_0}$, $\Xi$
Output: $\mathscr{T}$
$\mathscr{T} = \{\}$
$\rho_{\text{obs}}, \theta_{\text{obs}} \leftarrow$ getObservedVelocity($\mathscr{H}$)
$s_{t_0} = (x_{t_0}, y_{t_0}, \rho_{\text{obs}}, \theta_{\text{obs}})$
for $t = t_0 + 1, \dots, t_0 + T_p$ do
    $x_t, y_t \leftarrow$ getNewPosition($s_{t-1}$)
    $\theta_s \leftarrow$ sampleDirectionFromCLiFFmap($x_t, y_t, \Xi$)
    $(\rho_t, \theta_t) \leftarrow$ predictVelocity($\theta_s$, $\rho_{t-1}$, $\theta_{t-1}$)
    $s_t \leftarrow (x_t, y_t, \rho_t, \theta_t)$
    $\mathscr{T} \leftarrow \mathscr{T} \cup s_t$
return $\mathscr{T}$
```

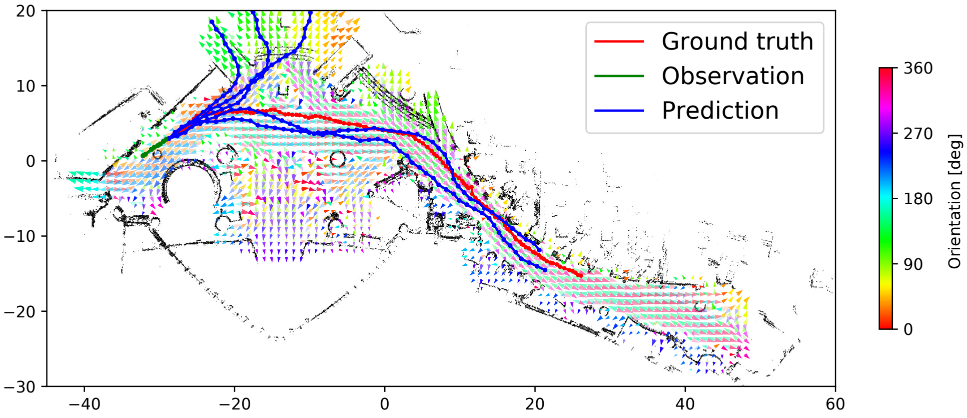


Figure 10: Long-term (50 s) motion prediction result obtained with CLiFF-LHMP for one person in the ATC dataset. **Red** line: ground truth trajectory. **Green** line: observed trajectory. **Blue** lines: predicted trajectories. The CLiFF-map is shown with colored arrows.

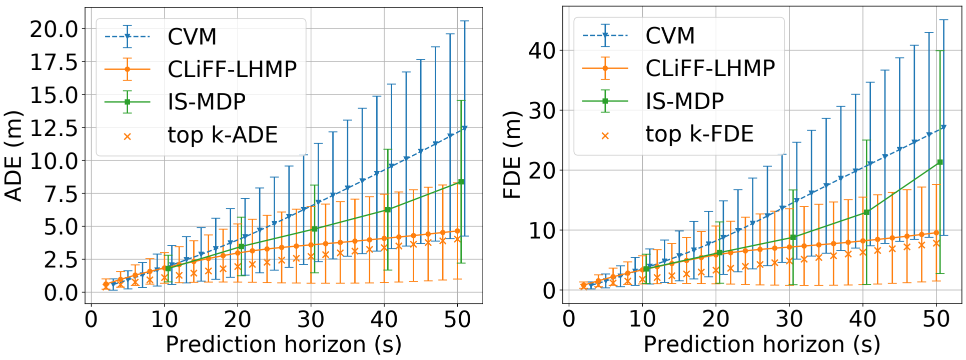


Figure 11: Average displacement error (ADE) and final displacement error (mean ± one std. dev.) in the ATC dataset with prediction horizon 1–50 s.

[23], CLiFF-LHMP method does not make any assumptions on the optimality of human motion and instead generalizes the features of human-space interactions from the learned MoD. Furthermore, our method does not require a list of goals in the environment as input, in contrast to prior planning-based prediction methods. Finally, our method can flexibly estimate the variable time end-points of human motion, predicting both short- and long-term trajectories, in contrast to the prior art which always predicts up to a fixed prediction horizon.

We have also demonstrated a *class-conditioned* application of CLiFF-LHMP, where different CLiFF-maps are constructed for people with different roles, and show that doing so further increases prediction performance (Almeida et al. [26]).

4.4 LaCE-LHMP

Detecting and identifying abnormal trajectories is a major challenge in motion modeling and prediction. Existing methods typically identify abnormal motions by comparing them to expected behaviours [27] or measuring deviations from normal motions [28]. However, these approaches require labelled data for supervised learning.

To address the limitations of prior work, especially regarding accuracy and sensitivity to anomalies in long-term prediction, we propose the Laminar Component Enhanced LHMP approach (LaCE-LHMP) [29]. This approach is inspired by data-driven airflow modeling, which estimates laminar and turbulent flow components and uses predominantly the laminar components to make flow predictions. Based on the hypothesis that human trajectory patterns also manifest laminar flow (that represents predictable motion) and turbulent flow components (that reflect more unpredictable and arbitrary motion), LaCE-LHMP extracts the laminar patterns in human dynamics and uses them for human motion prediction.

The LaCE-LHMP approach consists of training and prediction phases, as shown in Figure 12. The training phase first extracts the underlying laminar component from the observed trajectories and learns an MoD, expressed through a set of probabilistic representations of the target area, i.e., the LaCE model. In the prediction phase, both the observed recent trajectory sequence and the learned LaCE model influence the predicted trajectory, depending on the degree of local laminar dominance. In order to select the contributions from both factors depending on the local situation, we propose an adaptive sampling process. Once a likely direction is sampled, the current state can be propagated to predict sequences of future states.

We compare the performance of our approach with three baselines: CLiFF-LHMP, Trajectron++ [30] and a constant velocity model, using the ATC dataset. Trajectron++ (T++) represents a state-of-the-art approach employing a graph-structured generative neural network based on a conditional-variational autoencoder.

4.5 Flow-aware motion planning

Furthermore, we have studied the effect of different MoD-aware sampling methods for *motion planning* on MoDs. A manuscript describing this study is currently under review for the Robotics and Autonomous Systems journal.

We have proposed improvements to the existing Dijkstra-graph sampling heuristic that is used in the CLiFF-RRT* and DTC-RRT* [31] methods; and we show that an *ellipsoidal heuristic*, inspired by Gammell et al. [32], can also be used with maps of dynamics, and propose two novel sampling heuristics.

We have experimentally validate several sampling heuristics through a comprehensive evaluation (> 37 000 runs) of their performance on real-world data from densely populated environments. Our results show that the proposed sampling heuristics help both to

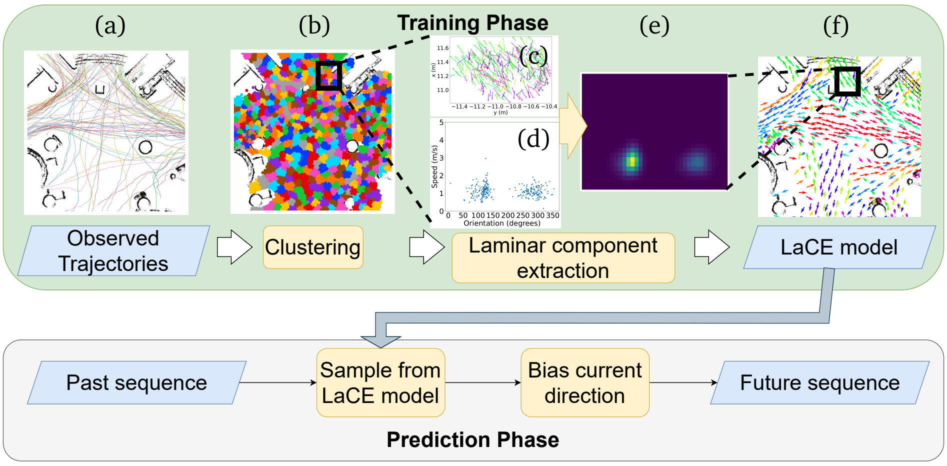


Figure 12: Diagram illustrating the training and prediction phases of the LaCE-LHMP approach. In the training phase, observed trajectories **(a)** are used. Velocity observations, which are depicted in **(c)** for (x, y) and **(d)** for ω-v distribution, are clustered using K-means into K clusters, shown in **(b)**. From each cluster's joint ω-v distribution, a discrete ω-v histogram Γ^R is estimated to extract the laminar component Γ^L, as shown in **(e)**. The directions with the highest likelihood in Γ^L are represented by colored arrows in the LaCE model **(f)**. The LaCE model is then utilized for prediction.

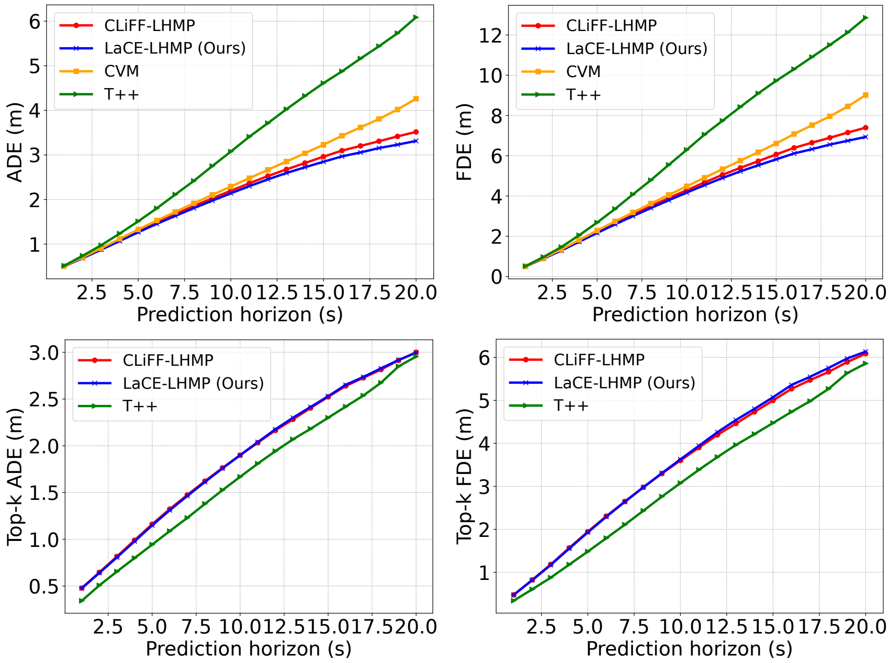


Figure 13: ADE/FDE (**top**) and top-k ADE/FDE (**bottom**) in the ATC dataset with a prediction horizon 1–20 s. Predictions with the LaCE model are more accurate during the whole considered period, as indicated by lower ADE/FDE values, which signify improved performance.

find solutions and reach low-cost solutions quickly. In particular, we devise a hybrid sampling heuristic that finds solutions quickly, especially in constrained areas where uninformed sampling struggles: hybrid sampling heuristics achieve 90% success rate after 16 s compared to 64 s for uniform sampling. This hybrid method combines Dijkstra graph search, intensity-importance sampling, and the dynamics-aware ellipsoidal heuristic (delimiting the sampling region based on a theoretical upper bound of human-aware path cost).

5 Reliability-aware mapping and safe localization (T3.4)

5.1 Registration quality assessment

Scan registration is a central part of both the mapping and the localization pipelines from T3.1. As such, automatic assessment of the result of scan matching is important for detecting and mitigating errors and improving mapping and localization. We have reported our registration quality assessment *CorAl* [33] in D3.2 and briefly cover it here for completeness.

CorAl computes the average differential entropy in two point clouds, comparing the local point entropy in each point cloud separately to the union of the point clouds. A key idea is to estimate the entropy inherent in the scene from the entropy in the separate point clouds, which enables CorAl to accurately assess quality in a range of different environments. The decision boundaries between aligned vs non-aligned point clouds can be learned in a self-supervised fashion from accurately aligned scans with poses.

5.2 Localisation quality assessment

Precise localization is key to most mobile robot systems, not least those deployed in industrial settings. However, even state-of-the-art lidar-based systems may fail or lose accuracy, particularly in feature-sparse environments (e. g., fully stacked warehouse aisles or transport corridors).

Our aim is to be able to predict localization risk (i.e., the risk of generating inaccurate pose estimates) and account for it by taking preemptive measures, e.g., such that a planner can generate “risk-aware” paths that take both the risk of inaccurate localization and the path length into account.

In D3.2, we reported on *alignability maps* [34], meant to associate a cost to map areas that lack features helpful for precise scan alignment. In contrast to CorAl (section 5.1) which assesses pairwise alignment “after the fact”, alignability maps are helpful for *proactively* avoiding inaccurate localization.

An alignability map, in our case, is a 2D grid map in which each cell represents the expected alignability that can be obtained from different scans within that area. Our quantitative experiments [34] have shown that alignability can be used as an indicator of localization error and we have validated, with Granger causality tests, that it also serves to *anticipate* the occurrence of errors. These results were covered in D3.2.

New in the final implementation of localization risk maps is a second layer that quantifies the expected level of *dynamics* in the map, as areas with many changes also pose a risk of inaccurate localization. This work is about to be submitted for a journal publication, but we will provide a brief overview in this deliverable.

We quantify dynamics in this work by relying on the independent Markov chain approach (iMac) [35], although our implementation differs in some aspects. The iMac approach is based on an occupancy grid map representation, and it considers each cell as an independent Markov chain with two states: occupied or free. This is aimed at modeling

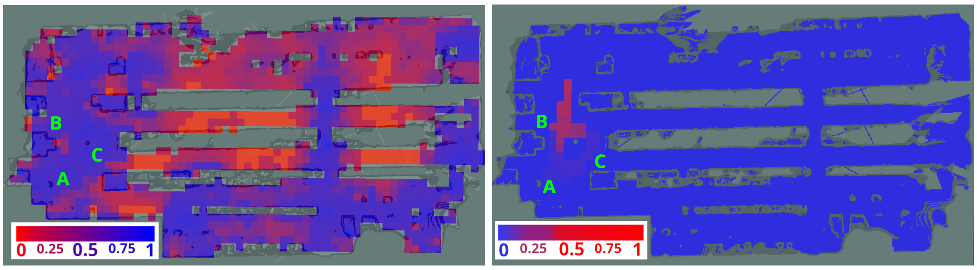


Figure 14: An example of a Localization Risk Map (LRM) from a warehouse environment. Left: alignability layer $\mathscr{L}_{\mathrm{A}}$. Right: dynamics layer $\mathscr{L}_{\mathrm{D}}$. Red values indicate higher risk (lower alignability, higher dynamics).

dynamics in an environment by quantifying the expected amount of transitions between states that take place in a particular cell. The transition probabilities for each Markov chain are defined on the expected values of two different Poisson processes, one that models the probability of a cell changing its state from free to occupied, and another one that models the probability of a cell changing its state from occupied to free.

Figure 14 shows an example of a localization risk map from a warehouse environment, showing the alignability and dynamics layers.

We have also introduced a novel probabilistic model in the form of a Bayesian network that enables the prediction of localization errors given the conditions of the environment. We have opted for the use of Bayesian networks since they are grounded on a rigorous mathematical framework that allows for a compact and intuitive representation of expert knowledge and enables to perform deductions on such knowledge while considering the uncertainty of the domain. The model proposed in this work is a Conditional Linear-Gaussian (CLG) Bayesian network, which includes both discrete and continuous random variables. The proposed CLG Bayesian network structure is depicted in Figure 15, which captures the dependencies among the random variables considered. Alignability and dynamics influence localization error. However, these two variables do not depend on each other. Furthermore, it is reasonable to state that the initial error and the traveled distance do not rely on each other either. The error obtained at the end of a given path may change depending on the initial one, even for the same environmental conditions (e. g., a severe initial error might not be possible to correct even with favorable conditions.) Therefore, if available, an estimate of the initial error can also be provided as input for the model. The model in Figure 15 can, after training on a set of trajectories with known localization error, be used to predict the localization risk for a given path.

Our preliminary results indicate that travel distance is an important indicator of localization risk but including information from the alignability and dynamics map layers over the path further decreases the difference between the mean and variance of the estimated localization risk and the mean and variance of the actual one.

5.3 Map quality assessment

Occupancy grid maps are widely used in the robotics landscape, including DARKO, and offer a convenient way to leverage image-based learning methods.In occupancy grid maps, regions can misrepresent the environment, for example, where a wall is too thick or regions that are only partially explored and the map is incomplete in those areas. In practice, maps are usually assessed qualitatively by a human expert, but these assessments are not easily reproducible and tend to vary significantly between people. Furthermore, assessing all of the regions of a map by a human is a time-consuming task. With this variability in

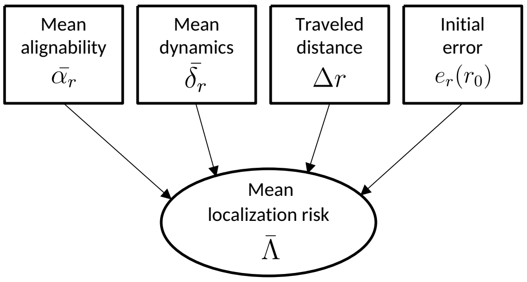

Mean alignability $\bar{\alpha_r}$
Mean dynamics $\bar{\delta_r}$
Traveled distance Δr
Initial error $e_r(r_0)$
Mean localization risk $\bar{\Lambda}$

Figure 15: Graph structure of the Conditional Linear-Gaussian (CLG) Bayesian network proposed for the prediction of localization risk. The squared nodes represent discrete random variables while the round one represents a continuous random variable.

mind, a data-driven method is chosen to limit potential sources of errors and ensure a reproducible and repeatable process for mapping quality assessment.

We contribute to *robot introspection* through self-assessment of map quality, using only a given map as input, to be independent of the underlying mapping algorithm. Current methods for map quality assessment rely on ground truth or labeled information for training classifiers, both of which can be difficult to obtain. We instead propose a self-supervised learning approach to map quality assessment of 2D occupancy grid maps, which identifies regions of a map that require further investigation. Errors may manifest in a map in different ways. For instance, local misregistration often appears as doubled or slightly bent walls. Errors due to faulty loop closure may appear as overlapping map segments. More subtly, errors due to sparse coverage may manifest as poorly reconstructed or partially missing obstacles. Other errors would include remnants of moving obstacles and clutter that were not properly cleared from the map.

As described in D3.2, our proposed approach uses a variational autoencoder (VAE) to learn the encoding of maps in an unsupervised way. An autoencoder trained on patches of the map should converge on an encoding that captures the majority of the map and represents the environment accurately, and the encoding will poorly represent the poor sections. We can then analyze the reconstruction loss of a patch by comparing the input patch cell values with those generated using the encoder. Cell values with a higher reconstruction loss should then correspond to regions with potential map errors.

The inputs to the VAE are map patches, $\mathbf{P}$, extracted from the occupancy grid map. The objective of the autoencoder is to minimize the reconstruction error and match the latent space prior $p(\mathbf{z})$ by jointly optimizing the generative parameters θ and embedding parameters ϕ. The encoder network maps from patches to Gaussians in the latent space $q_\phi(\mathbf{z} \mid \mathbf{P}) = \mathscr{N}(\mu_\phi(\mathbf{P}), S_\phi(\mathbf{P}))$ and the decoder network maps from latent space positions to patches $f_\theta(\mathbf{z}) = \mathbf{P}'$. The reconstruction loss for patches $\mathscr{L}_{\theta,\phi}(\mathbf{P})$ is implemented as $\mathscr{L}_{\theta,\phi}(\mathbf{P}) = \text{argmax}_{\theta,\phi}\left(\mathbb{E}_{\mathbf{z}\sim q_\phi(.|\mathbf{P})}\left(\frac{||\mathbf{P}-\mathbf{P}'||^2}{2c}\right) - D_{KL}(q_\phi(. \mid \mathbf{P}) \mid\mid p(.))\right)$. The parameters are learned end-to-end.

In our evaluation, the proposed approach identified the same bad map cells compared to a baseline map quality assessment tool that required labelled map patches to train a classifier. The primary advantage of our method is that it requires no labels or ground truth, which can be difficult or infeasible to obtain. Additionally, the method was successful when assessing maps with various sources of error from environments not encountered during training.

Example output is shown in Figure 16, for an occupancy grid map from the same warehouse environment as shown in Figure 14.

The unsupervised and reference-free map quality assessment tool that was described

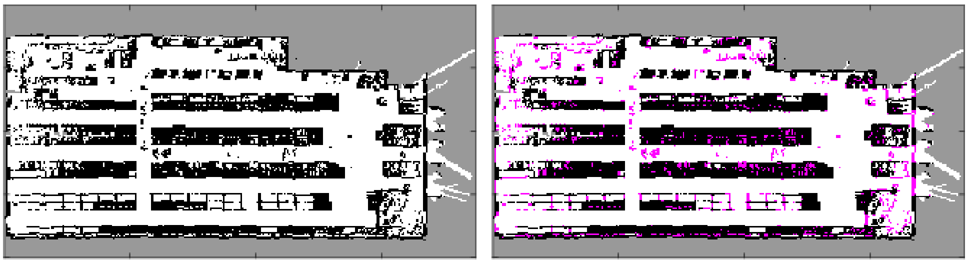


Figure 16: Warehouse grid map (left) and grid map with *bad* cells highlighted by our reference-free map quality assessment method (right). In this example, the method mostly highlights clutter and sparsely covered regions as *bad*.

in D3.2 has also been demonstrated at the Milestone 3 demo and stakeholder meeting. We are preparing a journal manuscript describing the technical details and experimental outcomes in more detail.

6 Summary

This report describes the components included in the DARKO mapping system as of Month 48 (December 2024). Efficient mapping has been implemented initially using the 3D-NDT map representation (which allows accurate localization using a sparse grid of 3D Gaussians, compared to dense octrees or point cloud maps) and further on by exploring efficient representations for neural surface reconstruction (Section 2.2) and improved rendering quality of 3D Gaussian splatted maps (Section 2.3). Our work on maps of dynamics includes an efficient online version of the CLiFF representation (Section 4.2) as well as long-term human motion prediction with CLiFF (Section 4.3) and the novel LaCE representation (Section 4.4). Reliability-aware mapping and safe localization has been addressed through self-supervised registration quality assessment (Section 5.1), dynamics- and alignability-aware localization risk maps, including a Bayesian model for risk assessment (Section 5.2), and a variational autoencoder for reference-free 2D map quality assessment (Section 5.3).

References

[1] Daniel Adolfsson, Stephanie Lowry, Martin Magnusson, Achim J. Lilienthal, and Henrik Andreasson. “A Submap per Perspective - Selecting Subsets for SuPer Mapping that Afford Superior Localization Quality”. In: *European Conference on Mobile Robots*. 2019.

[2] Henrik Andreasson, Daniel Adolfsson, Todor Stoyanov, Martin Magnusson, and Achim J. Lilienthal. “Incorporating Ego-motion Uncertainty Estimates in Range Data Registration”. In: *IEEE Int. Conf. on Intell. Rob. and Systems (IROS)*. Sept. 2017, pp. 1389–1395.

[3] Jari Saarinen, Henrik Andreasson, Todor Stoyanov, and Achim J Lilienthal. “3D Normal Distributions Transform Occupancy Maps: An Efficient Representation for Mapping in Dynamic Environments”. In: *The International Journal of Robotics Research* 32.14 (2013), pp. 1627–1644.

[4] Jari Saarinen, Henrik Andreasson, Todor Stoyanov, and Achim Lilienthal. “Normal Distribution Transform Monte-Carlo Localization (NDT-MCL)”. In: *Proc. IEEE/RSJ Int. Conf. on Intell. Robots and Syst.* 2013, pp. 382–389.

[5] Shuo Sun, Malcolm Mielle, Achim J. Lilienthal, and Martin Magnusson. “3QFP: Efficient neural implicit surface reconstruction using Tri-Quadtrees and Fourier feature Positional encoding”. In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. 2024, pp. 4036–4044.

[6] Xingguang Zhong, Yue Pan, Jens Behley, and Cyrill Stachniss. “Shine-mapping: Large-scale 3d mapping using sparse hierarchical implicit neural representations”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pp. 8371–8377.

[7] Ignacio Vizzo, Tiziano Guadagnino, Jens Behley, and Cyrill Stachniss. “VDBFusion: Flexible and Efficient TSDF Integration of Range Sensor Data”. In: *Sensors* 22.3 (2022).

[8] Shuo Sun, Malcolm Mielle, Achim J Lilienthal, and Martin Magnusson. “High-Fidelity SLAM Using Gaussian Splatting with Rendering-Guided Densification and Regularized Optimization”. In: *arXiv preprint arXiv:2403.12535* (2024).

[9] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. “The Replica Dataset: A Digital Replica of Indoor Spaces”. In: *arXiv preprint arXiv:1906.05797* (2019).

[10] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. “Nice-slam: Neural implicit scalable encoding for slam”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 12786–12796.

[11] Mohammad Mahdi Johari, Camilla Carta, and François Fleuret. “ESLAM: Efficient Dense SLAM System Based on Hybrid Representation of Signed Distance Fields”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 17408–17419.

[12] Erik Sandström, Yue Li, Luc Van Gool, and Martin R Oswald. “Point-SLAM: Dense Neural Point Cloud-based SLAM”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 18433–18444.

[13] Chi Yan, Delin Qu, Dong Wang, Dan Xu, Zhigang Wang, Bin Zhao, and Xuelong Li. “GS-SLAM: Dense Visual SLAM with 3D Gaussian Splatting”. In: *arXiv preprint arXiv:2311.11700* (2023).

[14] Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. “SplaTAM: Splat, Track & Map 3D Gaussians for Dense RGB-D SLAM”. In: *arXiv preprint arXiv:2312.02126* (2023).

[15] Malcolm Mielle, Martin Magnusson, and Achim J. Lilienthal. “The Auto-Complete Graph: Merging and Mutual Correction of Sensor and Prior Maps for SLAM”. In: *Robotics* 8.2 (2019).

[16] Pratik Agarwal, Gian Diego Tipaldi, Luciano Spinello, Cyrill Stachniss, and Wolfram Burgard. “Robust map optimization using dynamic covariance scaling”. In: *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*. 2013, pp. 62–69.

[17] Tomasz Piotr Kucner, Martin Magnusson, Erik Schaffernicht, Victor H. Bennetts, and Achim J. Lilienthal. “Enabling Flow Awareness for Mobile Robots in Partially Observable Environments”. In: *IEEE Robotics and Automation Letters (RA-L)* 2.2 (Apr. 2017), pp. 1093–1100.

[18] Olivier Cappé and Eric Moulines. "On-Line Expectation–Maximization Algorithm for latent Data Models". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 71.3 (2009), pp. 593–613.

[19] Yufei Zhu, Andrey Rudenko, Luigi Palmieri, Lukas Heuer, Achim J. Lilienthal, and Martin Magnusson. "Fast Online Learning of CLiFF-maps in Changing Environments". In: *IEEE Int. Conf. on Rob. and Autom. (ICRA)*. 2025.

[20] D. Brščić, T. Kanda, T. Ikeda, and T. Miyashita. "Person tracking in large public spaces using 3-D range sensors". In: *IEEE Trans. on Human-Machine Systems* 43.6 (2013), pp. 522–534.

[21] Yufei Zhu, Andrey Rudenko, Tomasz P. Kucner, Luigi Palmieri, Kai O. Arras, Achim J. Lilienthal, and Martin Magnusson. "CLiFF-LHMP: Using Spatial Dynamics Patterns for Long-Term Human Motion Prediction". In: *IEEE Int. Conf. on Intell. Rob. and Systems (IROS)*. 2023.

[22] A. Rudenko, T. P. Kucner, C. S. Swaminathan, R. T Chadalavada, K. O. Arras, and A. J. Lilienthal. "THÖR: Human-Robot Navigation Data Collection and Accurate Motion Trajectories Dataset". In: *IEEE Robotics and Automation Letters (RA-L)* 5.2 (2020), pp. 676–682.

[23] A. Rudenko, L. Palmieri, A. J. Lilienthal, and K. O. Arras. "Human Motion Prediction under Social Grouping Constraints". In: *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. 2018.

[24] C. Schöller, V. Aravantinos, F. Lay, and A. Knoll. "What the constant velocity model can teach us about pedestrian motion prediction". In: *IEEE Robotics and Automation Letters (RA-L)* 5.2 (2020), pp. 1696–1703.

[25] Andrey Rudenko, Luigi Palmieri, Wanting Huang, Achim J Lilienthal, and Kai O Arras. "The Atlas Benchmark: an Automated Evaluation Framework for Human Motion Prediction". In: *Proc. of the IEEE Int. Symp. on Robot and Human Interactive Comm. (RO-MAN)*. 2022.

[26] Tiago Rodrigues de Almeida, Yufei Zhu, Andrey Rudenko, Tomasz P. Kucner, Johannes A. Stork, Martin Magnusson, and Achim J. Lilienthal. "Trajectory Prediction for Heterogeneous Agents: A Performance Analysis on Small and Imbalanced Datasets". In: *IEEE Robotics and Automation Letters (RA-L)* (2024), pp. 1–8.

[27] W. Liu, D. Lian W. Luo, and S. Gao. "Future Frame Prediction for Anomaly Detection – A New Baseline". In: *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[28] Tharindu Fernando, Simon Denman, Sridha Sridharan, and Clinton Fookes. "Soft+Hardwired attention: An LSTM framework for human trajectory prediction and abnormal event detection". In: *Neural networks* 108 (2018), pp. 466–478.

[29] Y. Zhu, H. Fan, A. Rudenko, M. Magnusson, E. Schaffernicht, and A. J. Lilienthal. "LaCE-LHMP: Airflow Modelling-Inspired Long-Term Human Motion Prediction By Enhancing Laminar Characteristics in Human Flow". In: *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*. 2024.

[30] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. "Trajectron++: Dynamically-Feasible Trajectory Forecasting With Heterogeneous Data". In: *Proc. of the Europ. Conf. on Computer Vision (ECCV)*. 2020, pp. 683–700.

[31] Chittaranjan S Swaminathan, Tomasz P Kucner, Martin Magnusson, Luigi Palmieri, and Achim J Lilienthal. "Down The CLiFF: Flow-aware Trajectory Planning under Motion Pattern Uncertainty". In: *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 6176–6181.

[32] Jonathan D. Gammell, Siddhartha S. Srinivasa, and Timothy D. Barfoot. "Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic". In: *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. 2014, pp. 2997–3004.

[33] Daniel Adolfsson, Manuel Castellano-Quero, Martin Magnusson, Achim J. Lilienthal, and Henrik Andreasson. "CorAl: Introspection for robust radar and lidar perception in diverse environments using differential entropy". In: *Robotics and Autonomous Systems* (2022), p. 104136.

[34] Manuel Castellano-Quero, Tomasz Piotr Kucner, and Martin Magnusson. "Alignability maps for the prediction and mitigation of localization error". In: *IROS 2023 Workshop on "Closing the Loop on Localization"*.

[35] Jari Saarinen, Henrik Andreasson, and Achim J. Lilienthal. "Independent Markov chain occupancy grid maps for representation of dynamic environment". In: *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. 2012, pp. 3489–3495.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 101017274