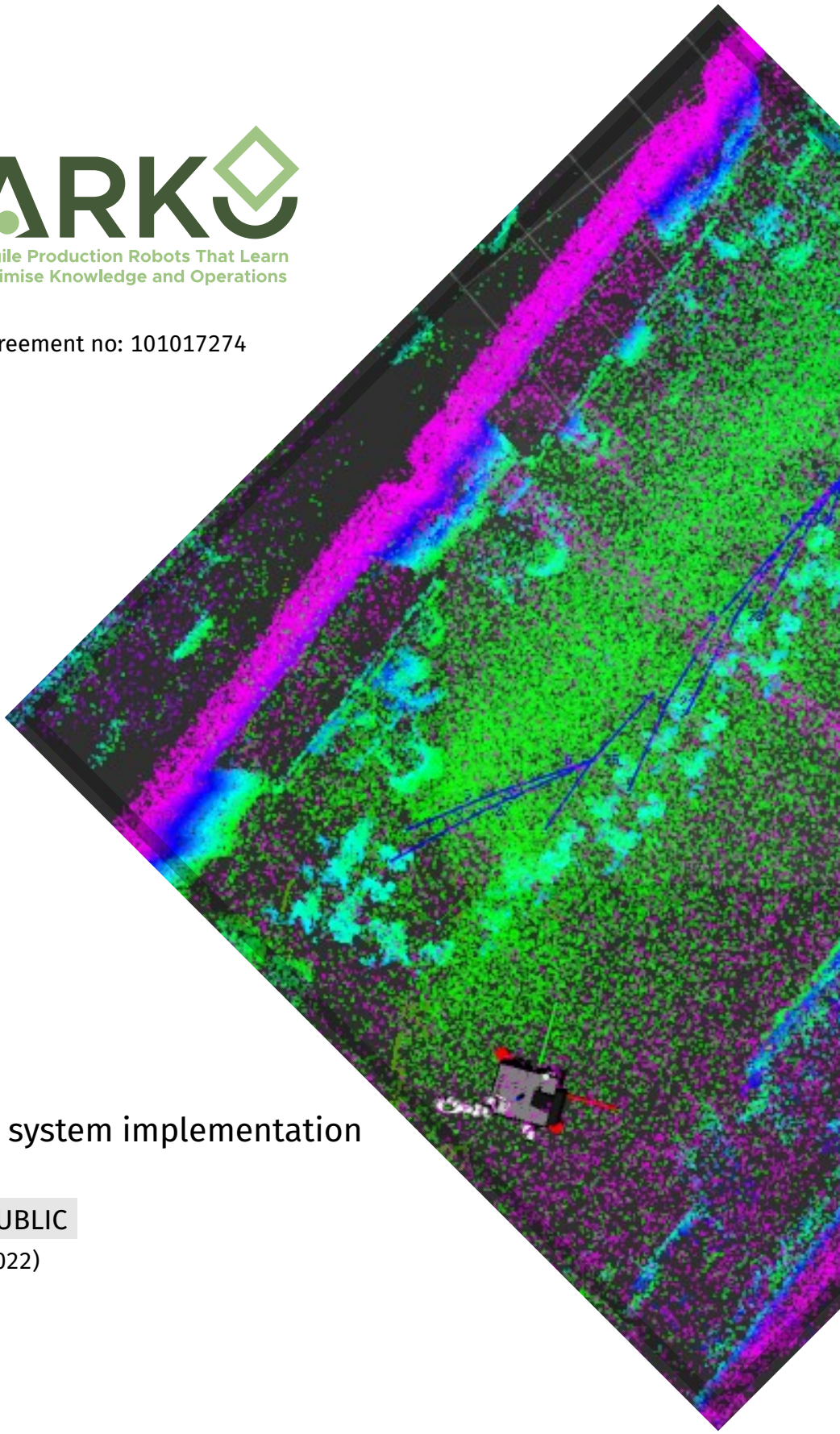
DARKO

Dynamic Agile Production Robots That Learn and Optimise Knowledge and Operations

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



DELIVERABLE 3.1

Prototype mapping system implementation

Dissemination Level: PUBLIC

Due date: month 17 (May 2022)

Deliverable type: Software

Lead beneficiary: ORU

1 Introduction

This deliverable consists of the initial implementation of mapping and localisation functionalities for tasks T3.1, T3.2 and T3.3 – including essential geometric mapping and localisation, baseline software for simultaneous localisation and mapping (SLAM) with heterogeneous maps, and constructing and serving several types of maps of dynamics to the DARKO system – as well as initial software for T3.4, in terms of assessment of localisation quality and localisation risk.

We provide links to the software repositories used by the implementation, and some examples of the running system.

Summary of progress towards DARKO's objectives

The overall objective of WP3 is to work towards *hands-off, failure-aware construction of rich map representations beyond mere geometry*. This goal directly maps to DARKO Objective 3 (efficient deployment) as well as Objective 2 (human-robot co-production). Concretely, we aim to address the following research questions. What are specific performance considerations w. r. t. surface representations and robust estimation techniques and how can learned features be efficiently used in scan matching for 3D mapping (T3.1). How can disparate sources of spatial information (e. g., unstructured human input) be combined in order to provide mutual benefits (T3.2)? How can we efficiently learn and represent, and exploit, flow and activity patterns in the mapped environment (T3.3)? How can a mapping system be endowed with introspective capabilities such that it becomes both failure-aware and failure-resilient (T3.4)?

The prototype mapping system software implementation available as of Month 17, which is reported in this deliverable, comprise baseline versions of software for general lidar-based maps for navigation (T3.1), merging lidar data with floor-plan line drawings (T3.2), creating and using several map of dynamics representations (T3.3), as well as introspective measures for scan alignment verification (failure awareness) and alignability maps for predicting localisation failures (failure resilience) (T3.4).

The porting efforts that have been done to make earlier work available to this project are summarised per task in the following sections. Most of the novel implementation work has been done in the software packages listed for T3.4 in Section 5.

Table 1: List of abbreviations

ACG	auto-complete graph
MoD	map of dynamics
NDT	normal distributions transform
NDT-MCL	normal distributions transform (NDT) Monte Carlo localisation
NDT-OM	NDT occupancy map
SLAM	simultaneous localisation and mapping

2 Mapping and localisation (T3.1)

For generating 3D and 2D geometric navigation maps, we employ a graph-based SLAM method [1, 2] based on NDT occupancy map (NDT-OM) submaps [3], implemented in the robust_mapping repository with dependencies as listed below. For localising in the NDT-OM map graph, we use a graph-aware version of NDT Monte Carlo localisation (NDT-MCL) [4] (graph_map), which has served as a baseline localisation system in several other projects as well. The output of the prototype mapping system in T3.1 is a 3D NDT-OM map (for localisation), a 3D point cloud map (mainly for visualisation), and a 2D grid map that can be readily integrated with standard planar motion planners.

Example output can be seen in Figure 1.

Please find below a rosinstall snippet listing the required repositories at the correct branch.

```
mapping/robust_mapping:
  type: git
  url: git@gitsvn-nt.oru.se:iliad/software/robust_mapping.git
  version: darko
mapping/graph_map_public:
  type: git
  url: https://gitsvn-nt.oru.se/software/graph_map_public.git
  version: merging
mapping/ndt_core:
  type: git
  url: git@gitsvn-nt.oru.se:software/ndt_core_public.git
  version: melodic-devel
mapping/ndt_tools:
  type: git
  url: git@gitsvn-nt.oru.se:software/ndt_tools_public.git
  version: melodic-devel
mapping/velodyne_pointcloud_oru:
  type: git
  url: https://github.com/dan11003/velodyne_pointcloud_oru.git
  version: master
```

The code base has been ported to ROS Melodic to match with the base navigation stack currently in use in DARKO. The porting efforts can be summarised as follows: Updating the code to use C++14 instead of C++11, workarounds for changes in liblz4-dev, bug fixes relating to namespaces for custom messages, updating dependencies for ROS Melodic and Noetic, updates in paths and topics to match the DARKO software architecture (D8.2).

Future versions of this implementation will include output from research in T3.1 that is novel to DARKO, including deeper studies of alternative surface representations and optimisation schemes for accurate mapping and localisation.

3 Heterogeneous map merging (T3.2)

We have ported the implementation of the auto-complete graph (ACG) [5] originally developed specifically together with an older version of the graph_map and ndt_core libraries from above, and ROS Kinetic, to work with the present version of these libraries, and ROS Melodic.

Figure 2 shows an example of how the software looks when it is running. Alternatively to the "uninformed" SLAM system from T3.1 (robust_mapping), when using ACG for SLAM, the robot is localised with NDT-MCL using a 2D NDT map extracted from the current shape

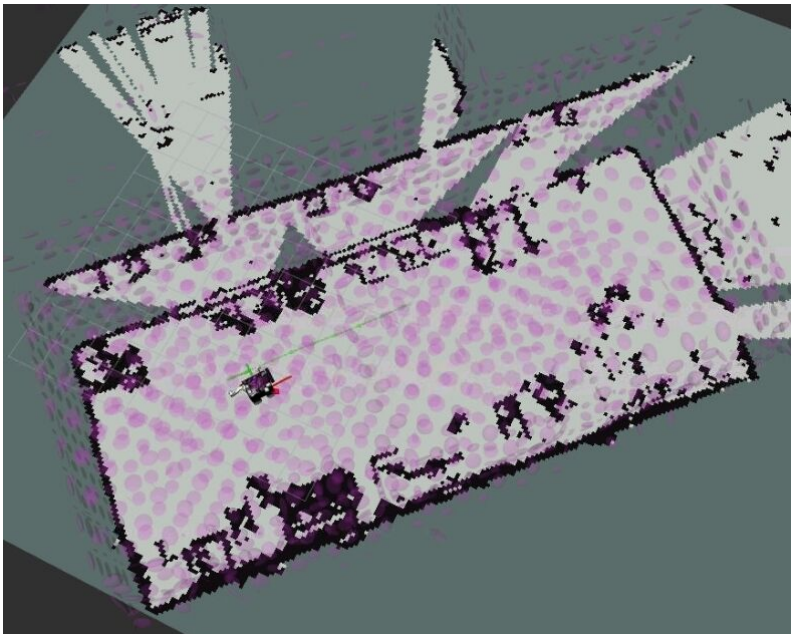


Figure 1: Geometric 2D and 3D maps from the ORU robotlab, created with the DARKO prototype mapping system. The 3D NDT-OM map is shown with purple ellipsoids, and a 2D map for planar navigation has been extracted from the underlying point cloud. A 3D point cloud map is also generated, but not shown in this picture.

of the prior map, while a sensor-based NDT submap is being created. Corners and walls extracted from the sensor-based map are associated to corners and walls in the prior with edges in the ACG, and the graph is repeatedly optimised with a set of two robust back-ends in tandem (a Huber kernel followed by dynamic covariance scaling [6]), in order to deal with large numbers of false corner associations.

The rosinstall snippet below lists the code base used – excluding external dependencies.

```
mapping/Auto_Complete_Graph:
  type: git
  url: https://github.com/MalcolmMielle/Auto_Complete_Graph.git
  version: master
```

The porting effort can be summarised as follows: Updating dependencies for ROS Melodic and Noetic, bug fixes relating to namespaces for custom messages, more parameterised launch files (less hard-coded configurations).

4 Maps of dynamics (T3.3)

The D3.1 prototype implementation also includes code for several types of *maps of dynamics* (MoDs), meant to enable the system to adapt to site-specific motion patters. We include CLiFF-map [7], STeF-map [8], and Gaussian mixture model trajectory maps (GMMT) based on Bennewitz et al. [9].

The code base includes software for learning these long-term map representations as well as map servers used to serve the maps to other parts of the DARKO system, such as global motion planning and long-term human motion prediction.

Figure 2: Demonstration of the auto-complete graph (ACG). The prior map (in this case extracted from a crude line drawing) is shown with black lines (corneras as brown squares). The current live sensor data (2D lidar) is shown with red points. The particle cloud, used in an MCL implementation that localises the sensor data against the prior map, is shown as red arrows.

Figure 3 shows an example of a CLiFF-map (T3.3) from this implementation, overlayed on top of a geometric map (from T3.1).

```
mods/mods_ros:
  type: git
  url: https://gitsvn-nt.oru.se/darko/software/mod_ros.git
  version: darko-integration
```

Concretely, the following has been done to make earlier work available to this project: Adding motion-intensity layer to the ROS implementation of CLiFF maps, updates in paths and topics to match the DARKO software architecture (D8.2)

5 Reliability-aware mapping (T3.4)

Although the initial software for reliability-aware localisation and mapping (T3.4) were originally not planned until D3.2 ("prototype implementation of map quality assessment and localisation assessment tool"), we do include a prototype implementations within this scope already in D3.1.

The first is a refactoring of the CorAl method for verification of scan alignment, which has also been published in Adolfsson et al. [10]. This refactoring includes, in addition to the previous 3D implementation, also extensions to 2D range data, including 2D radar scans.

We have also started working on *localisation risk maps*, the first implementation of which being an *alignability map* (see Figure 4). Prototype software for alignability mapping is available below.

```
mapping/alignability:
  type: git
  url: https://gitsvn-nt.oru.se/software/alignability.git
  version: master
```

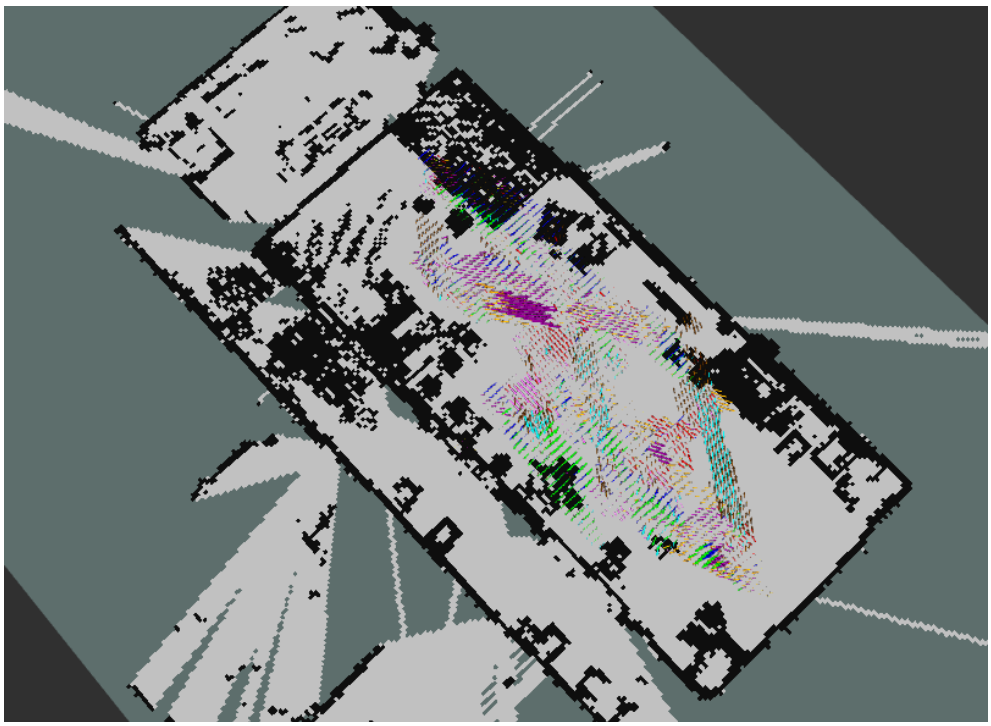


Figure 3: Example of CLiFF map (coloured arrows) overlayed on a 2D grid map.

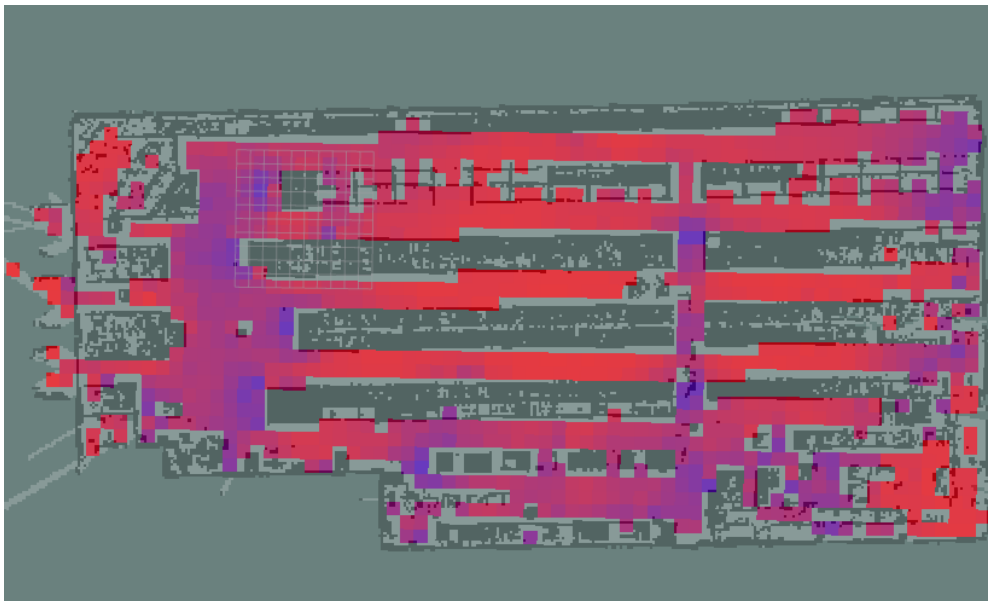


Figure 4: An *alignability map* from a real-world warehouse environment. Red regions indicate feature sparsity, which entails a higher risk of inaccurate localisation.

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] 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).

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

[7] 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* 2.2 (Apr. 2017), pp. 1093–1100.

[8] Sergi Molina, Grzegorz Cielniak, Tomáš Krajník, and Tom Duckett. “Modelling and Predicting Rhythmic Flow Patterns in Dynamic Environments”. In: *Towards Autonomous Robotic Systems*. Ed. by Manuel Giuliani, Tareq Assaf, and Maria Elena Giannaccini. Cham: Springer International Publishing, 2018, pp. 135–146.

[9] Maren Bennewitz, Wolfram Burgard, Grzegorz Cielniak, and Sebastian Thrun. “Learning Motion Patterns of People for Compliant Robot Motion”. In: *Int. Journal of Robotics Research (IJRR)* 24.1 (2005).

[10] 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.

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