Mobile Robots with Novel Environmental Sensors for Inspection of Disaster Sites with Low Visibility

Project start: January 1, 2015

Duration: 3.5 years

Deliverable 4.1

Technical specifications for multi-layered information storage model

Due date: month 36 (December 2017)

Lead beneficiary: LUH

Dissemination Level: PUBLIC

Main Authors:

Björn Zeise (BZ, LUH), Paul Fritsche (PF, LUH), Patrick Hemme (PH, LUH)

Version History:

0.1: Initial version, BZ, Nov 2017

0.2: Updated version, BZ, Dec 2017

0.3: Revised version, PF and PH, Dec 2017

Contents

Abstract

This document is a technical report within the scope of the Horizon 2020 project SmokeBot. SmokeBot's objective is to improve the application of mobile robots in disaster scenarios with low visibility conditions. A key element of the project is to develop perception algorithms that support the cooperation of humans and machines in search and rescue missions. This is done with the aid of a novel, multimodal sensor unit, which combines radar, gas, thermal and traditional vision sensors.

This report summarizes the results of Task 4.1 ("General Disaster Information Model"). Task 4.1 is part of Work Package 4 ("Situation Analysis"), which aims at robustly analyzing the current situation and predicting future progression. This is done by interpreting the data originating from the different information sources available on the robot. The objective of Task 4.1 in particular is to design the architecture of a multi-layered interconnected information structure that contains all the relevant information for decision making, which is part of other work packages.

1. Introduction and Purpose of this Document

SmokeBot aims at supporting first responders in situations that are too dangerous for human personnel. Using a robot equipped with a wide range of heterogeneous sensors can be of benefit to the rescue forces and the robot itself when it comes to self-preservation [6] [7]. In order to accomplish tasks such as hazard detection and mapping, all the information provided by the sensors needs to be processed and stored in a meaningful way using an information model.

The information model covered in this report is called General Disaster Information Model (GDIM). While most of the basic components of such an information model have already been evaluated [1] [2] [4], the focus on the domain of disaster robotics is still an active research domain [5].

The report is organized as follows:

- First, we show the results of the requirement analysis that we conducted after consulting all the partners in order to figure out GDIM's most important features.
- Afterwards, we present our approaches to the three key topics that GDIM focuses on: Data structure, time-variant behavior and data access optimization.
- Subsequently, we explain some details on GDIM's implementation.
- Finally, we conclude the report with a summary.

2. Requirement Analysis

In order to determine the necessary requirements, a workshop regarding GDIM was held. The discussion, especially with the end users, resulted in a number of requirements that the final system has to meet. Subsequently, we make a distinction between key features and additional requirements/specifications.

2.1 Key Features

Adaption of GDIM data when SLAM optimizes the map

Although no sensor data fusion is performed inside GDIM, a procedure that adapts multimodal maps stored in GDIM (temperature, smoke, gas, etc.) according to the SLAM process is one of the key elements of the GDIM framework. Therefore, maps inside GDIM are linked through transformations similar to the submap linking used in the graph SLAM approach presented in WP1. When the transformations are updated, GDIM is able to re-generate the multimodal maps.

GDIM contains relevant information for robot and operator

First of all, it is important for the human operator to have access to the most relevant information. That means that not all the sensor data has to be stored inside GDIM. Instead, processed data, i.e. representations of the environment containing distributions of obstacles, temperatures, gases, smoke, hazards, etc. will be recorded.

The discussion with the end users revealed that 3D environment information (such as partial 3D maps or 360-degree LiDAR scans) will most likely confuse the human operator. Therefore, the focus regarding information provided for the operator will be on 2D representation only. Apart from that, the robot itself will be able to access more complex information needed for self-preservation and navigation purposes by polling the respective processes.

Map data is stored in a multi-layered manner

Due to the variety of heterogeneous sensors used in SmokeBot, the resulting environmental maps will be stored in separate layers of a spatially aligned grid map. Special attention has to be paid on map synchronization and the handling of different (spatial) resolutions. In addition to metric grid maps, GDIM offers the ability to store semantic data (information about individual rooms of a building) as well.

Optimized data access

GDIM contains a comprehensive specification for data access. This includes fixed definitions of ROS topic names and types. Additionally, it is possible to query the status (temperature, gas concentration, general hazard level, etc.) of specific points in the map.

2.2 Additional Requirements

Support of the Robot Operating System (ROS)

Using ROS as a middleware and communication framework, GDIM provides interfaces for storing and accessing information. While data is not necessarily processed inside GDIM, common and customized message types will be used for inter-process communication.

No sensor data fusion inside GDIM

GDIM mainly provides functionalities for storing and accessing map data. The methods for creating those maps (SLAM, gas distribution and airflow mapping, signal strength mapping, hazard mapping) will be treated in separate tasks.

Basic time-variant behavior

In agreement with the consortium, we decided to include only a very basic time-variant behavior into GDIM. That means that map data can be logged with according timestamps if necessary.

Preparation for data visualization and robot-interface data transfer

Since some of the other tasks in the project strongly rely on information stored inside of GDIM, respective interfaces have to be provided. In particular, this is the case for Task 6.2 (“Data Visualization”) and Task 6.3 (“Robot-Interface Data Transfer”). Taking into account the use case scenario (and according limited wireless connection), GDIM considers adaptive functionalities for Wi-Fi bandwidth and data publishing/subscribing frequencies.

3. Layered Data Model

In mobile service robotics – and especially in the domain of search-and-rescue robotics – it is important for the robot and its operator to have as much information about the environment as possible. Layered information models serve the need for representing different kinds of environmental information. The different layers are connected to each other via a reference coordinate system. This way, information at specific points in the map can be retrieved from every individual layer. The general structure of layered environment models is depicted in Figure 1.

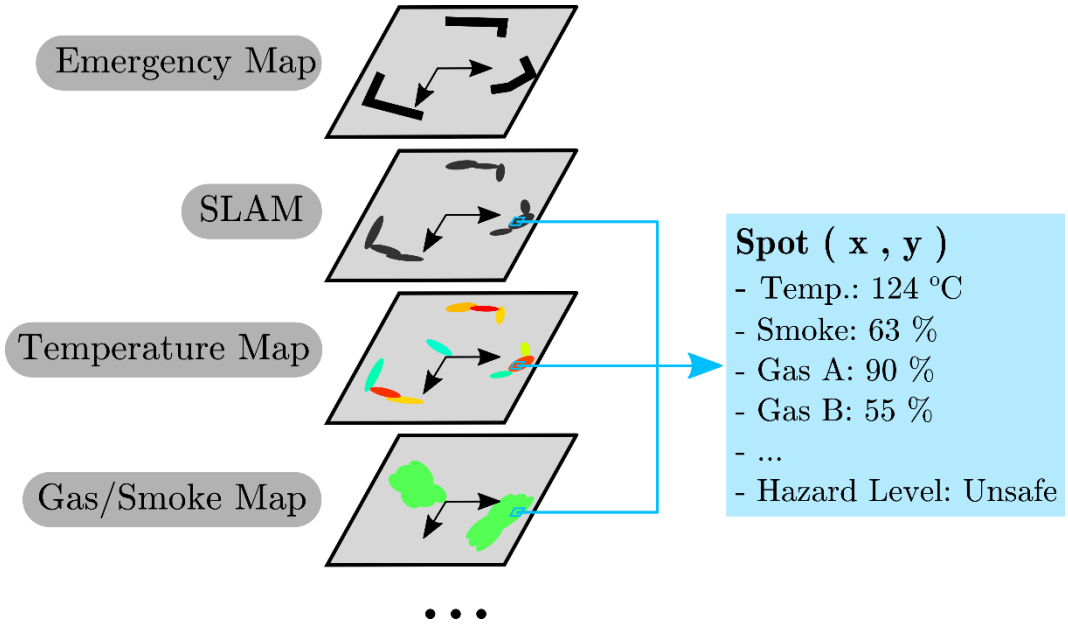

Figure 1: General concept of layered environment models: The illustration shows several different layers (prior known emergency map, obstacle map generated using SLAM during operation, temperature map using colors to indicate temperature values, gas/smoke map highlighting areas of high concentration). Each of those layers can have its own coordinate frame. Information for specific spots in the environment can be requested from the individual layers (blue box on the right).

The goal of using those layered models can be to represent:

- different levels of abstraction [8] using data from homogeneous sensor sources,
- different environmental properties (such as physical obstacles, colors, temperatures, gas/smoke concentrations, humidity, etc.) using data from heterogeneous sensors [11],
- time-variant environments, i.e. several instances of the same environment at different points in time [2],
- dynamic and static objects in the environment [12] or
- information provided by different robots in the same environment (i.e. from different points of view).

Of course, the list above makes no claims of being complete, but it gives an overview on the possibilities of layered environment maps.

3.1 Data Structure Definition

As depicted in the requirement analysis, GDIM is supposed to only contain information that is relevant to the robot and its operator. In order to get to know what information is needed during operation, we created a complete list of data produced in our system. The list and the decision whether to store or not to store the corresponding information is given in Table 1.

Table 1: Overview on data produced in the system (bluely marked data is part of GDIM).

Category	GDIM Storage	Type	Usage	Part of GDIM?
Raw Sensor Data (directly from sensors)	partially stored in GDIM	2D Range Data	2D radar	-
		3D Range Data	3D radar	-
			3D LiDAR	-
		Images	**Thermal images**	**X**
			RGB images	-
		In Situ Data	Gas sensor	-
			RF signal strength	-
			Temperature	-
			Humidity	-
Processed/Fused Data (generated using sensor data)	not stored in GDIM	2D Range Data	2D fused radar/LiDAR	-
			Laser-Radar-Ratio (LRR) data	-
		3D Range Data	3D fused radar/LiDAR	-
			3D data from thermal stereo vision	-
		Hazard Data	Low-level hazard (hotspot, gas source, no-go area, Wi-Fi dead spot)	-
			High-level hazard (explosion, flashover, backdraft, rollover)	-
Map Data (generated with the help of interpreting algorithms using sensor data and/or processed data)	mostly stored in GDIM	2D Metric Maps	**Obstacle submaps**	**X**
			Airflow map	-
			Gas distribution maps	**X**
			Smoke maps	**X**
			Emergency map	-
			RF strength map	**X**
			Temperature maps	**X**
			Hazard map	**X**
			Navigation cost map	**X**
			Teleop map	**X**
		3D Metric Maps	3D occupancy grid map	-
		2D Feature Maps	**Wi-Fi repeater location map**	**X**

		Geometric Maps	NDT-Graph map	-
		Submap Transformations	**TF**	**X**
		Semantic Data	**Per-Room information**	**X**
Navigation Data (generated using sensor data, processed data and/or map data)	not stored in GDIM	Data related to Navigation Tasks	Paths	-
			Control commands	-
			Odometry	-
			Robot pose	-
Status Data	not stored in GDIM	Robot-related	Battery state	-
			CPU load	-
		Network-related	Wi-Fi repeater status	-
			Wi-Fi bandwidth used	-

Besides the definition of what needs to be stored inside GDIM, the question of how to store and access data on the robot needed to be answered. Since we build our system upon the ROS framework, we decided to use as much commonly used data/message types as possible. Table 2 lists all the relevant data that needs to be part of GDIM and the according ROS data types.

Table 2: GDIM ROS message/service types and units/value ranges (values marked with a star depict parameters that can be adjusted depending on the scenario).

Kind of Data	ROS Message/Service Type	Units, Value Ranges and Comments
Thermal images	sensor_msgs/Image	RGB values according to chosen color map
Obstacle submaps	nav_msgs/OccupancyGrid	0 (free)…100 (occupied); -1 (unknown)
Gas distribution maps (multiple, one for each kind of gas)	grid_map_msgs/GridMap	Several layers for gas components and different kinds of gases; 0 (minGas*)…1 (maxGas*); -1 (unknown)
Smoke maps (in situ, LRR, fused)	nav_msgs/OccupancyGrid	0 (no smoke)…100 (maxSmoke*); -1 (unknown)
RF strength map	nav_msgs/OccupancyGrid	0 (no signal)…100 (good reception); -1 (unknown)
Temperature maps (in situ, remotely, fused)	nav_msgs/OccupancyGrid	0 (minTemp*)…100 (maxTemp*); -1 (unknown)
Hazard map	nav_msgs/OccupancyGrid	0 (harmless)…100 (unsafe); -1 (unknown)
Navigation cost map	nav_msgs/OccupancyGrid	0 (safe to traverse)…100 (untraversable due to obstacles/hazards); -1 (unknown)
Teleop map	nav_msgs/OccupancyGrid	0 (no limitations)…100 (untraversable due to low-visibility/connectivity); -1 (unknown)
Wi-Fi repeater location map	geometry_msgs/PolygonStamped	List of points that denote repeater dropping locations
TF	geometry_msgs/Transform	Array of transformations containing translation

		and rotation between subsequent submaps
Rooms	std_msgs/UInt8	Number of recognized rooms
Per-Room information	Custom message type	Information for each room: id, location, size, avg. temperature, avg. present gases, avg. smoke value, avg. signal strength, max. hazard level
Spot-wise information	Custom message type	Information for a specific spot in the environment: temperature, present gases, smoke value, signal strength, hazard level

3.2 Synchronization and Handling of Different Resolutions

Temporal synchronization of GDIM data is achieved applying the commonly used *message_filters* package [9]. Using this package, an approximate time policy makes sure that the timestamps of relevant data such as maps, robot poses and according transformations are matched.

GDIM makes no regulations for spatial resolutions of the maps provided by other packages. Since usually all the map metadata comes together with the map itself, GDIM can store the maps without further knowledge.

3.3 Submap Adaption due to Graph Optimization

One of the main problems of building environmental maps using SLAM occurs when the robot comes back to a place that was visited earlier. The robots then has to “close the loop”, which means that the whole map has to be regenerated taking into account every observation made earlier. In T1.5/D1.8, mapping is performed using a graph-based SLAM approach. In graph-based SLAM, the graph containing several nodes, which are connected by links (transformations), can be optimized depending on the current situation. During optimization, the transformations between the nodes are updated. Accordingly, the whole map changes, since every node is coupled with a set of local submaps. Submaps are partial views of the environment taking into account only a limited number of observations. Each submap is linked to one of the graph nodes and therefore has its own local coordinate frame. The initialization of a new submap is performed using a criterion based on the robot’s travelled distance.

In SmokeBot, loop closure is not only a problem of SLAM, but also of GDIM. GDIM stores multiple map layers containing additional environmental and semantic information. Those map layers are built upon the maps, transformations and robot poses provided by the SLAM. During operation, this data is updated continuously. Creating static map layers that do not take into account those updates would lead to inconsistency over time. Figure 2 illustrates this inconsistency (which is tackled using the submap adaption approach): While the SLAM updates its map and the robot poses continuously, the other mapping methods (exemplarily depicted for temperature mapping in this case) relying only on the robot’s current pose in the world coordinate frame would generate faulty maps. This is due to the fact that during

graph optimization the updated robot pose can possibly “jump” to an unexpected value compared to the values provided just a short time before. Those faulty maps would then contain the same features a couple of times (bottom right in the illustration). With the help of GDIM, map building methods (such as temperature, smoke, gas and signal strength mapping) can avoid those inconsistencies by directly using map and transformation updates originating from the SLAM.

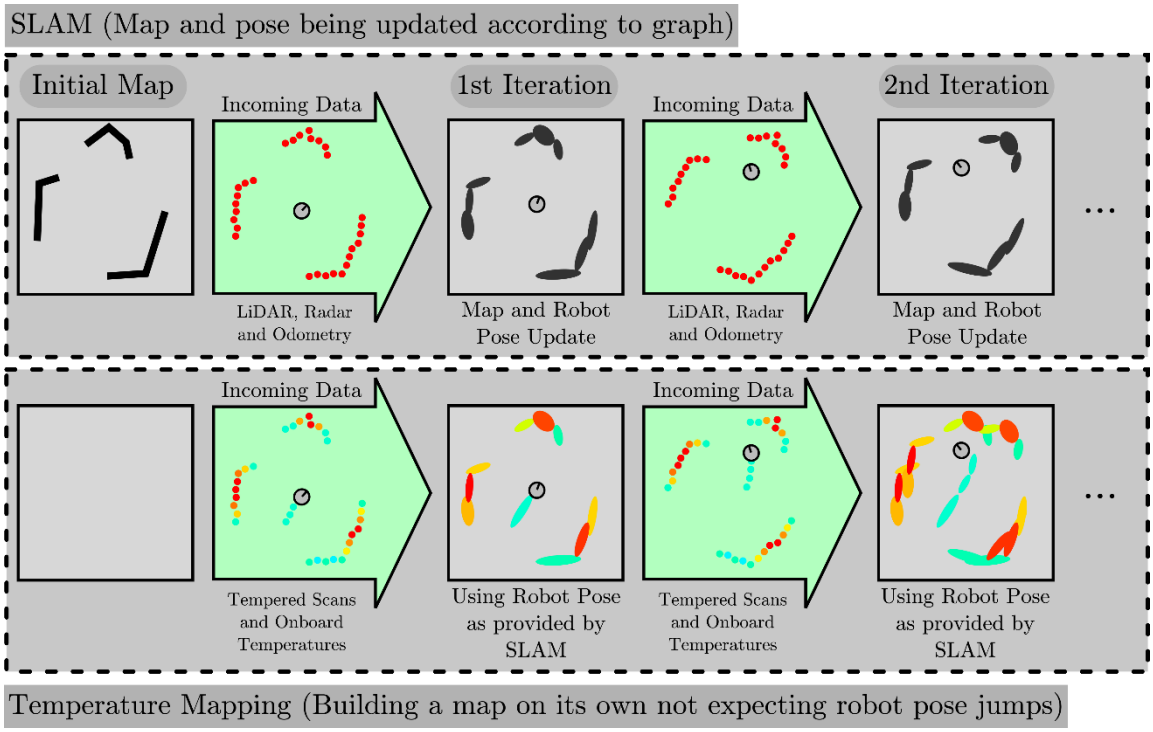

Figure 2: Inconsistency in case of static map generation: The box on the top roughly depicts the obstacle map generation using the SLAM approach. Starting with an initial map, a couple of map and pose iterations are estimated using incoming data. During map and pose update, the robot pose can possibly change in a volatile manner. If we would use a static map approach for the temperature mapping (as depicted in the box on the bottom), such a “pose jump” could lead to faulty double detection of the same features. That is why in GDIM, we propose to use a submap adaption approach that generates submaps for each modality (temperature, gas, smoke, signal strength) and connects those submaps via transformations provided by the SLAM (just like the SLAM method itself does).

Similar to the graph-based SLAM that generates submaps depending on distance travelled, a submap adaption method is part of GDIM. All the maps containing additional environmental information (such as temperature, smoke, gas and signal strength distributions) can be built in such a submap manner. That means that each of those submaps has its own local coordinate system (map origin). The individual submaps are linked with each other by spatial transformations. Those transformations are the same as for the submaps used in the SLAM approach. Figure 3 illustrates this concept.

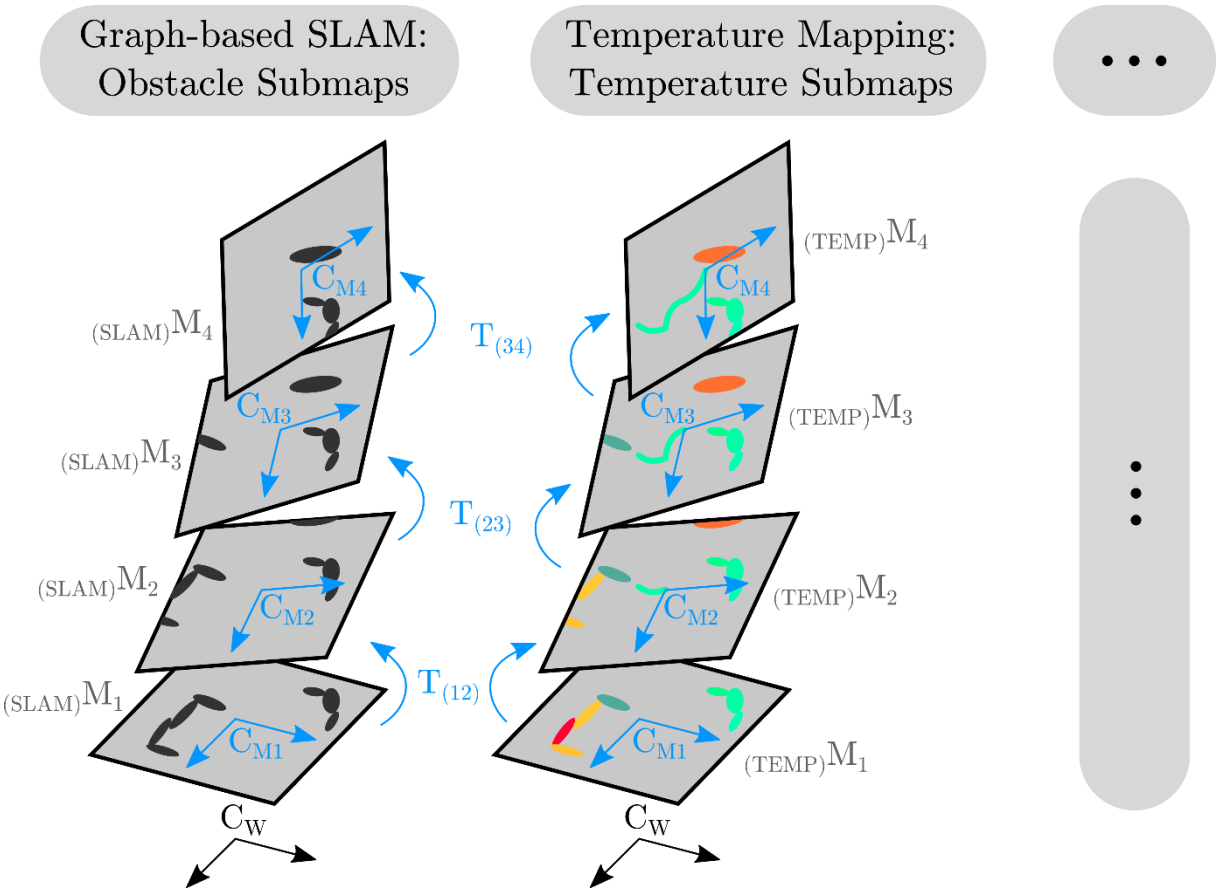

Figure 3: Submap concept in the case of obstacle and temperature maps: The left column depicts a set of obstacle submaps ${}_{(\mathrm{SLAM})}M_i$ with origin C_{Mi} provided by the SLAM method. Each of those submaps is linked with its predecessor and successor via transformations T_{ij}. The criterion for generating a new submap is the distance travelled by the robot. Similar to the SLAM method, temperature mapping (center column; map elements colored according to temperature values) is generating a set of temperature submaps ${}_{(\mathrm{TEMP})}M_i$ with origin C_{Mi} that are also linked via transformations T_{ij}. Using those submaps and the latest transformations, an overall obstacle/temperature map can be calculated by overlaying all submaps.

The SLAM method builds a graph consisting of a set of obstacle maps ${}_{(\mathrm{SLAM})}M_i$. Each of those maps is related to its own coordinate system C_{Mi}. The coordinate systems are linked via geometric transformations T_{ij}. The illustration shows that – with respect to the world coordinate frame C_W – the detected obstacles have almost the same location in every map, although the maps are shifting/rotating. A similar behavior is used to generate other kinds of maps. Figure 3 exemplarily depicts this for temperature maps. Since temperature mapping uses the same transformations and submap coordinate systems as the SLAM method, map updates and loop closures can conveniently be handled. An overall map of each modality (obstacles, temperatures, gas, smoke, etc.) can be calculated by overlaying all submaps using the latest transformations provided by the SLAM method.

4. Time-variant Behavior

In discussion with the consortium, we decided only to implement a very basic time-variant behavior. This is in contrast to more sophisticated approaches as presented e.g. in [2] and [3]. Especially the end user remarked that such a complex time-variant information storage and handling would rather confuse the robot operator than help understanding the environment.

We made the decision, that most of the volatile data produced by the system will not be saved for a longer period. Instead of having a complex solution for time-variance, a basic data/map logging functionality has been implemented in GDIM. Using this, it is possible to enable/disable periodic data logging as well as to take “snapshots” of the current situation. The concepts of data logging and snapshots are explained below.

4.1 Periodic Data Logging

Periodic data logging is configurable via the ROS *rqt* framework (for debug usage) and services (for compatibility with remote control). Regarding the debug usage, the logging feature can be enabled and disabled by the operator using *rqt_reconfigure* package [10]. Additionally, the repetition rate (in Hz) at which data will be saved can be adjusted. The following information will be saved to the hard disk in every logging interval:

- Latest iteration of NDT-graph (from SLAM)
- Latest accumulated/global maps of the following:
  - SLAM (obstacle map)
  - Temperature mapping
  - Smoke mapping
  - Gas mapping
  - Signal strength mapping
- Latest submaps of the aforementioned
- Geometric transformation between the latest submaps to the previous submap
- Semantic data as “per-room-information” (text file)
- Latest thermal image (false-colored)

4.2 Snapshots

The data saved using the snapshot functionality is the same as the above listed except for submaps and their transformations. This is due to the fact that – without the context (previous or subsequent submaps) – the submaps are not of much value.

5. Data Access

GDIM is the central information unit that stores processed data (especially maps) of multiple kinds. As depicted in Figure 4, the main information contributors are SLAM (WP1), Temperature Mapping (WP3), RF Signal Strength Mapping (WP5), Gas Mapping (WP2) and Hazard Mapping (WP4). GDIM stores the latest map versions provided by those processes. Depending on the data logging properties (see previous section), the information can be saved periodically or once-only. Processes such as Visualization (WP6) and Navigation/Self-Preservation (WP5) can access the data via uniquely defined topics and services.

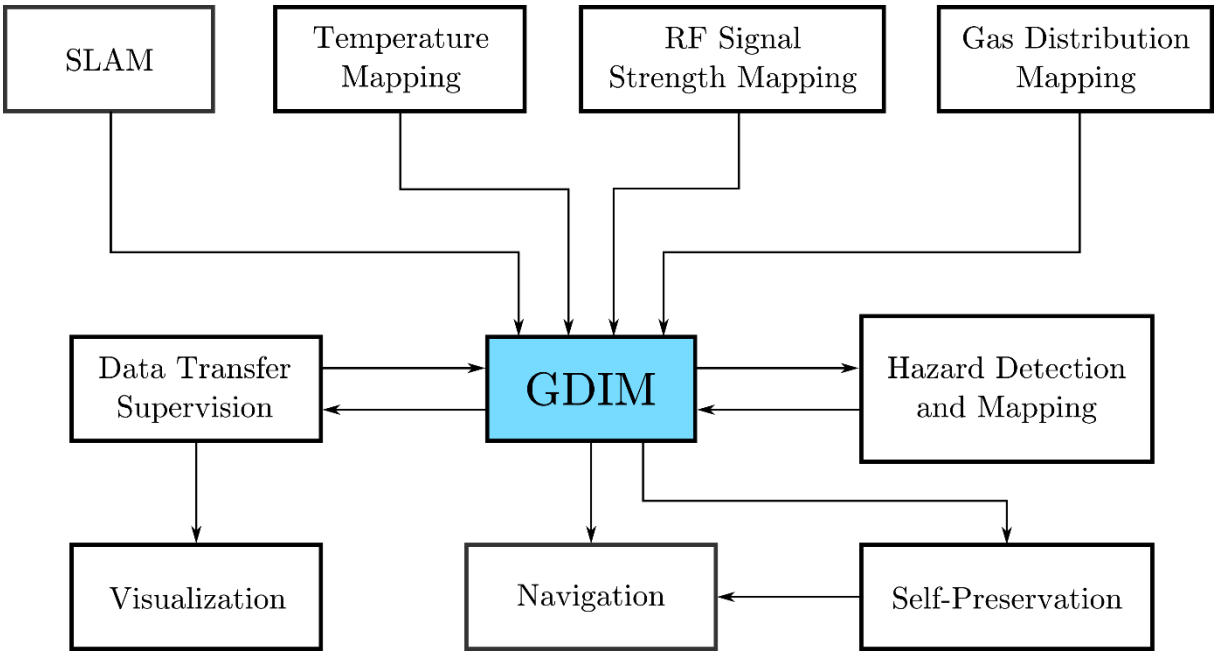

Figure 4: GDIM data flow

Since GDIM uses the ROS framework for communication, data is transferred to and from GDIM via topics and services. Table 3 gives an overview on the information stored in GDIM and according topic/service names. For data sent to GDIM, the namespace is */gdim/input/*, while for outgoing data it is */gdim/output/*. The topic/service names listed in the table are the default ones and can be remapped using the corresponding ROS functionality.

Table 3: GDIM topics and services

Kind of Data	Data Source	Type	GDIM (input) /gdim/input/…	GDIM (output) /gdim/output/…
Thermal images	WP3	Topic	thermal_image	thermal_image
Obstacle maps	WP1	Topic	slam_submap	slam_map
Gas distribution maps	WP2	Topic	gas_A_submap_1	gas_A_map_1

			gas_A_submap_2 gas_A_submap_3 gas_B_submap_1 gas_B_submap_2 gas_B_submap_3 …	gas_A_map_2 gas_A_map_3 gas_B_map_1 gas_B_map_2 gas_B_map_3 …
Smoke maps	WP1	Topic	smoke_submap_remote smoke_submap_insitu smoke_submap_fused	smoke_map_remote smoke_map_insitu smoke_map_fused
RF strength map	WP5	Topic	rf_signal_submap	rf_signal_map
Temperature maps	WP3	Topic	temp_submap_remote temp_submap_insitu temp_submap_fused	temp_map_remote temp_map_insitu temp_map_fused
Hazard map	WP4	Topic	hazard_map	hazard_map
Navigation cost map	WP4	Topic	nav_cost_map	nav_cost_map
Teleop map	WP4	Topic	teleop_map	teleop_map
Wi-Fi repeater location map	WP7	Topic	repeater_map	repeater_map
TF	WP1	Topic	submap_transform	submap_transform
Rooms	WP1/7	Topic (in) Service (out)	rooms	rooms
Per-Room information	WP1/7	Topic (in) Service (out)	room_information	room_information
Spot-wise information	WP4	Service (out)	(generated in GDIM)	spot_information

GDIM does not process all the incoming information. Data such as a thermal image can be subscribed from the GDIM node, but it is not changed and only passed through. In contrast, most of the map information entering GDIM is used to create modality (temperature, gas, smoke, etc.) maps that take into account graph optimization performed by the SLAM algorithm (see Section 3.3).

5.1 Access to Specific Information

GDIM provides services for polling information about specific rooms and spots/positions in the environment. Polling is triggered by visualization and human-robot interface, respectively. Using the services *rooms*, *room_information* and *spot_information*, GDIM outputs the following information:

Per-Room Information

Service name:	/gdim/output/rooms
Request:	Empty (std_srvs/Empty)
Response:	Total number of rooms detected in the environment (std_msgs/Int8)
Service name:	/gdim/output/room_information
Request:	Room number (std_msgs/Int8)
Response:	Custom Message Type containing: - Avg. temperature (std_msgs/Float32) - Avg. present gases (std_msgs/Float32[]) - Avg. smoke value (std_msgs/Int8) - Avg. signal strength (std_msgs/Int8) - Max. hazard level for room (std_msgs/Int8)

Position Information

Service name:	/gdim/output/spot_information
Request:	Spot location (geometry_msgs/Point32)
Response:	Custom Message Type containing: - Temperature (std_msgs/Float32) - Present gases (std_msgs/Float32[]) - Smoke value (std_msgs/Int8) - Signal strength (std_msgs/Int8) - Hazard level (std_msgs/Int8)

5.2 Points of Intersection with Human-Robot Interface

In order to meet the requirements for WP6 (“Human-Robot Interface”), GDIM provides a functionality to dynamically adjust the publishing rate of outgoing topics that are intended for visualization. The adjustment can be performed by polling a service in the following way:

Service name:	/gdim/output/pub_rate
Request:	Custom Message Type containing: - Topic to adjust (std_msgs/String) - Desired frame rate (std_msgs/Float32)
Response:	Custom Message Type containing: - Previous (old) frame rate (std_msgs/Float32) - Current (new) frame rate (std_msgs/Float32)

In real use case scenarios, the adjustment will be controlled by a bandwidth supervision tool that is developed within the scope of T6.3 (“Robot-Interface Data Transfer”). In moments with good Wi-Fi connection (and high bandwidth available), the publishing rate will be higher than for moments with a bad connection.

6. Implementation Notes

In this section, we outline the implementation of GDIM's key features. Since the code base of GDIM is subject to change until the end of the project, detailed implementation notes and comments on the usage of individual ROS nodes can be found in the code documentation.

6.1 Layered Data Structure

The internal storage of maps inside GDIM is performed using the *grid_map* package [13]. Each submap is stored as an individual layer in a grid map. Sets of subsequent submaps form an array of grid maps. The grid map layers are names according to the topic/service names listed in Table 3.

GDIM is also capable to store non-map data such as the number of rooms and information regarding those rooms. In most cases, such information is also related to the submap. For that reason, this kind of information is stored in arrays similar to the ones for sets of subsequent submaps. Those arrays make use of the message data formats specified in the previous sections.

6.2 Process Overview

GDIM is a framework that contains several functionalities in different decentralized modules. It provides the following functionalities:

- Storage of map data using layered grid maps
- Storage of non-map data using arrays
- Adaption of map data in case of (SLAM) graph optimization (Section 3.3)
- Publication rate adjustment (Section 5.2)
- Data logging (Section 4)
- Data access (Section 5.1)

In order to generate submaps of different modalities (temperature, gas, smoke, etc.), T4.2 provides submapping methods. Essentially, the submapping node is not part of GDIM, although heavily used by it.

6.3 Submap Adaption

The submap adaption (also described in Section 3.3) merges several submaps (of different modalities) to one fused map. Basically, all the submaps are overlaid according to their transformations. In order to generate one final map, a decision about values of map cells has to be made, since for every cell of the final map, several submap values might be available. Currently, there are two modes implemented in the framework:

- **Maximum cell value fusion:**
  The final value of the fused map cell is the maximum value of this cell over all submaps.
- **Mean cell value fusion:**
  The final value of the fused map cell is the mean value of all the submap cell values.

Depending on the modality, further fusion approaches for individual map cell values can be implemented in GDIM.

7. Conclusion

In this document, we described the technical specifications for the multi-layered information storage information model (GDIM) that is used in the SmokeBot project. In detail, we presented the key features of GDIM and explained their practical implementation. Besides this, we defined several specifications (topics and services for communication, message types and value ranges) that are used in the other work packages.

Since the work on GDIM is ongoing until the end of the project, some details explained above are subject to change. Nevertheless, the key features and requirements listed in this report can be regarded as fixed.

References

[1] Fiosins, M.; Zeise, B.; Gernert, B.; Schildt, S.; Fritsche, P.; Manesh, R.; Müller, J.; Wagner, B. and Wolf, L.: **dCIM: An Agent-Based Distributed Common Information Model for Teams of Mobile Robots**, in 'Proc. Autonomous Agents and Multi-Agent Systems at Scale (Workshop)', 2015.

[2] Hentschel, M. and Wagner, B.: **An Adaptive Memory Model for Long-Term Navigation of Autonomous Mobile Robots**, 'Journal of Robotics', vol. 2011, Article ID 506245, 2011.

[3] Biber, P. and Duckett, T.: **Dynamic maps for long-term operation of mobile service robots**, in 'Proc. Robotics: Science and Systems (RSS)', pp. 17-24, Cambridge, USA, 2005.

[4] Zender, H.; Martínez Mozos, O.; Jensfelt, P.; Kruijff, G.-J.M. and Burgard, W.: **Conceptual spatial representations for indoor mobile robots**, in 'Robotics and Autonomous Systems', vol. 56, issue 6, pp. 493-502, 2008.

[5] Kohlbrecher, S.; Romay, A.; Stumpf, A.; Gupta, A.; von Stryk, O.; Bacim, F.; Bowman, D. A.; Goins, A.; Balasubramanian, R. and Conner, D. C.: **Human-robot Teaming for Rescue Missions: Team ViGIR's Approach to the 2013 DARPA Robotics Challenge Trials**. 'J. Field Robotics', 32: 352-377, 2015.

[6] Murphy, R. R.: **Trial by fire [rescue robots]**, in 'IEEE Robotics & Automation Magazine', vol. 11, no. 3, pp. 50-61, 2004.

[7] Tadokoro, S.: **Rescue robotics: DDT project on robots and systems for urban search and rescue**, Springer London, 2009.

[8] Mozos, O. M.; Jensfelt, P.; Zender, H.; Kruijff, G. J. M. and Burgard, W.: **From labels to semantics: An integrated system for conceptual spatial representations of indoor environments for mobile robots**, in 'ICRA Workshop: Semantic Information in Robotics', 2007.

[9] http://wiki.ros.org/message_filters (as of 2017/12/12)

[10] http://wiki.ros.org/rqt_reconfigure (as of 2017/12/12)

[11] Kleinschmidt, S. P. and Wagner, B.: **GPU-accelerated Multi-sensor 3D Mapping for Remote Control of Mobile Robots using Virtual Reality**, in 'Proc. 13th International Conference on Informatics in Control, Automation and Robotics (ICINCO)', vol. 2, pp. 19-29, Lisbon, Portugal, 2016.

[12] Wolf, D. and Sukhatme, G. S.: **Online simultaneous localization and mapping in dynamic environments**, in 'Proc. IEEE International Conference on Robotics and Automation (ICRA)', vol. 2, pp. 1301-1307, 2004.

[13] http://wiki.ros.org/grid_map (as of 2017/12/18)