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

Grant agreement no: 645101

Project start: January 1, 2015

Duration: 3.5 years

# Deliverable 6.2

# Software Toolkit – Data Visualization

Due date: month 40 (April 2018)

Lead beneficiary: ORU

Dissemination Level: Public

**Main Authors:**

Malcolm Mielle (ORU)
Martin Magnusson (ORU)

# Contents

## 1. Introduction

This document is a technical report within the scope of the Horizon 2020 project, SmokeBot. In search and rescue scenarios, a robot needs to perform navigation in unknown environments. The SmokeBot project focuses on civil robots supporting fire brigades and aims to enhance the performance of robots under harsh conditions, for example: smoke, dust or fog.

This report provides an overview of the software toolkit developed in Task 6.2, "Data Visualization". Task 6.2 is part of Work Package 6, "Human robot interface", and aims at giving proper visualization tools for the human operator.

The software extends the capabilities of Taurob controller by means of a setup of software plugins, and leverages the design of the GDIM as a layered map architecture to create an intuitive visualization of the environment. The interface takes as input the GDIM and fuse its different layers to present them to the user in a easily understandable image view, enabling the visualization of both sensor data such as thermal measurements and the different maps used by the robot (for example, the prior emergency map and the map built using the sensor measurements). The colors and display modes of the different layers can be adjusted by modifying a configuration file. Also, the interface offers help in the navigation and control of the robot by displaying a video feed of the camera, by signaling risk of collisions with the environment, and by displaying visual warnings concerning the environment and the status of the robot itself. Finally, buttons are used to give direct commands to the robot. Figure 1 shows an overview of the extended interface with all the plugin modules, which are further described in the remainder of this document.

The plugins have been developed by both Örebro and Hannover University.

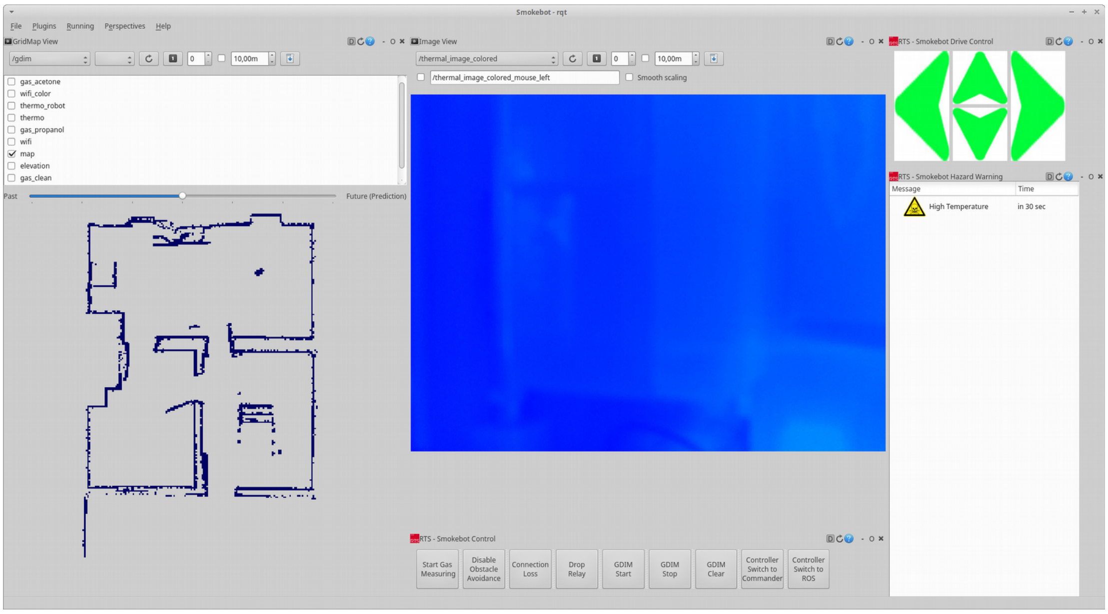


*Figure 1: Interface with all plugins*

## 2. Grid map visualization

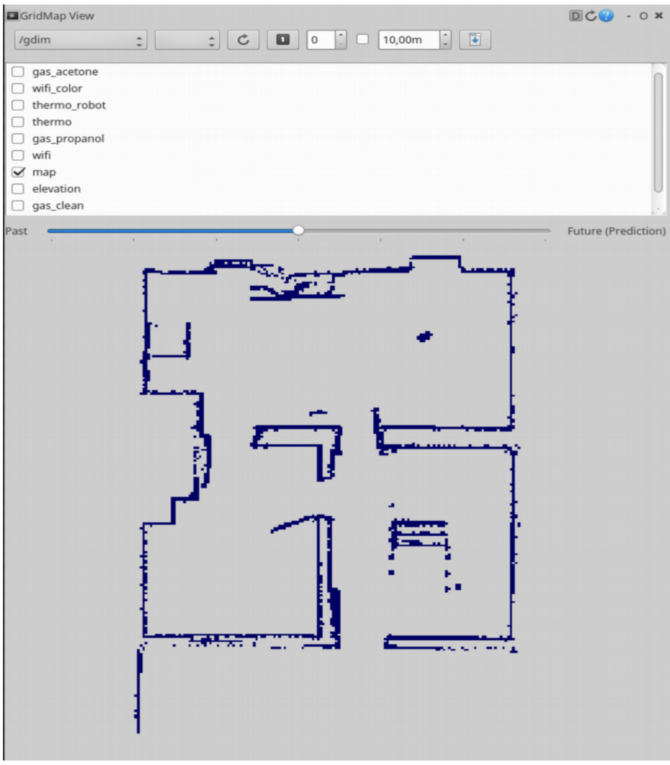


*Figure 2: Grid map plugin showing a map of the environment explored by the robot, overlayed on an emergency map (solid lines) used as a prior.*

The GDIM is a layered representation of the environment where each layer represents a modality: the robot built map, emergency map, gas sources, WIFI coverage, etc. However, this layer representation by itself is not practical. The format is heavy and layers cannot always be used independently of each others. Indeed, WIFI coverage or gas sources without the robot built map of walls and other obstacles are not useful, and exploration and navigation without the emergency map is diificult.

The grid map interface uses the different layers returned by the GDIM to create an intuitive map representation displaying information from the different layers.

## 2.a Layers of the GDIM

The user interface displays a list of the name of all layers provided by the GDIM. The user can select multiple layers in the list and each layer is fused in one representation. The parameters determining the color of each layer in the final image are determined using a yaml configuration file. Here is an example of such parameters for a given map:

```
map:
        min_value: 0.0
        max_value: 100.0
        min_color_r: 255
        min_color_g: 255
        min_color_b: 255
        max_color_r: 0
        max_color_g: 0
        max_color_b: 0
```

Where min_value and max_value correspond to the possible min and max value in a grid_map layer. The value of each cell is then mapped onto the min and max color values.

During the fusing process, each layer is converted into an openCV image using the parameters set into the yaml file. Then, all layer images are fused in one: for each pixel, if the same pixel in the final image already has a value, its new value is the average of its current value and of that of the input pixel. Otherwise the input pixel value is directly used. The alpha channel is set to the least transparent alpha between the pixel in the final image and the one to be fused in. This way, where layers could be masking one another, transparency is used to visualize them both at the same time in a meaningful and helpful way.

## 2.b Visualization in time

Sometimes, firemen need to be able to either visualize past data, for example to see where a gas source used to be, or a prediction of the future state of the maps, to see if a flash-over is likely to happen or not. The grid_map interface enables the user to visualize the state of the map at different points in time. Using a slider, which can be seen in Figure 2 above the grid map image, the user can choose to either go back in time and see past states of the map or to use the GDIM prediction node and visualize possible future states of the map.

## 2.c Robot model

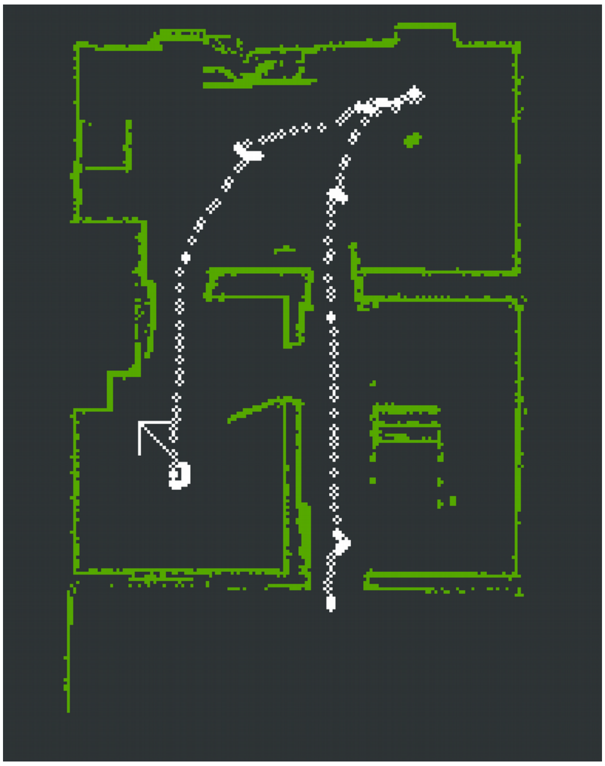

*Figure 3: Robot map in green and robot odometry and heading in white*

The robot model is displayed on the layer image by displaying the odometry, the pose, and the heading of the robot. The odometry is displayed by points in order not to hinder the user's view of the environment, and a robot representation is used to show the pose and orientation of the robot. In order not to create problems when trying to teleoperate the robot with past or future maps, the robot model displayed is always the current one, and not a past or predicted one.

## 3. Navigation and teleoperation

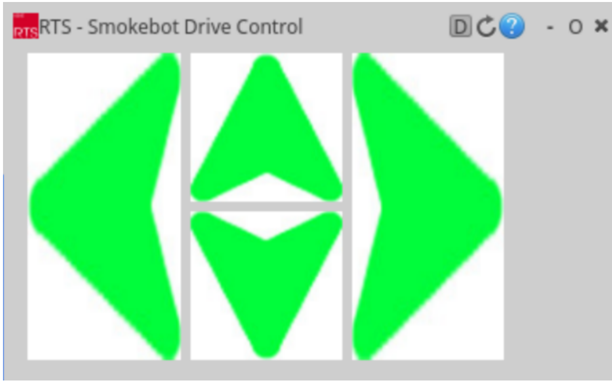


*Figure 4: Collision arrows are green when no obstacles are detected. They become red otherwise.*

Teleoperating the robot is a complicated task. It's even more complicated when smoke is blocking the camera views. While the robot model visible on the map enables the user to teleoperate the robot, there is a need for more help in controlling the robot. Hence, a plugin to visualize risks of collision and to help suggesting directions to take while teleoperating was developed.

The plugin displays four green arrows indicating the four directions around the robot: front, back, left, and right. When the robot is at risk of colliding with an obstacle, the arrow pointing toward the obstacle becomes red. Thus, each arrow is warning the user about risks of collision and indicate the direction from which the robot needs to move away.

## 4. Image view

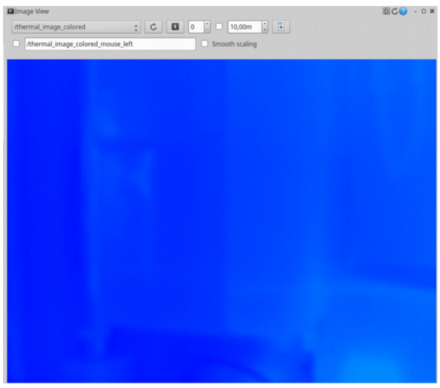

*Figure 5: Thermal camera image*

The image view enables the user to visualize camera feeds when needed. This can be either normal cameras or thermal cameras to use when smoke is present or to detect hotspots. It should be noted that a temperature map can also be visualized in the grid map interface. The user only need to select the camera for the feed to be displayed.

# 5. Control panel


*Figure 6: Control panel with all buttons.*

During the exploration, the robot can take a number of actions that need to be triggered by the user. For example, recording of data, dropping relays, or switching between different modes of command. For this purpose, we developed the control panel. The buttons on the control panel enable the user to:

- Start gas measuring: if pressed, the robot will start to do a gas measurement used to find and identify gas sources.
- Enable/Disable obstacle avoidance: if pressed, the obstacle avoidance is either enabled or disabled. If on, the robot will not allow the user to drive through obstacle and walls.
- Connection loss: simulates a connection loss and triggers the safety mechanism (for testing purposes)
- Drop relay: drops a relay used by the robot to keep the Wifi connection alive on long explorations.
- Start GDIM: triggers GDIM to start collecting data and building a layer model.
- Stop GDIM: turns GDIM data collection off.
- Clear GDIM: resets the current GDIM model built.
- Controller switch to commander: if pressed, the robot can now be teleoperated using Taurob commander.
- Controller switch to ROS: if pressed, the robot can now be controlled using the ROS interface.

## 6. Hazard warnings

Sometimes, the user will need to visualize possible dangers quickly. For example, if the temperature quickly rises, it's important that the user be made aware of this without delays. The user also needs to be able to know information about the internal state of the robot, for example if the CPU is overheating. Hence, the hazard warning panel is used to warn the user about possible risks and dangers, and gives more graphical information about the states of the sensors.

Six types of warning can be given: high temperature, toxic environment, risk of explosion, zones without Wifi, internal problem, and battery low. Each warning is associated with an icon for fast visual recognition. All warnings are raised by the hazard prediction and detection nodes.

- The high temperature warning is given when the temperature around the robot reaching a dangerously high level.

- The toxic environment warning is given when gas is around the robot.

- A risk of explosion is raised when there is gas and hot temperature create the risk for such explosion or flash-over or other type of danger.

- Warns the user that the robot is in a zone with weak Wifi signal. This is determined using the coverage map.

- Internal problem is a warning that something wrong in the robot: CPU temperature is too high, CPU load is too high, RAM usage is too high…

- The low battery warning is raised when the battery dip under 24 volts low. The battery becomes critical if it is under 23.5 volts.