ETH zürich

Tutorial on Path Planning

ETH Robotics Summer School

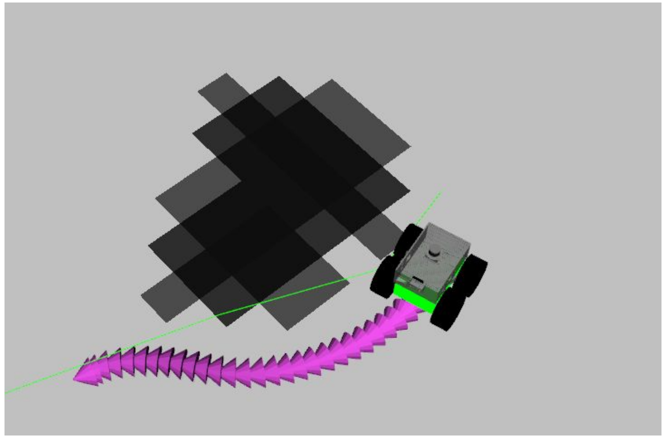
Nick Lawrance and Luca Bartolomei

5th July, 2022

V4RL VISION FOR ROBOTICS LAB

Tutorial Objectives

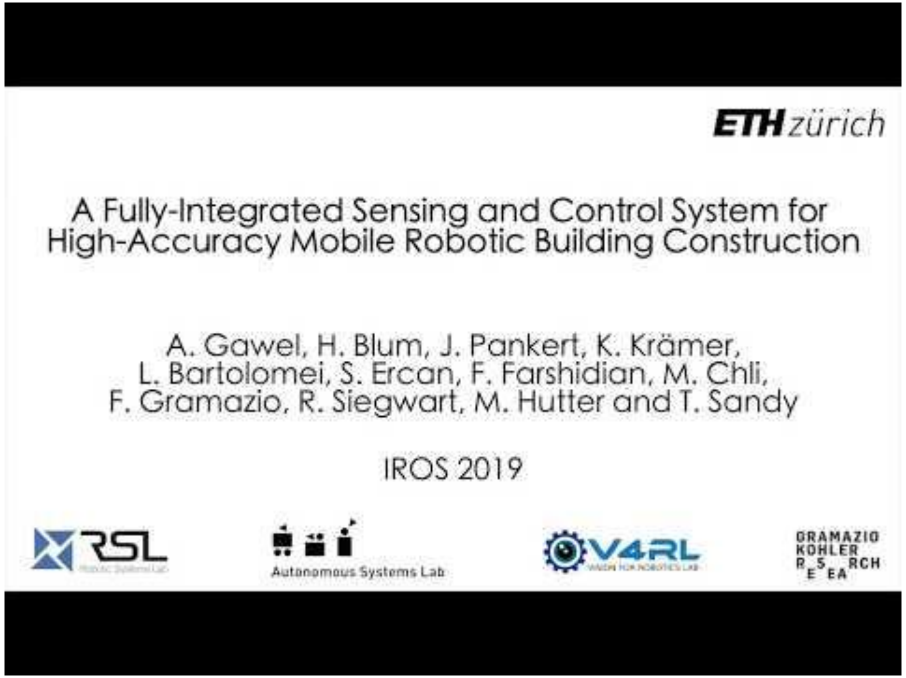
- (Brief) introduction to Path Planning
- Description of path-planning pipeline:
  - Global Planning
  - Local Planning
- How to install and run the packages



(Brief) introduction to Path Planning

Path planning:

- Autonomous goal-oriented navigation
- Obstacle avoidance



ETH zürich

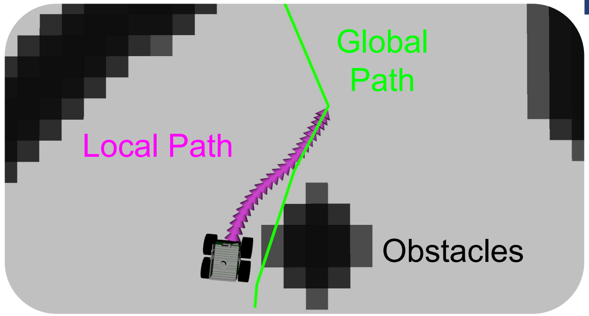
A Fully-Integrated Sensing and Control System for High-Accuracy Mobile Robotic Building Construction

A. Gawel, H. Blum, J. Pankert, K. Krämer, L. Bartolomei, S. Ercan, F. Farshidian, M. Chli, F. Gramazio, R. Siegwart, M. Hutter and T. Sandy

IROS 2019

RSL
Autonomous Systems Lab
V4RL
GRAMAZIO KOHLER RESEARCH

(Brief) introduction to Path Planning



- Hierarchical architecture

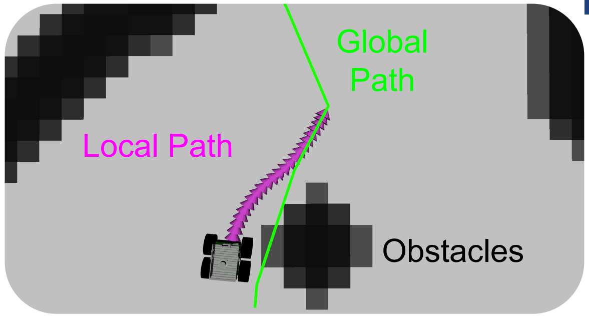
Global Planner →

- Coarse path to goal configuration
- Optimistic (unknown space is free)
- Low re-planning rate

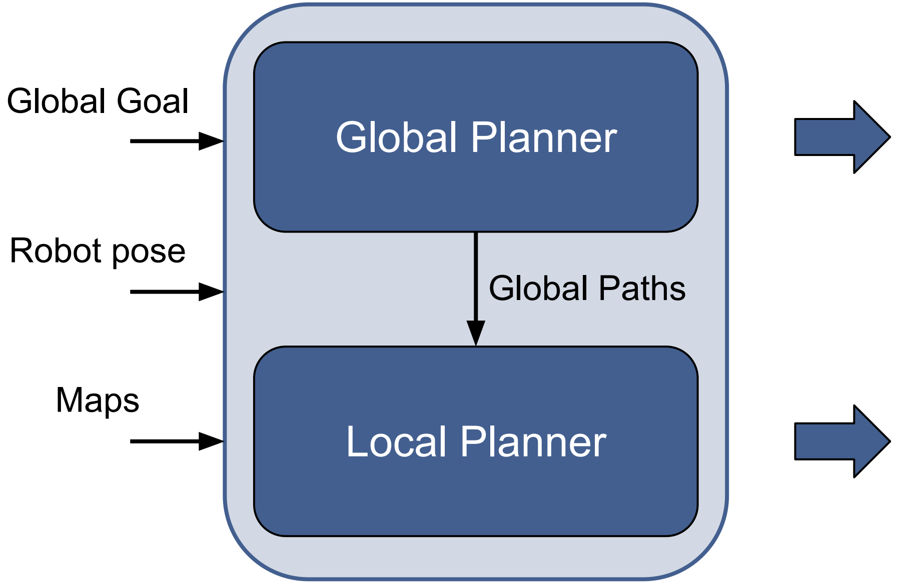
Local Planner →

- Local obstacle avoidance
- Feasible trajectories for controllers
- High re-planning rate

(Brief) introduction to Path Planning



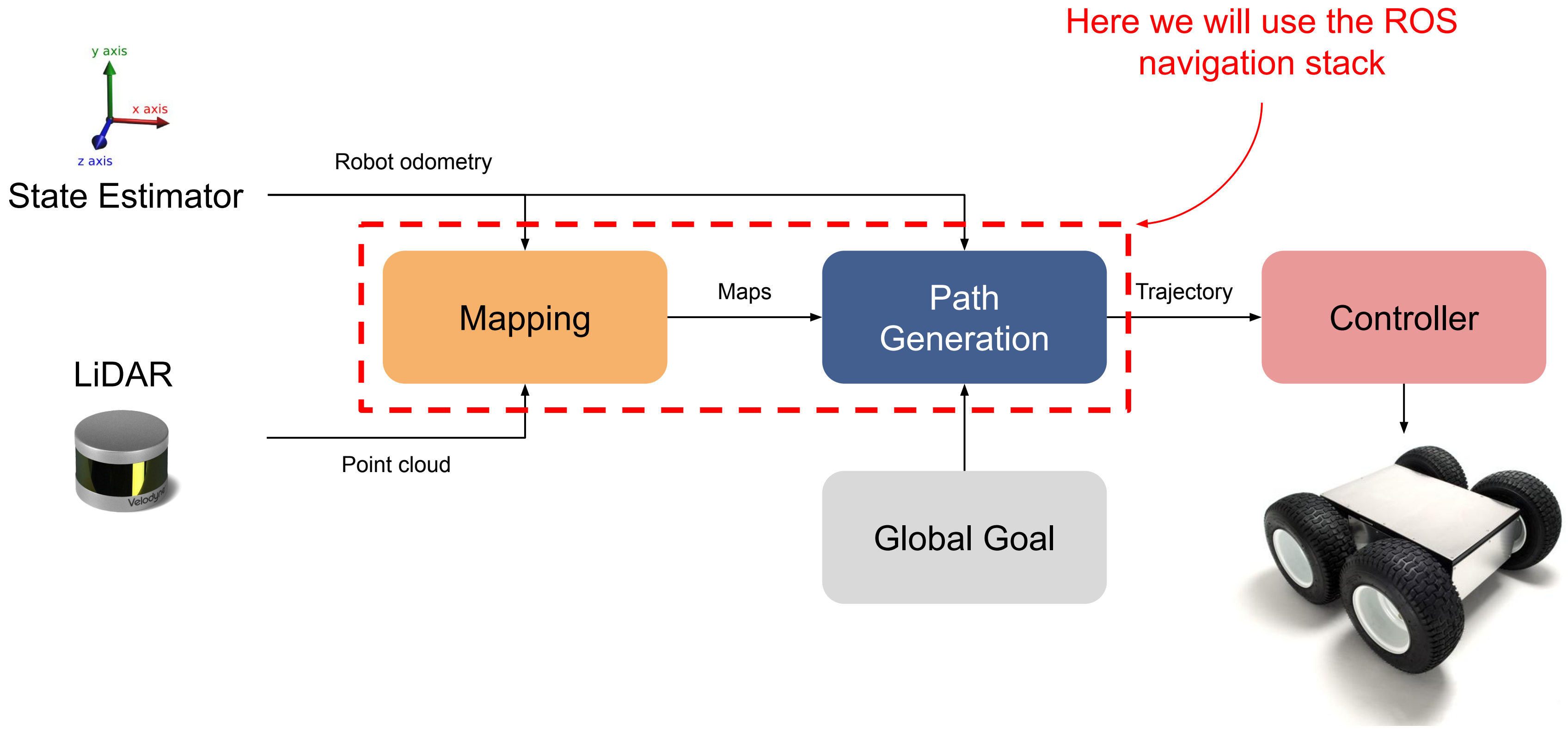
- Hierarchical architecture



- Coarse path to goal configuration
- Optimistic (unknown space is free)
- Low re-planning rate

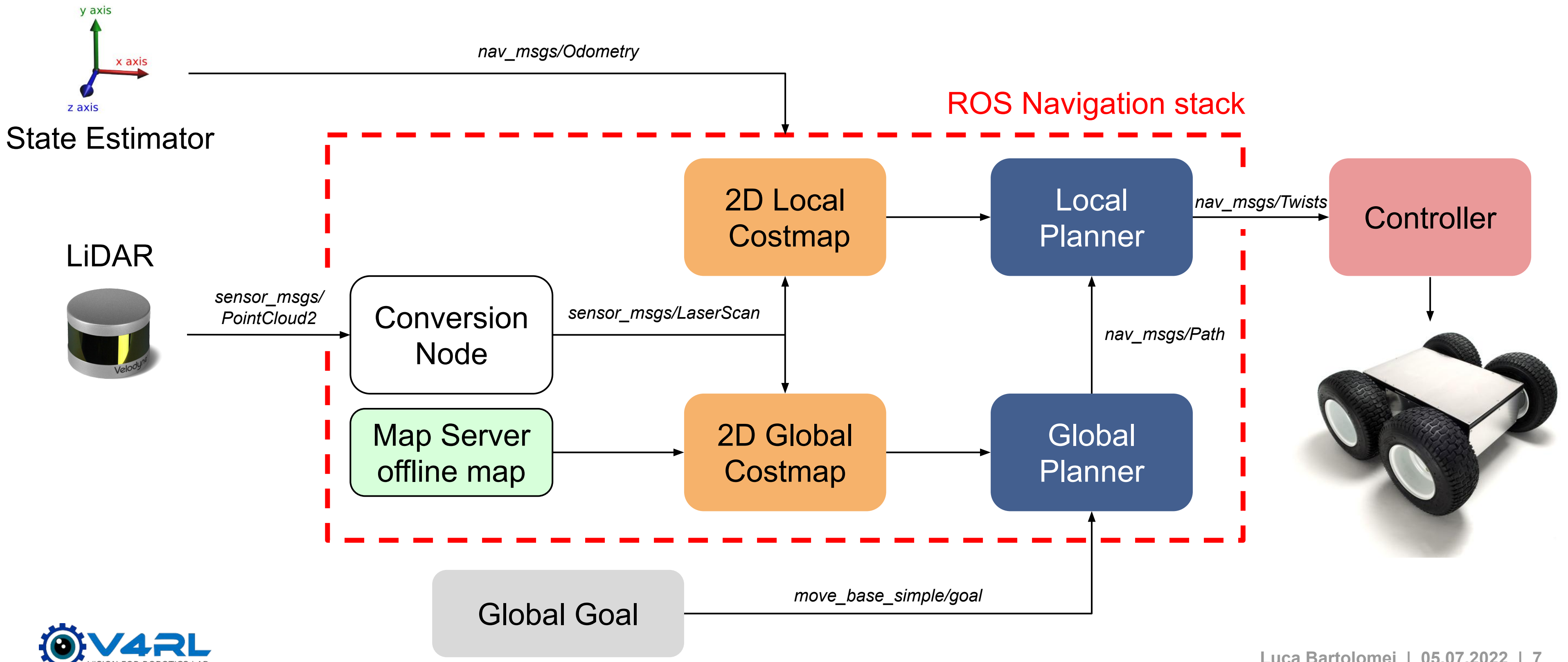
- Local obstacle avoidance
- Feasible trajectories for controllers
- High re-planning rate

Pipeline Overview



Detailed Pipeline

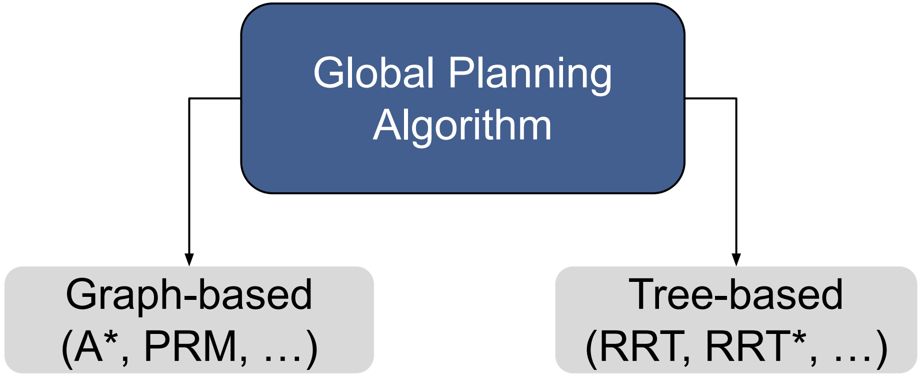
Docs: move_base - ROS Wiki

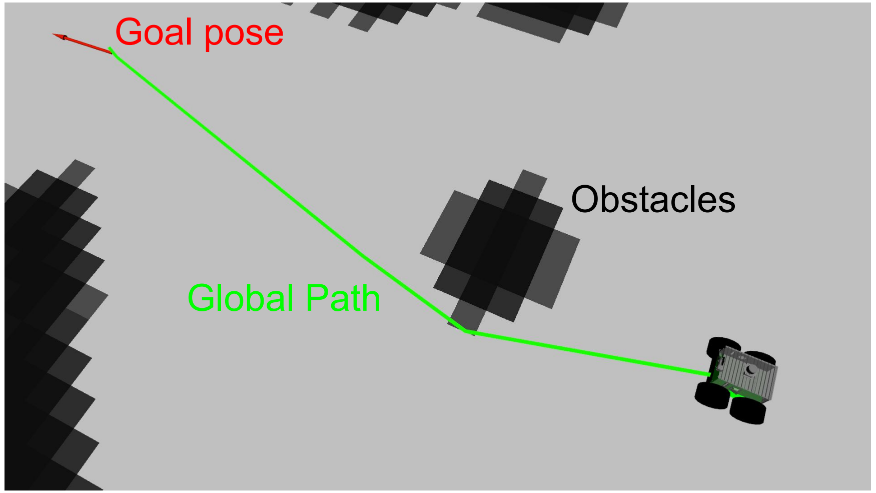


Path Planner: Global Planner

Docs: move_base - ROS Wiki

- Path from current state to global goal
- Optimistic → unknown space = free





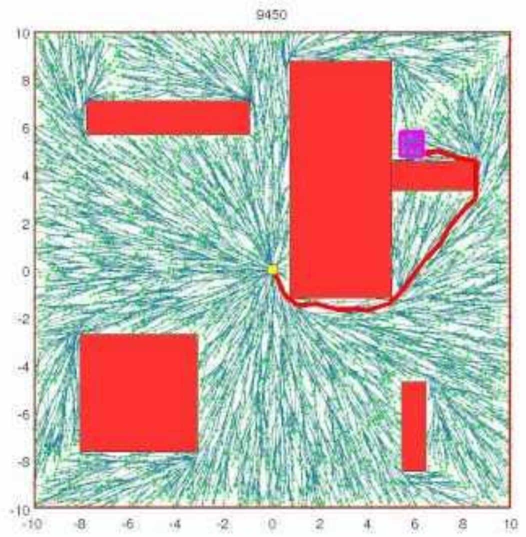
➡ **OMPL** (Open Motion Planning Library) - https://ompl.kavrakilab.org

Path Planner: Global Planner

Docs: move_base - ROS Wiki

- Custom implementation based on OMPL
  - Code: `smb_path_planner/smb_ompl_planner`
- Basic version: RRT*
  - Probabilistically complete and optimal algorithm

RRT*



"Sampling-based Algorithms for Optimal Motion Planning", S. Karaman and E. Frazzoli, IJJR 2011

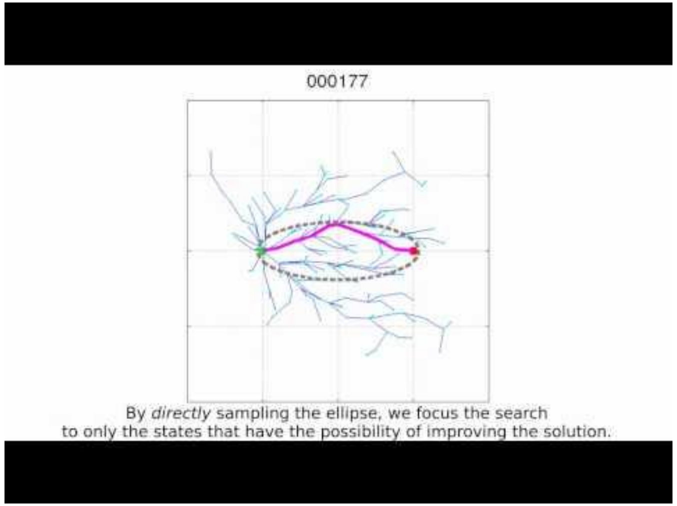
ETH zürich

Path Planner: Global Planner

Docs: move_base - ROS Wiki

- Custom implementation based on OMPL
  - Code: `smb_path_planner/smb_ompl_planner`
- Basic version: RRT*
  - Probabilistically complete and optimal algorithm
- Possibility to swap solver from config file:
  - Informed RRT*
  - RRT#
  - PRM

Informed RRT*

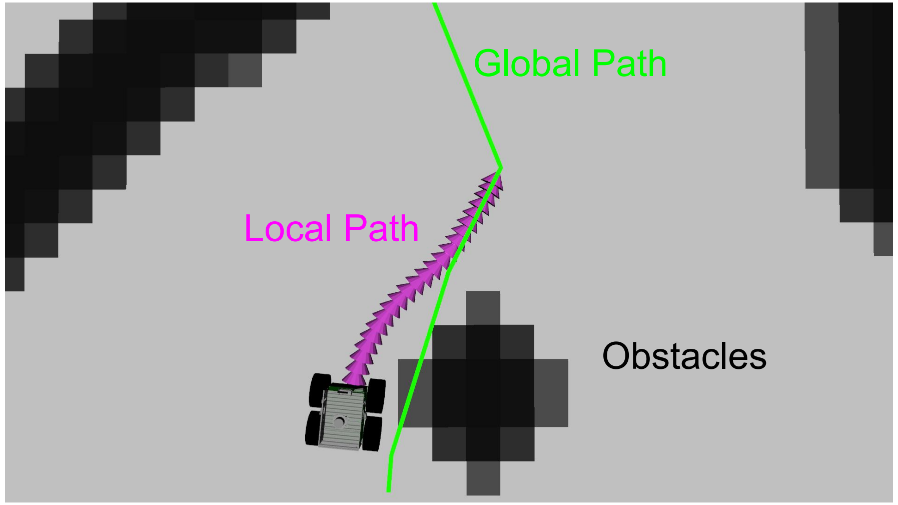

000177
By *directly* sampling the ellipse, we focus the search
to only the states that have the possibility of improving the solution.

"Informed RRT: Optimal Incremental Path Planning Focused through an Admissible Ellipsoidal Heuristic"*, J.D. Gammell, IROS 2014

V4RL VISION FOR ROBOTICS LAB

Path Planner: TEB Local Planner

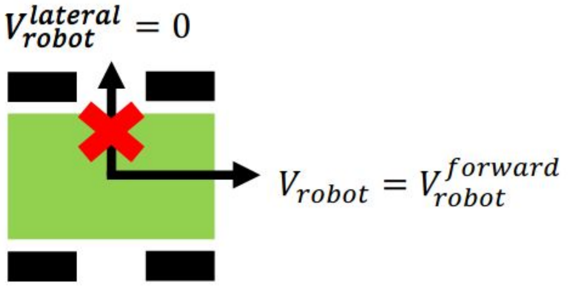
- Compute locally optimal paths
  - Vehicle dynamics
  - Obstacle avoidance
- Pessimistic
  → unknown space = occupied
- Use Time-Elastic-Band (TEB) Local Planner
  - C. Rösmann et al., IROS 2017
  - Docs: teb_local_planner - ROS Wiki

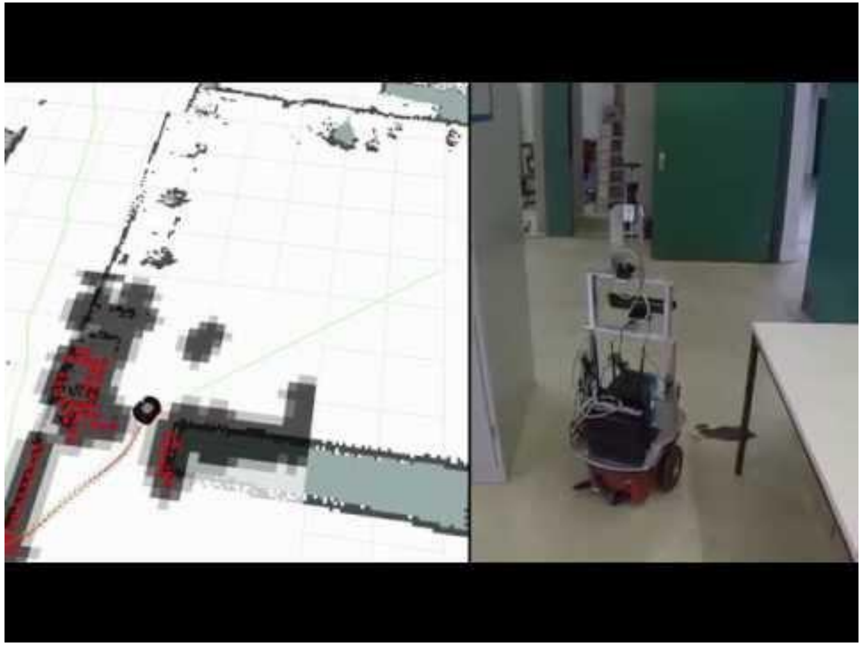


Path Planner: TEB Local Planner

Docs: teb_local_planner - ROS Wiki
Link to TEB paper

- Online trajectory optimization
- Objective: reach goal in minimum time
  - Kinodynamic constraints (velocity, acceleration, …)
  - Non-holonomic kinematics





Link to video

ETH zürich

Path Planner: TEB Local Planner

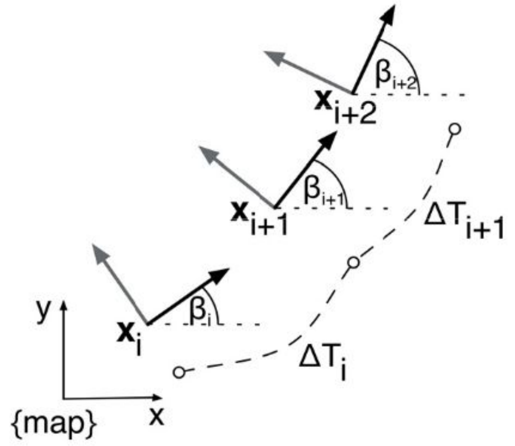
Docs: teb_local_planner - ROS Wiki
Link to TEB paper

- Trajectory parametrized as a rubber band
  - Waypoints:

$$\mathbf{x}_i = (x_i, y_i, \beta_i)^T \in \mathbb{R}^2 \times S^1$$

  - Timing between waypoints:

$$\tau = \{\Delta T_i\}_{i=0\ldots n-1}$$



- Optimize cost function:

$$f(B) = \sum_k \gamma_k f_k(B)$$

Function to be optimized

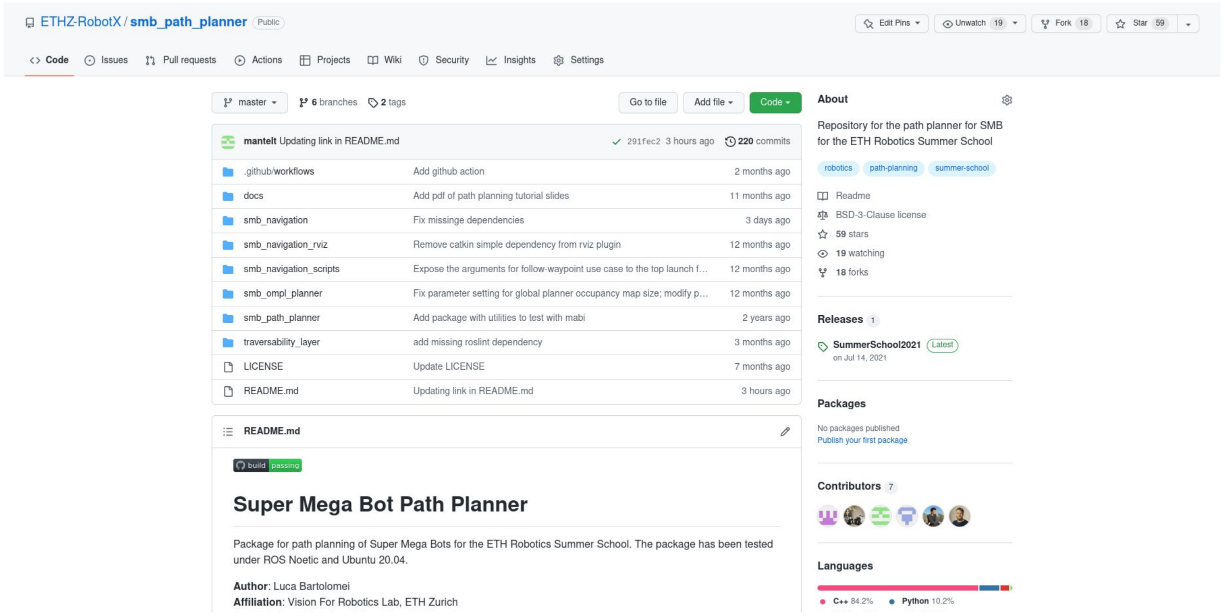
Waypoints and timing sequences

Weighted sum of
- Objectives (e.g. shortest path), and
- Soft constraints (e.g. kinematics, dynamics)

ETH zürich

Installation & Simulation Set-up

- Main webpage: https://github.com/ETHZ-RobotX/smb_path_planner


ETHZ-RobotX / **smb_path_planner** Public

Edit Pins | Unwatch 19 | Fork 18 | Star 59

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

master | 6 branches | 2 tags | Go to file | Add file | Code

mantelt Updating link in README.md ✓ 291fec2 3 hours ago 220 commits

.github/workflows	Add github action	2 months ago
docs	Add pdf of path planning tutorial slides	11 months ago
smb_navigation	Fix missinge dependencies	3 days ago
smb_navigation_rviz	Remove catkin simple dependency from rviz plugin	12 months ago
smb_navigation_scripts	Expose the arguments for follow-waypoint use case to the top launch f…	12 months ago
smb_ompl_planner	Fix parameter setting for global planner occupancy map size; modify p…	12 months ago
smb_path_planner	Add package with utilities to test with mabi	2 years ago
traversability_layer	add missing roslint dependency	3 months ago
LICENSE	Update LICENSE	7 months ago
README.md	Updating link in README.md	3 hours ago

README.md

build passing

Super Mega Bot Path Planner

Package for path planning of Super Mega Bots for the ETH Robotics Summer School. The package has been tested under ROS Noetic and Ubuntu 20.04.

Author: Luca Bartolomei
Affiliation: Vision For Robotics Lab, ETH Zurich

About

Repository for the path planner for SMB for the ETH Robotics Summer School

robotics | path-planning | summer-school

Readme
BSD-3-Clause license
59 stars
19 watching
18 forks

Releases 1

SummerSchool2021 Latest
on Jul 14, 2021

Packages

No packages published
Publish your first package

Contributors 7

Languages

C++ 84.2% | Python 10.2%

V4RL VISION FOR ROBOTICS LAB

Installation & Simulation Set-up

- Main webpage: https://github.com/ETHZ-RobotX/smb_path_planner
- Simply follow the instructions to install the planner
- Make sure code is up-to-date:
    - E.g. in the folder `catkin_ws/src`, run `vcs pull`

Main Packages	Description
`smb_navigation`	Main package (utilities, launch and configuration files)
`smb_navigation_scripts`	Utility scripts (follow waypoints, point cloud processing)
`smb_ompl_planner`	Global planner based on OMPL library
`smb_navigation_rviz`	RViz plugin to select the navigation goal

V4RL VISION FOR ROBOTICS LAB

Running in Simulation - basic usage

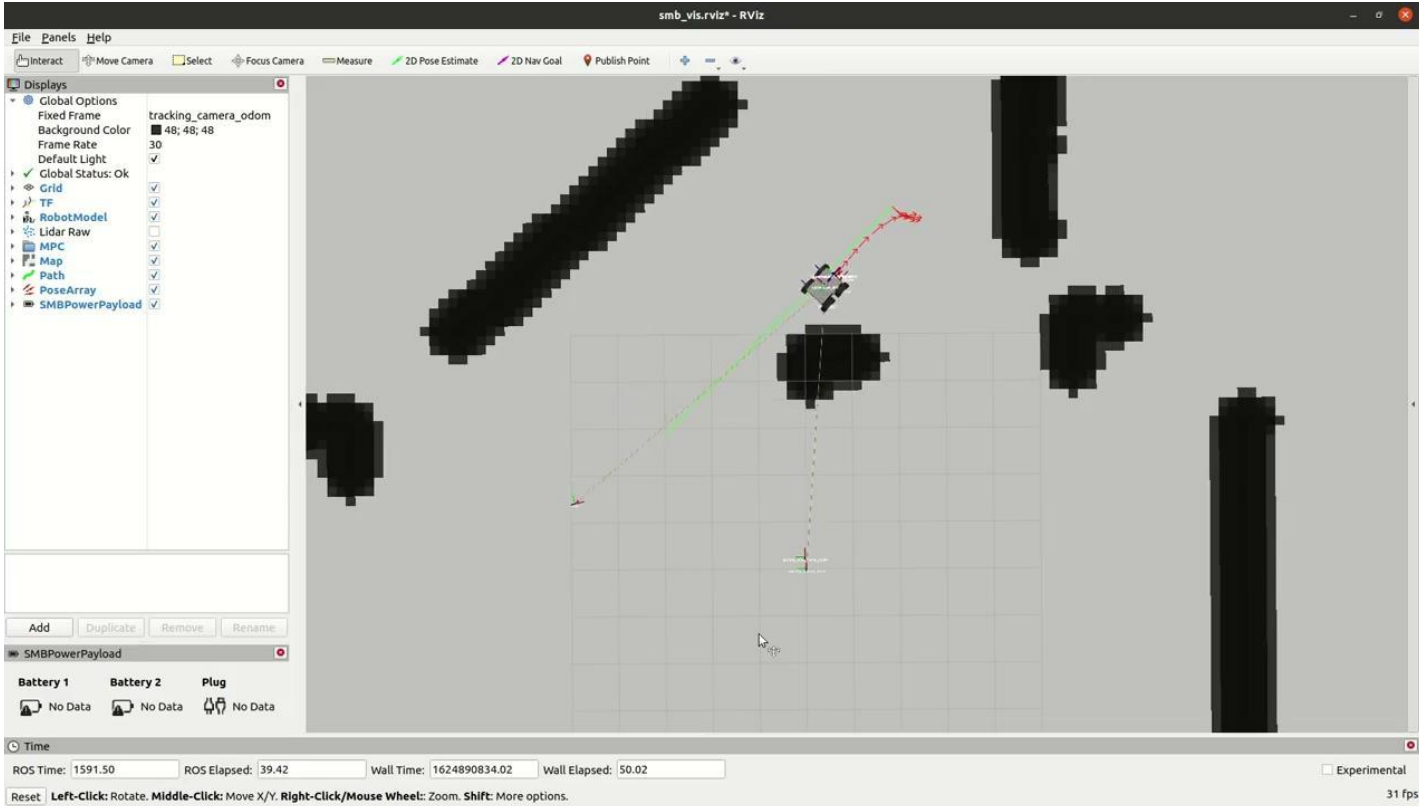
- Refer to the documentation

- Quick start:
  1. Start the simulation:
     `$ roslaunch smb_gazebo sim.launch world:=planner_tutorial`
     a. You should be able to see the SMB in RViz
     b. To see the SMB in Gazebo, add `launch_gazebo_gui:=true`

  2. Start the planner:
     `$ roslaunch smb_navigation navigate2d_ompl.launch sim:=true global_frame:=tracking_camera_odom`
     a. You should see the occupancy map in RViz

  3. Send goal position using "`2D Nav Goal`" button in RViz

Running in Simulation - basic usage



Global Options

Fixed Frame	tracking_camera_odom
Background Color	■ 48; 48; 48
Frame Rate	30
Default Light	✓
✓ Global Status: Ok	
Grid	✓
TF	✓
RobotModel	✓
Lidar Raw	
MPC	✓
Map	✓
Path	✓
PoseArray	✓
SMBPowerPayload	✓

SMBPowerPayload

Battery 1	Battery 2	Plug
No Data	No Data	No Data

ROS Time: 1591.50 ROS Elapsed: 39.42 Wall Time: 1624890834.02 Wall Elapsed: 50.02

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift**: More options.

31 fps

Running in Simulation - basic usage

- Refer to the documentation

  - Run `rqt_reconfigure` to tune the planner online (remember to save the parameters!):

    ```
    $ rosrun rqt_reconfigure rqt_reconfigure
    ```

  - Play with the different global planners
    - Change OMPL planner by editing config file:
      `smb_navigation/config/ompl_global_planner.yaml`
    - Run `move_base` global planner (using A*):

      ```
      $ roslaunch smb_navigation navigate2d.launch sim:=true
      global_frame:=tracking_camera_odom
      ```

Running in Simulation - advanced features

- Refer to the documentation for advanced features
- It is possible to specify...
  - ... a different odometry topic:
    ```
    $ roslaunch smb_navigation navigate2d_ompl.launch  odom_topic:=/odom_new
    ```
  - ... different reference frames:
    ```
    $ roslaunch smb_navigation navigate2d_ompl.launch  global_frame:=map_new
    robot_base_frame:=base_new
    ```

Running in Simulation - advanced features

- Refer to the documentation for advanced features
- It is possible to specify...
    - … a different odometry topic:

      ```
      $ roslaunch smb_navigation navigate2d_ompl.launch  odom_topic:=/odom_new
      ```
    - … different reference frames:

      ```
      $ roslaunch smb_navigation navigate2d_ompl.launch  global_frame:=map_new
      robot_base_frame:=base_new
      ```
- Use existing global maps for planning (e.g. from previous missions)
- Follow a set of waypoints

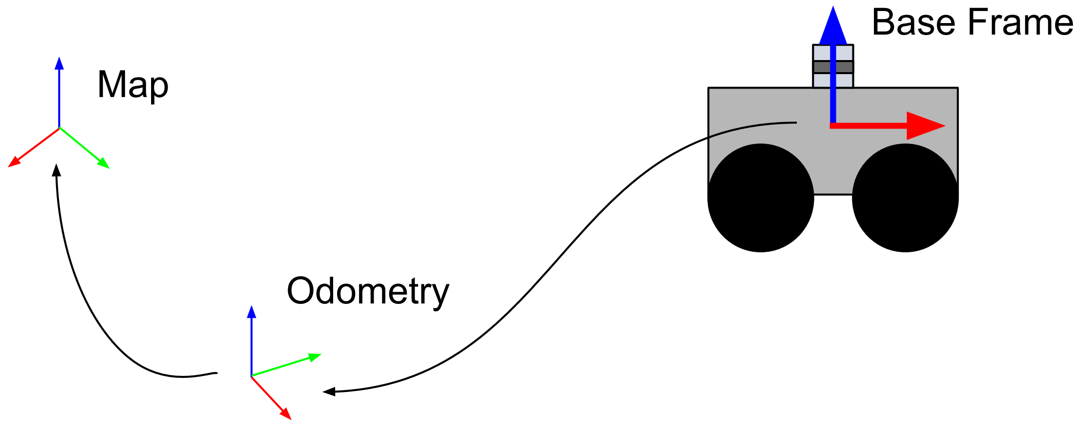
Running in Simulation - advanced features

- Refer to the documentation for advanced features
- It is possible to specify...
  - ... a different odometry topic:
    ```
    roslaunch smb_navigation navigate2d_ompl.launch odom_topic:=/odom_new
    ```
  - ... different reference frames:
    ```
    roslaunch smb_navigation navigate2d_ompl.launch global_frame:=map_new
    robot_base_frame:=base_new
    ```
- Use existing global maps for planning (e.g. from previous missions)
- Follow a set of waypoints

V4RL VISION FOR ROBOTICS LAB

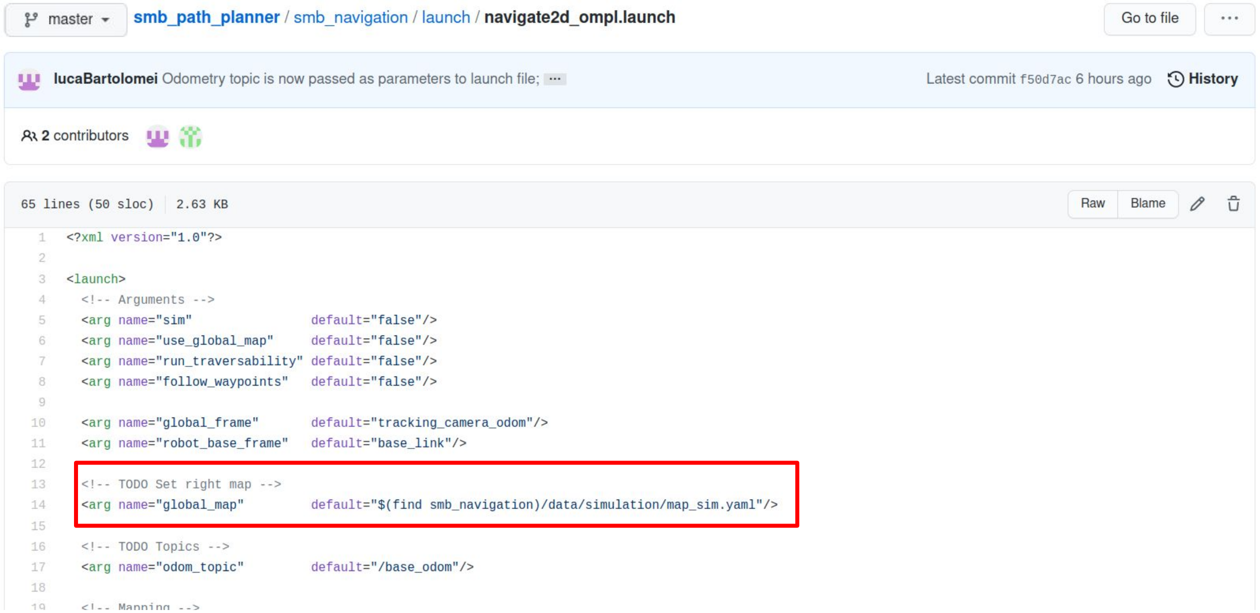
Running in Simulation - use existing global maps (doc)

- Global planning → static and globally consistent map
  - Finds the shortest path in the complete map (*Map* frame)
  - Uses TF connections to retrieve transformations (from *Base Frame* to *Map*)
- Local planning & control → *Odometry* frame
  - Use odometry information directly (drifting, but continuous estimates)



Running in Simulation - use existing global maps (doc)

- For this tutorial, we provide a map of the simulation environment
- Follow these steps:
  - Set the path to the global map in the launch file (already done for simulation)


master smb_path_planner / smb_navigation / launch / navigate2d_ompl.launch

lucaBartolomei Odometry topic is now passed as parameters to launch file; … Latest commit f50d7ac 6 hours ago History

2 contributors

65 lines (50 sloc) | 2.63 KB

```
<?xml version="1.0"?>

<launch>
  <!-- Arguments -->
  <arg name="sim"                default="false"/>
  <arg name="use_global_map"     default="false"/>
  <arg name="run_traversability" default="false"/>
  <arg name="follow_waypoints"   default="false"/>

  <arg name="global_frame"       default="tracking_camera_odom"/>
  <arg name="robot_base_frame"   default="base_link"/>

  <!-- TODO Set right map -->
  <arg name="global_map"         default="$(find smb_navigation)/data/simulation/map_sim.yaml"/>

  <!-- TODO Topics -->
  <arg name="odom_topic"         default="/base_odom"/>

  <!-- Mapping -->
```

Running in Simulation - use existing global maps (doc)

- For this tutorial, we provide a map of the simulation environment
- Follow these steps:
  - Set the path to the global map in the launch file (already done for simulation)
  - Run the planner:

```
$ roslaunch smb_navigation navigate2d_ompl.launch sim:=true
global_frame:=tracking_camera_odom use_global_map:=true
```

- The global planner now uses a ***static* global map**
  - Use `rqt_reconfigure` to turn on the obstacle and inflation layers

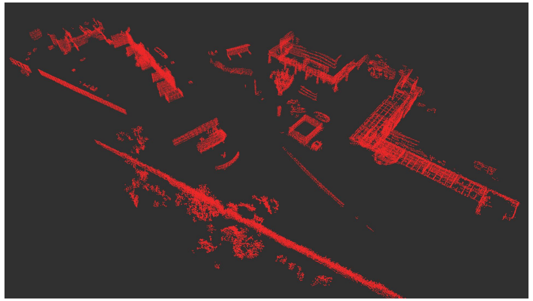
Running in Simulation - use existing global maps (doc)

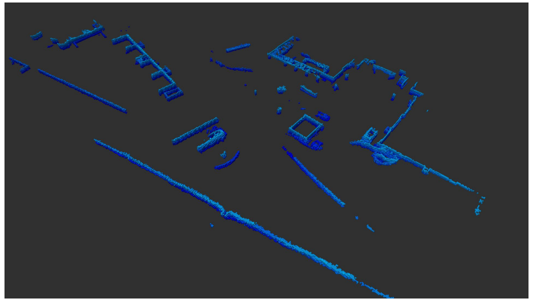
Running in Simulation - create global maps (doc)

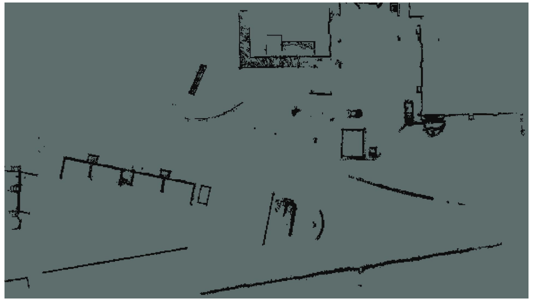
- But how do I create a global map (e.g. from SLAM)?

Running in Simulation - create global maps (doc)

- But how do I create a global map (e.g. from SLAM)?
- These instructions show how to create an occupancy map from `*.pcd` files


Original `*.pcd` file
(3D map)


OctoMap
(3D map)


Occupancy Grid
(2D map)

Running in Simulation - create global maps (doc)

Can take several minutes of computation!

- If you have a `compslam_map.pcd` file created by `smb_slam`, run default script:
  - Start `roscore` on your machine
  - `$ cd smb_path_planner/smb_navigation/script`
  - `$ python3 pcd_to_grid_map_default_paths.py`

Manual work!

- Using *default parameters* (`resolution, z_min, z_max`) defined in `pcd_to_gridmap.sh`!
  - Inspect the intermediate results in RViz

- This will create a `*.yaml` and a `*.pgm` files in `smb_navigation/data/test`
  - Make sure the origin in the `*.yaml` file does not contain NaNs
    → Otherwise, replace them with zeros
  - Check that the path to `*.pgm` file in `*.yaml` file is correct (relative paths are ok)

Running in Simulation - create global maps (doc)

Can take several minutes of computation!

- If you want more control, use the script directly:
  - `$ cd smb_path_planner/smb_navigation/script`
  - `$ chmod +x pcd_to_gridmap.sh`
  - `$ ./pcd_to_gridmap.sh <input_file> <output_folder> <run_rviz>`

`True` or `False`

Manual work!

- Using *default parameters* (`resolution, z_min, z_max`) defined in `pcd_to_gridmap.sh`!
  - Inspect the intermediate results in RViz
- This will create a `*.yaml` and a `*.pgm` files in the specified output folder
  - Make sure the origin in the `*.yaml` file does not contain NaNs
    → Otherwise, replace them with zeros
  - Check that the path to `*.pgm` file in `*.yaml` file is correct (relative paths are ok)

Running in Simulation - create global maps (doc)

- `pcd_to_gridmap.sh` uses default parameters
  - `resolution` → resolution of the grid map
  - `z_min`
  - `z_max`

  $\Big\}$ Height range of the point cloud

- **Method 1**: trial-and-error
  - Adjust the parameters and inspect results in RViz
  - Repeat until you get a reasonable result

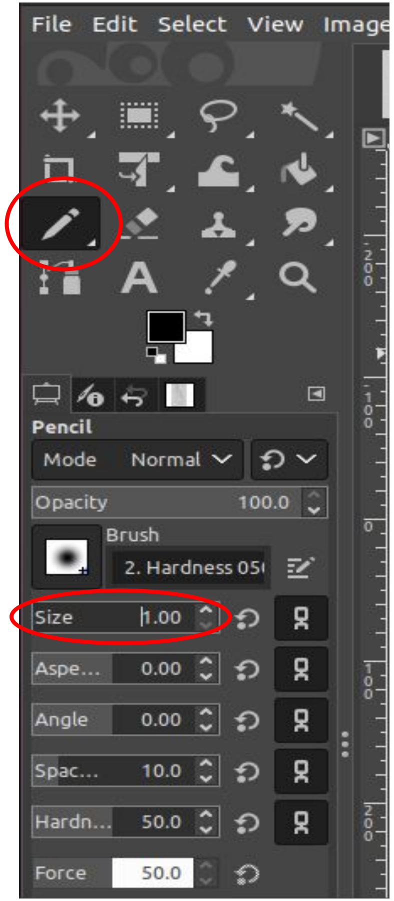
- **Method 2**: use image editing tools
  - Correct the image to remove artifacts

Running in Simulation - create global maps (doc)

- **Method 2**: use image editing tools
  - Correct the image to remove artifacts

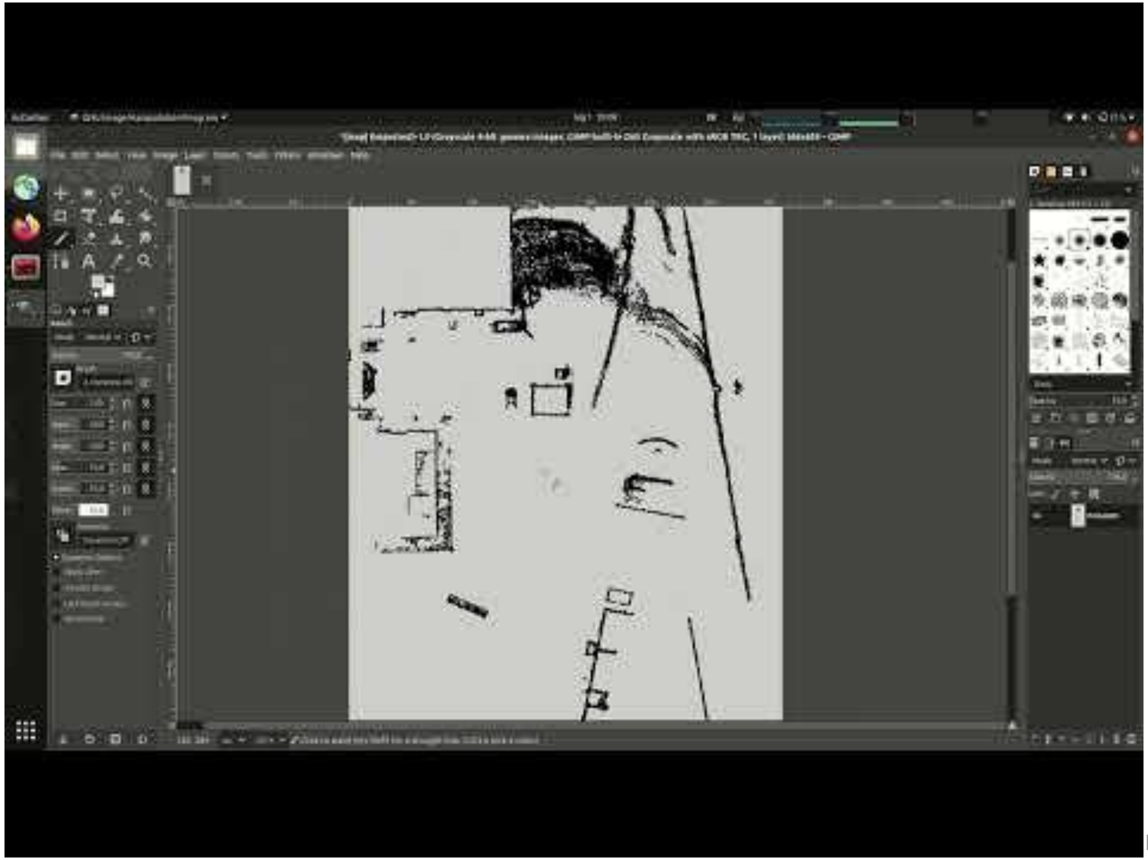
- Example with `GIMP` (install with `sudo apt install gimp`)

  1. Start `GIMP` and open the file `map.pmg`
  2. Select the pencil tool and choose proper size (e.g. 2)
  3. Select the right color and then you can:
     a. Remove artifacts
     b. Draw additional "fake" obstacles (e.g. borders)
  4. Save the new file:
     a. *File → Export As →* `.pgm` extension



Pencil
Mode Normal
Opacity 100.0
Brush 2. Hardness 050
Size 1.00
Aspe... 0.00
Angle 0.00
Spac... 10.0
Hardn... 50.0
Force 50.0

Running in Simulation - create global maps (doc)



Running in Simulation - advanced features

- Refer to the documentation for advanced features
- It is possible to specify…
  - … a different odometry topic:
    ```
    roslaunch smb_navigation navigate2d_ompl.launch odom_topic:=/odom_new
    ```
  - … different reference frames:
    ```
    roslaunch smb_navigation navigate2d_ompl.launch global_frame:=map_new
    robot_base_frame:=base_new
    ```
- Use existing global maps for planning (e.g. from previous missions)
- Follow a set of waypoints

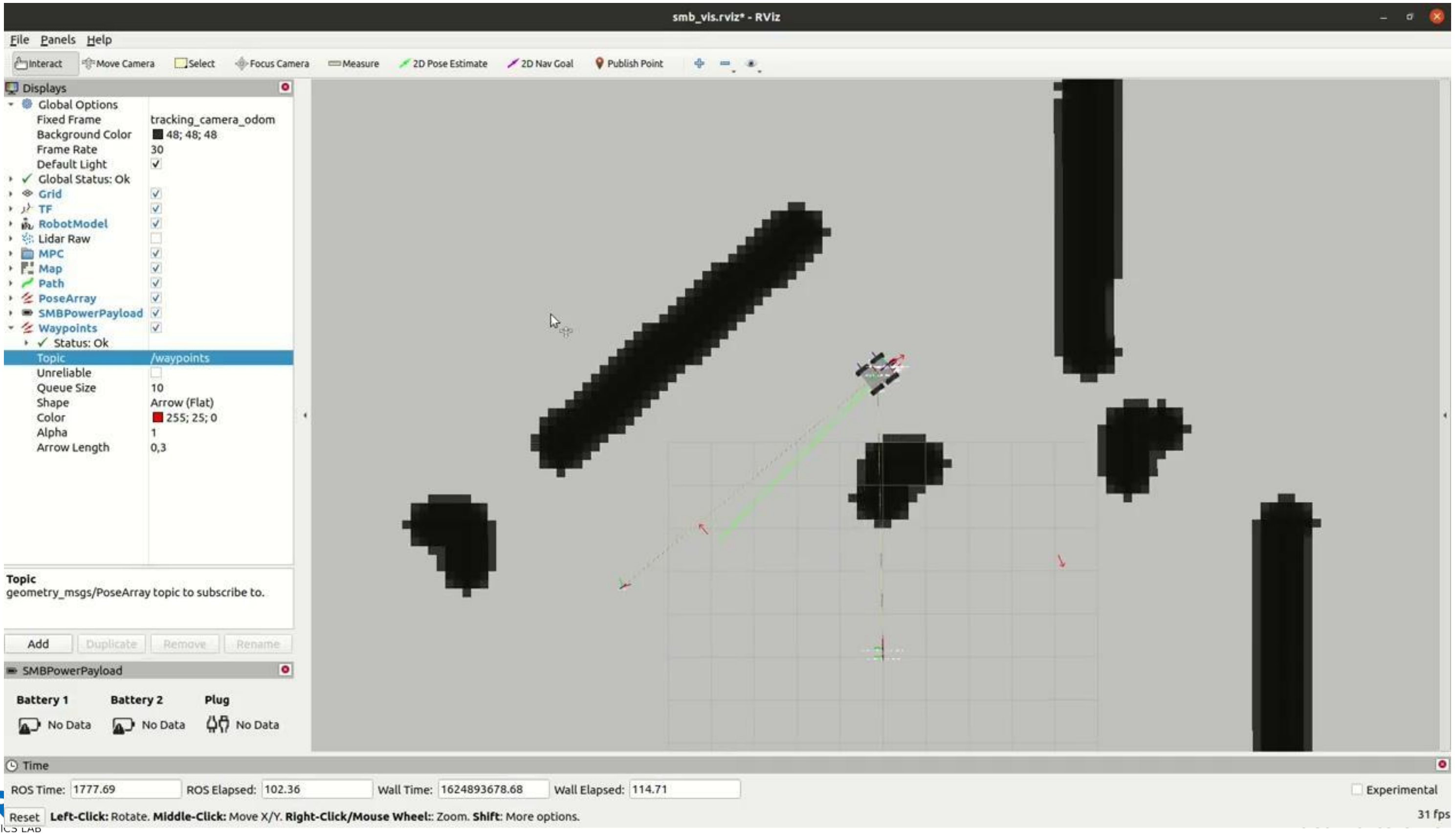
Running in Simulation - follow waypoints (doc)

- Start the simulation and run the command:

```
$ roslaunch smb_navigation navigate2d_ompl.launch sim:=true
global_frame:=tracking_camera_odom  follow_waypoints:=true
```

- **Online**:
  - Start the simulation and the planner
  - Set sequence of waypoints in RViz (button "`2D Pose Estimate`")
  - Call via terminal: `$ rostopic pub /path_ready std_msgs/Empty -1`
  - The waypoints will be stored in a file
    - Location specified in the launch file (parameter: `output_folder`)

Running in Simulation - follow waypoints (doc)


smb_vis.rviz* - RViz
File Panels Help
Interact Move Camera Select Focus Camera Measure 2D Pose Estimate 2D Nav Goal Publish Point
Displays
Add Duplicate Remove Rename

Global Options
Fixed Frame: tracking_camera_odom
Background Color: ■ 48; 48; 48
Frame Rate: 30
Default Light: ✓
Global Status: Ok
Grid ✓
TF ✓
RobotModel ✓
Lidar Raw
MPC ✓
Map ✓
Path ✓
PoseArray ✓
SMBPowerPayload ✓
Waypoints ✓
Status: Ok
Topic: /waypoints
Unreliable
Queue Size: 10
Shape: Arrow (Flat)
Color: 🟥 255; 25; 0
Alpha: 1
Arrow Length: 0,3

Topic
geometry_msgs/PoseArray topic to subscribe to.

SMBPowerPayload
Battery 1: No Data
Battery 2: No Data
Plug: No Data

Time
ROS Time: 1777.69 ROS Elapsed: 102.36 Wall Time: 1624893678.68 Wall Elapsed: 114.71 Experimental

Reset **Left-Click:** Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift**: More options. 31 fps

Running in Simulation - follow waypoints (doc)

- Start the simulation and run the command:

```
$ roslaunch smb_navigation navigate2d_ompl.launch sim:=true
global_frame:=tracking_camera_odom  follow_waypoints:=true
```

- **Online**:
  - Start the simulation and the planner
  - Set sequence of waypoints in RViz (button “`2D Pose Estimate`”)
  - Call via terminal: `$ rostopic pub /path_ready std_msgs/Empty -1`
  - The waypoints will be stored in a file
    - Location specified in the launch file (parameter: `output_folder`)

- **Offline**:
  - Specify the input file in the launch file (folder and file name) - make sure it exists!
  - Start the simulation and the planner
  - Call via terminal: `$ rostopic pub /start_journey std_msgs/Empty -1`

Running on real SMB

All the advanced features tested in simulation can be used with the real robot as well!

- Refer to the documentation

1. Start the robot and the sensors
2. Launch the state estimation and control pipelines
   a. Make sure everything works correctly (e.g. move the robot around with joypad)
3. Start the planner:

```
$ roslaunch smb_navigation navigate2d_ompl.launch
```

4. Select a goal and start planning

Tutorial Complete!

- Follow the documentation and you should be fine
- Cheat sheet with all the most important commands

References: main configuration files

- Global Cost Map:
  smb_navigation/config/move_base_costmaps/global_costmap_params.yaml
- Local Cost Map:
  smb_navigation/config/move_base_costmaps/local_costmap_params.yaml
- Global Planner:
  smb_navigation/config/move_base_costmaps/ompl_global_planner.yaml
- Local Planner:
  smb_navigation/config/base_local_planner.yaml (sim / real)

V4RL VISION FOR ROBOTICS LAB