BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

# Advanced robotics lab

## ROBa - Laboratory number 6

https://github.com/Adam-Fabo/ROB-laboratories

Brno university of technology
Faculty of informatics
Adam Fabo
4.4.2023

# Advanced robotics lab

Welcome to the sixth laboratory. This laboratory will show you more advanced robotics concepts.

# Prerequisites

Prerequisites are that you have attended previous laboratories, and you know basics of ROS and ROS commands. Also, it is important that you have attended theoretical lecture about SLAM, gmapping and simulation environments.

# Goals of the laboratory

Goals of this laboratory are to show you more advanced topics. You will learn:

- Recording and replaying ROS messages using Trilobot
- Visualizing data from depth camera
- Basic simulations using gazebo
- Application of advanced topics like SLAM in simulation

# Theory

## Recording ROS messages

Recording messages is useful when data collecting process is tedious and takes a long time. This is why ROS has a command line tool called **rosbag**. This tool can record all messages sent from each node for a given time, and then it can replay the recorded messages. Message records are stored in **.bag** files.

Message recording of all messages in all topics can be done using the following command:

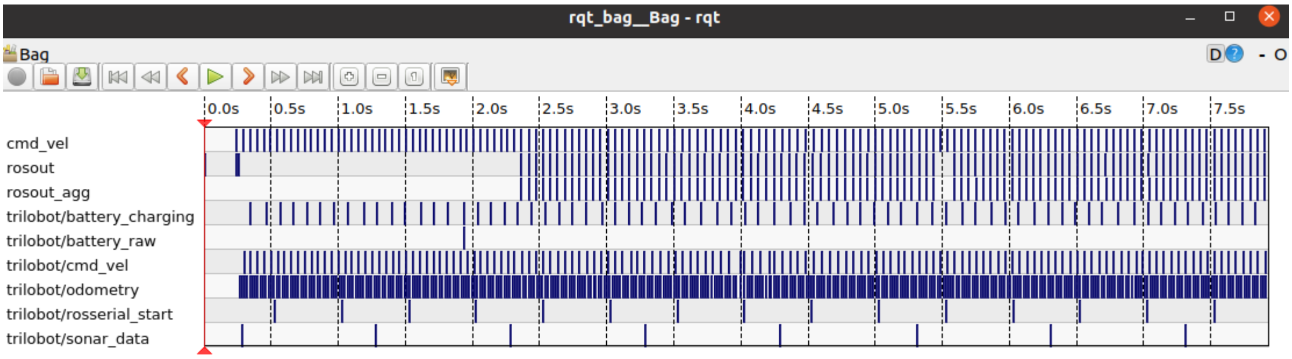
```
rosbag record -a -O <file_name>.bag
```

Replaying stored messages from .bag file can be done using following command:

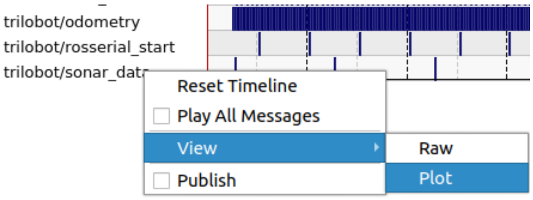
```
rosbag play <file_name>.bag
```

## Visualizing recorded messages

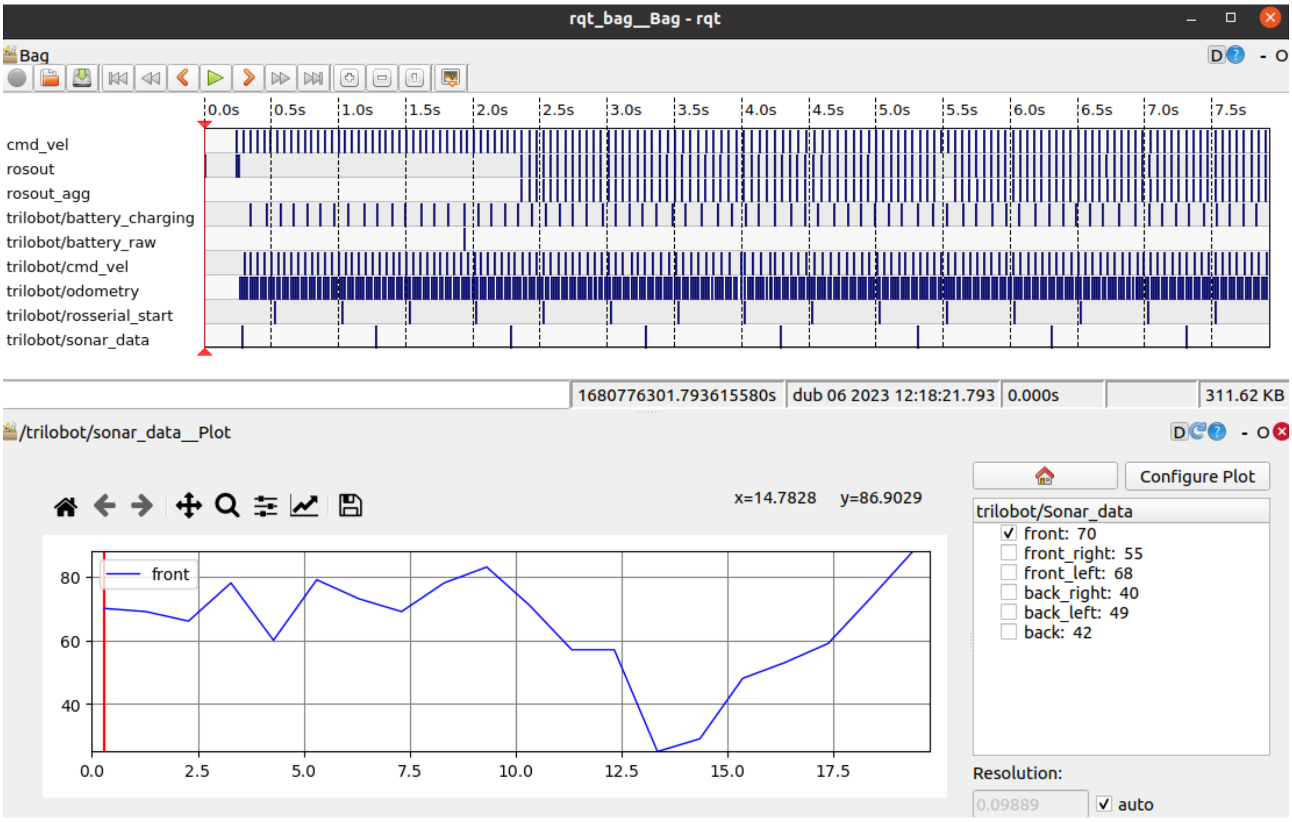
Tool **rqt_bag** offers GUI for visualization of recorded messages in .bag file. It can display graphs, photos and videos that have been captured using rosbag command. Contents of .bag file can be visualized by running: `rqt_bag <file_name>.bag`



Visualizing graphs is possible by right-clicking at topic and selecting View->Plot:



After that, new part of the window opens up where data from the topic are plotted as 2D graph:



1680776301.793615580s | dub 06 2023 12:18:21.793 | 0.000s | 311.62 KB

x=14.7828 y=86.9029

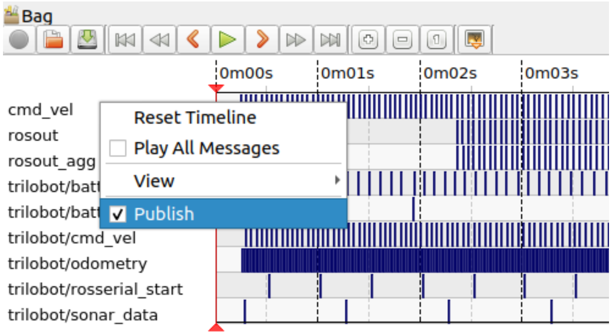
trilobot/Sonar_data

☑ front: 70
□ front_right: 55
□ front_left: 68
□ back_right: 40
□ back_left: 49
□ back: 42

Resolution:

0.09889 ☑ auto

Tool rqt_bag can also **replay** messages by pressing play icon. Messages are not published to the real topic by default, but it is needed to right-click at topic and tick publish box:

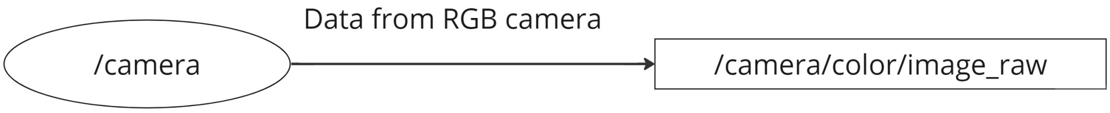


## Depth camera

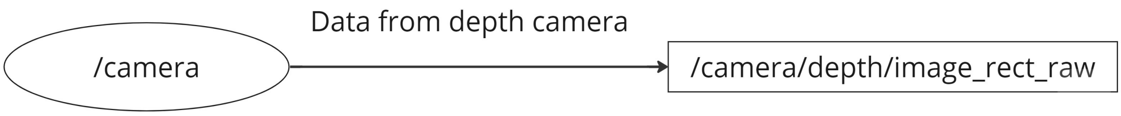
Robot Trilobot also has RGB camera and depth camera built in. Depth camera, unlike normal camera, does not output images in which pixels represent color. It outputs images where value of each pixel is a distance measured by the camera.

Figure 1: Depth camera

Data from RGB camera are sent into **/camera/color/image_raw** topic. This data are represent normal video stream that you would expect from any other classic camera.

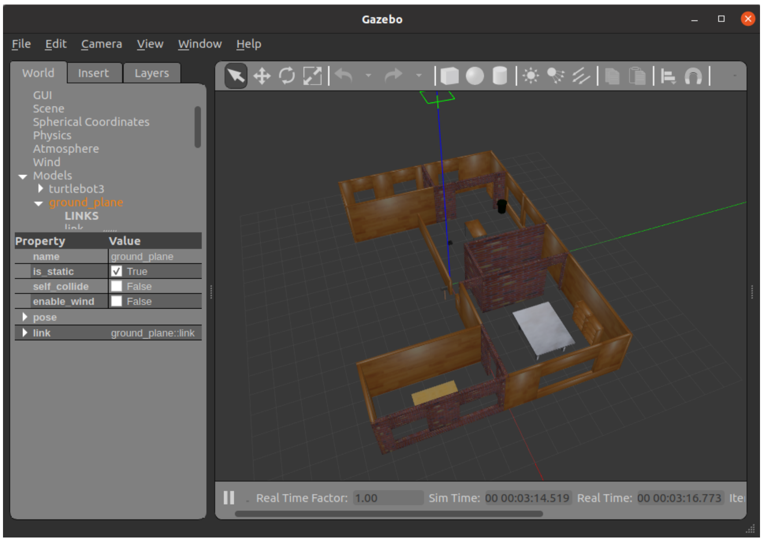


Data from depth camera are sent into **/camera/depth/image_rect_raw** topic.



## Gazebo simulation

A huge part of robotics is aimed at simulations. Simulations allow for testing concepts and algorithms without building the robot in real life. This saves money and time. In this laboratory is used Gazebo 3d open-source robotic simulator. It allows user to build its own robots and environments for testing.

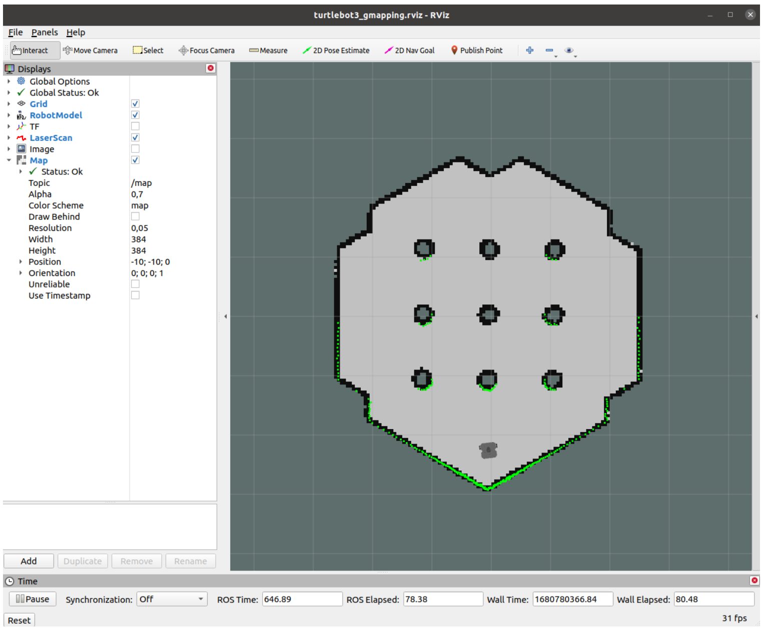


| Property | Value |
|---|---|
| name | ground_plane |
| is_static | ☑ True |
| self_collide | □ False |
| enable_wind | □ False |
| pose | |
| link | ground_plane::link |

Real Time Factor: 1.00 Sim Time: 00 00:03:14.519 Real Time: 00 00:03:16.773

## Visualization using rviz

Data visualization is a core part of robotics. That is why there are tools such rviz. It is 3d visualization tool for ROS. It can show data from sensor such as LIDARs, cameras, depth cameras ...


turtlebot3_gmapping.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
Time
Pause
Reset

| Displays | |
|---|---|
| Global Options | |
| Global Status: Ok | |
| Grid | ☑ |
| RobotModel | ☑ |
| TF | □ |
| LaserScan | ☑ |
| Image | □ |
| Map | ☑ |
| Status: Ok | |
| Topic | /map |
| Alpha | 0,7 |
| Color Scheme | map |
| Draw Behind | □ |
| Resolution | 0,05 |
| Width | 384 |
| Height | 384 |
| Position | -10; -10; 0 |
| Orientation | 0; 0; 0; 1 |
| Unreliable | □ |
| Use Timestamp | □ |

Synchronization: Off ROS Time: 646.89 ROS Elapsed: 78.38 Wall Time: 1680780366.84 Wall Elapsed: 80.48

31 fps

# Exercises

## Recording ROS messages

Use command line tool rosbag to record messages to and from Trilobot. After you have recorded the message flow, replay it and observe Trilobot

1. Turn on Triboot and put it on the ground in starting location
2. Navigate to **/catkin_ws/src/trilobot_arrows/src/** and run `python3 arrows_control_solved.py`
3. Record all ROS messages by running command: `rosbag record -a -O Trilo.bag`
4. Put the Pygame window with arrow control in focus and drive trilobot for 10-20 seconds
5. End recording of the messages by pressing Ctlr-C in terminal with rosbag process running
6. Close Pygame window

## Replaying ROS messages

Replay the recorded messages.

1. Put Trilobot on the place where you started recording
2. Replay recorded message by running command `rosbag play Trilo.bag`
3. Observe Trilobot replaying previous movement

## Visualizing collected data

Visualize the gathered data using rqt_bag.

1. Run command `rqt_bag Trilob.bag`. New window should open up.
2. In this window take a look at recorded data and see how messages are sent when controlling Tilobot. Focus on sonar_data and cmd_vel.
3. Plot the sonar data into a 2D graph by right-clicking on sonar data topic, and selecting View->Plot. Arrange windows so that graph is visible.
4. Replay movement of the Trilobot. Set Trilobot into its starting location. Right click at topic cmd_vel and tick checkbox publish. Replay the messages by pressing spacebar, or play button.

## Depth camera

Visualize data from normal and depth camera. Move around room with robot and look at the camera output.

1. Run command `rosrun image_view image_view image:=/camera/color/image_raw` to open new window with camera feed.
2. Navigate to **/catkin_ws/src/trilobot_arrows/src/** and run `python3 arrows_control_solved.py`
3. Navigate robot around the room and look at the camera output
4. Close the window with camera visualization and run visualization of depth camera by `rosrun image_view image_view image:=/camera/depth/image_rect_raw`
5. Once again, control robot using keyboard and look at the camera output

## Gazebo - Turtlebot3

Test if your simulation environment is working.

1. Close everything from previous exercises and turn off the Trilobot
2. Open new Gazebo window with simulation by running command `roslaunch turtlebot3_gazebo turtlebot3_empty_world.launch`
3. Run teleop that allows for keyboard control of the robot using: `roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch`
4. Control the robot from this terminal OR open Pygame window

## Gazebo - Turtlebot3 - SLAM

Use gazebo simulation for Simultaneous Localization and Mapping. Create a map of the simulated environment.

1. Keep everything running as it is
2. RUN command `roslaunch turtlebot3_slam turtlebot3_slam.launch slam_methods:=gmapping`
3. Control the robot with keyboard and map the whole simulated space
4. After whole simulation is mapped, save the map by running `rosrun map_server map_saver -f ~/map`

## Gazebo - Turtlebot3 - Navigation

Use map created in previous exercise for Navigation.

1. Close Rviz window
2. Run command `roslaunch turtlebot3_navigation turtlebot3_navigation.launch map_file:=$HOME/map.yaml`
3. Control robot with keyboard and take a look at the algorithm working
4. Change different starting locations by clicking at 2D pose estimate and once again control the robot
5. Close Pygame window
6. Use navigation goal button to set different locations and observe the robot moving