Road Map

ROS2 Autonomous Navigation Learning Guide

Getting an Overview

1. 2021 Thesis

Starting by reading 2021 thesis will provide an almost complete overview of autonomous systems components.

Related pages in the thesis are:

- 35 – 49: those contain general concepts
- 63 – 83: those pages focus on their implementation

Going through the project's code will also provide good understanding of the navigation and control processes

2. Navigation Concepts (1)

Reading this documentation section will provide a general understanding of navigation systems in ROS2. The sole objective is gaining familiarity with jargon and concepts. Concepts like behavior trees and Lifecycle nodes may not be easy to understand at first. You may need to re-read some parts a couple of times. However, for the time being, only basic understanding is needed. These concepts will be needed in integrating the whole project later on, thus a thorough understanding may not be needed till then.

Common Basics

1. Standard Coordinate Frames and Transforms (1, 2, 3)

The first link quick and light introduction focusing on the standards of mobile robots frames (the implementation in this tutorial is directed for ROS1, thus you can ignore it).

The second link elaborates a bit more on the concept of transforms and gives a hint on ROS2 implementation.

The third link gives a broader view on the whole transformations implementation in a conceptual way.

2. URDF - Robot Modeling and into to RViz(1, 2, 3)

the first link provides a quick intro to basic robot anatomy and URDF files intuition, as well as a an example that serves as practice on URDF files, launch files, transformations, and introducing RViz. Code files aren't thoroughly explained, but reading them will suffice.

The second link elaborates on the robot anatomy standards as well as on building URDF files - providing step-by-step implementation example. The example includes practice on launch files, and implementation of sending static transforms through a robot publisher node.

The third link explains in more – needed - details the URDF files, elaborating on its syntax as well as the XML syntax.

3. Gazebo (1, 2)

The first link is a walk-through of building SDF files of robots and opening them in Gazebo either using the terminal or launch files. Note that the code isn't explained but reading is sufficient as it's self-explanatory and embedded with lots of comments. Note also that you don't need to complete the entire article if you don't desire, you can stop after understanding the SDF file basic components, using plugins to publish and subscribe to ROS2 topics, and opening models/worlds in Gazebo using launch files and the terminal

To better understand the SDF syntax, check the official documentation in the second link

Domain Specific Concepts

1. Odometry (1, 2, 3, 4, 5, 6)

The first three links serve to explain the Pose part in the odometry message

The first link explains what is a robot's pose and its components: position and orientation. The first 5 slides are the only ones needed, although the rest of slides are generally useful. The slides explains 2D pose at first (whose orientation is a single angle) then generalize the concept to 3D pose. However, the slides use Euler Angles for 3D orientation, whereas ROS uses Quaternions, thus the following two links are needed.

The second link, the answer in it explains the Pose message interface in ROS, and how -briefly- Quaternion rotation relates to Euler Angles.

The third link explains in details what are Quaternions and how they relate to 3D rotation. It consists of an article embedded with interactive videos that visualize interactively every mathematical concept involved.

The fourth link is a quick video that explains briefly ,and without derivation, how encoder data is transformed into control data (how the rate of wheel rotation is transformed into Twist info. The video doesn't explain how to further transform this data into pose, but simply integrate it – multiply by dt and add resultant to initial pose).

The fifth link is a video explaining simply and in great details the derivation of the equations used to transform encoder readings to pose and twist. You may only watch from minute 6:00 to 20:00.

The sixth link is a short video illustrating the need and effect of fusing wheel encoders data with IMU data.

2. Localization

2.1 Pose Estimation (sensor fusion using EKF) (1, 2)

The first link is a brief, yet very comprehensive introduction to Sensor Fusion. You may consider watching the next two videos in the playlist as they provide further understanding of the application of sensor fusion to the localization problem.

The second link briefly explains Kalman Filter in an intuitive way

2.2 Global Localization (Monte Carlo localization) (1)

A brief yet comprehensive understanding of Monte Carlo localization, the need for it, and its role

3. Map Building using SLAM (1)

This link provides visual explanation of the working principle of SLAM, and its expected output

4. Motion Planning and Cost-maps (1)

The first link explains the high-level intuition of global costmaps and local costmaps. It explains briefly the concepts of global and local planning as well. You can skip the first 5 min. Beware that he doesn’t mention the role of velocity planning in local motion planning, so this video alone isn’t sufficient.

4.1 Global Path Planning (1, 2)

The first link explains briefly and visually the working principle of global path planning. You may skip the part explaining sampling-based methods if you wish.

The second link explains very briefly -in 5 min- the most popular search-based path finding algorithms and compares their performance. Note that the previous years used the Dijkstra algorithm.

4.2 Local Motion Planning (1, 2)

The first link explains very briefly, but comprehensively, the working principle of the DWA local planar. Explanation is aided visually using simulation.

The second link provides a more in-depth understanding regarding how the algorithm is responsible for dynamic obstacle avoidance as well as velocity planning.