Getting ROS container (preparation for hands-on part)

- go to https://github.com/apresland/ros2-turtlebot3-sim
- start with items 2. (download docker image)

ROS robot operating system

Andy Presland and Annina Blaas

Cpp User Group Zentralschweiz

June 13, 2022

Overview

1. ROS Introduction
2. ROS 1 vs. ROS 2
3. Some interesting details
4. Hands-on

ROS Intro

- ROS is a shortcut of **R**obot **O**perating **S**ystem
- not only "OS" but rather a robot development framework
- tools (simulation, debugging, visualisation, ...), code (core, "nodes", "drivers", ...) and community (tutorials, wiki, forum, ...)
- Open Source (BSD License, no "Copyleft")
- What is ROS (00:03:12 Video) from ROS: Home

ROS Intro cont.

- Development started 2007 at Stanford AI Lab
- Since 2012 developed from non-profit Open Source Robotics Foundation (OSRF)
- Since 2013 ROS Industrial Consortium (in addition)
- Annual Conference ROSCon
- Link to ROS: Home

ROS versions: ROS 1 vs. ROS 2

ROS 1

- Development from 2007 to 2020
- Long term support of latest version until 2025
- Huge community, extensive documentation
- Main OS: Ubuntu Linux

ROS 2

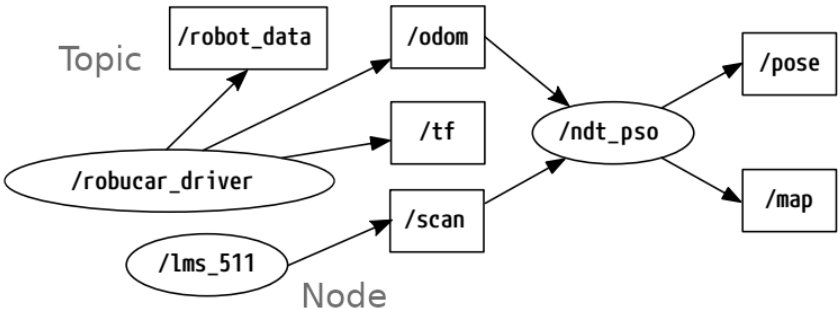
- Development from 2017 (ongoing)
- Not compatible, but inter operable with ROS
- Aim: real-time capability
- Supports small platforms (microROS for microcontrollers) ...
- ... and additional operating systems (macOS, Windows)

Into the details

(Graph) concept

- "Node": abstraction of hardware or logic, a node is an executable that uses ROS to communicate with other nodes
- "Topic": nodes can publish messages to a topic as well as subscribe to a topic to receive messages
- "Master": name service for ROS (i.e. helps nodes find each other)
- "rosout": ROS equivalent of stdout/stderr
- "Parameter Server": shared dictionary that is accessible via network APIs. Nodes can use this server to store and retrieve parameters at runtime, e.g. configuration parameters.

Visualization of nodes and topics (rqt_graph)



more details

roscore = Master + rosout + parameter server

- controls and monitors nodes and configuration
- first thing to start if using ROS 1
- no longer available on ROS 2 (replaced by peer-to-peer communication)

more details

ROS services

- alternative to topics
- for remote procedure calls, "client-server" approach
- request / response message pairs

ROS bags

- for recording and playback of messages
- efficient format
- e.g. for simulation

more details

build environment

- Own package related to cmake
  - "catkin_make = cmake + make"
- ROS 1: catkin package
- ROS 2: ament package

runtime environment

- ROS 1: Ubuntu Linux
- ROS 2: Ubuntu Linux, Red Hat Enterprise Linux, macOS, Windows 10
- cpp: built from libraries, e.g. from rclcpp "ros" cpp library
- Python: using packages installed in (virtual) environment

more details

ROS (debugging) tools

- ros2 wtf -r
- rqt tools
  - rqt_graph (see slide 8)
  - rqt_plot plots topic values over time
  - rqt_console displays log outputs
- rosbag
- rviz for data visualization, e.g. robot position in 3D
- ...

Personal summary

Pros

- Tools (rqt_image_view etc.)
- Learning curve (tutorials and forum)
- Widely used (components)

Cons

- Debugging
- "Safety" (Threads, ...)

Part 2 - Hands-on

Environment preparation: Andy Presland

Getting ROS container

- go to https://github.com/apresland/ros2-turtlebot3-sim
- from README do items 1. and 2.

Running ROS container

- go to https://github.com/apresland/ros2-turtlebot3-sim
- continue with item 3. of README
- the container contains a robot simulation and visualisation
- the robot can be teleoperated
- the simulated robot works like a physical ROS robot system and ROS tools can be used with it

Some information about the docker container

- Note that the folder **/workspace** of your linux system is mapped to folder **/dev_ws** of your container if the container is started from the correct location (i.e. folder **/ros2-turtlebot3-sim**). This allows to download / clone / change files on your native system and use the tools in the container to compile / run your code.
- Tools available in the container:
  - rviz2 (start from terminal with command rviz2): visualization of robot perception etc.
  - rqt (command rqt): graph, logging, image visualization (if there is a camera)
  - vi editor for viewing files

What's next? (Things you can try out)

You can

- run command ros2 topic list -v to plot a list of all active ros topics
- get to know rqt (see ROS 2 rqt introduction)
- run command ros2 wtf -r to print a summary of your ROS system and possible issues with it
- write a new node (see ROS 2 publisher tutorial)
- write a new service (see ROS 2 service tutorial)
- log to bagfile and play back (see ROS 2 bag tutorial)
- start gazebo simulation and write a control node that navigates the robot to the living room

Cheat Sheet

Command	Description
<ros2 cmd> -h	display help
ros2 node list	shows current ROS nodes
ros2 topic list -v	shows current ROS topics
ros2 topic info <topic>	information on a specific topic
ros2 run <packagename> <nodename>	starts this node
ros2 wtf	analyse system
ros2 bag record -a	start recording of all topics
ros2 param set	set a ROS parameter
ros2 param list	list all ROS parameters

Where to get help from

- ROS 1 Wiki
- ROS 1 Tutorials
- ROS 1 + 2 Forum
- ROS 2 (galactic geochelone) Documentation
- ROS 2 Tutorials
- ROS community is large and there is a lot of documentation, tutorials, a forum, ... on www.

References

- ROS: Home - https://www.ros.org/
- https://de.wikipedia.org/wiki/Robot_Operating_System
- https://answers.ros.org/question/351470/where-is-roscore-in-ros2/
- ROS 2 documentation: https://docs.ros.org/en/galactic/index.html
- https://answers.ros.org/question/187748/catkin-vs-cmake/

The End