



Introduction to ROS and the simulation

ST5 Autonomous robotics

Francis Colas

2022-09-02

Introduction

Autonomous Robots

- ▶ anatomy
- ▶ functions
 - ▶ perception
 - ▶ action
 - ▶ decision
 - ▶ learning
 - ▶ interaction

Aim of this session

- ▶ main ROS concepts
- ▶ some ROS tools
- ▶ simulation and robots

Need for a middleware

Robotic system

- ▶ many hardware components:
 - ▶ computers
 - ▶ network
 - ▶ motor controllers
 - ▶ sensors...
- ▶ many software components:
 - ▶ operating system
 - ▶ drivers
 - ▶ control
 - ▶ perception...
- ▶ research

Putting it all together: **middleware**

ROS is a middleware

Robot Operating System

- ▶ open-source middleware
- ▶ development environment
- ▶ communication library and tools
- ▶ packaging system
- ▶ plenty of existing modules
- ▶ community

What ROS is not

Robot Operating System

- ▶ *not* a (computer) operating system
 - ▶ official: Ubuntu Linux
 - ▶ experimental support for: macos, MS Windows, Fedora, Gentoo, Debian...
- ▶ *not* a programming language
 - ▶ official: C++, Python (2 until melodic)
 - ▶ experimental: Java, Lisp, Octave...
- ▶ *not* a hard real-time environment
- ▶ *not* designed for micro-controllers

01

Concepts

Structure

Central concept

- ▶ processing

Structure

Central concept

- ▶ processing

Processing units

- ▶ **node** (unix process)
- ▶ nodelet (thread)

Structure

Central concept

- ▶ processing

Processing units

- ▶ **node** (unix process)
- ▶ **nodelet** (thread)

Organization

- ▶ **package**: compilation unit
 - ▶ node(s)
 - ▶ message definitions
- ▶ **catkin**
 - ▶ build system based on **cmake**
 - ▶ dependency handling
 - ▶ packaging/deployment

Communication

Communication between nodes

- ▶ message
 - ▶ message passing
 - ▶ grouped in topics

Communication

Communication between nodes

- ▶ **message**
 - ▶ message passing
 - ▶ grouped in **topics**
- ▶ **services**
 - ▶ remote procedure call
 - ▶ pair of request and answer messages

Communication

Communication between nodes

- ▶ **message**
 - ▶ message passing
 - ▶ grouped in **topics**
- ▶ **services**
 - ▶ remote procedure call
 - ▶ pair of request and answer messages
- ▶ **actions**
 - ▶ tasks with significant duration
 - ▶ preemptible
 - ▶ continuous feedback

Communication

Communication between nodes

- ▶ **message**
 - ▶ message passing
 - ▶ grouped in **topics**
- ▶ **services**
 - ▶ remote procedure call
 - ▶ pair of request and answer messages
- ▶ **actions**
 - ▶ tasks with significant duration
 - ▶ preemptible
 - ▶ continuous feedback
- ▶ statically typed

Topics

Initialization

- ▶ **publisher**: node declaring writing on a topic
- ▶ **subscriber**: node declaring listening to a topic

Topics

Initialization

- ▶ **publisher**: node declaring writing on a topic
- ▶ **subscriber**: node declaring listening to a topic
- ▶ several publishers/subscribers allowed
- ▶ order irrelevant
- ▶ require a directory

Topics

Initialization

- ▶ **publisher**: node declaring writing on a topic
- ▶ **subscriber**: node declaring listening to a topic
- ▶ several publishers/subscribers allowed
- ▶ order irrelevant
- ▶ require a directory

Communication

- ▶ publisher transmits to each subscriber

Topics

Initialization

- ▶ **publisher**: node declaring writing on a topic
- ▶ **subscriber**: node declaring listening to a topic
- ▶ several publishers/subscribers allowed
- ▶ order irrelevant
- ▶ require a directory

Communication

- ▶ publisher transmits to each subscriber
- ▶ no need for the directory

Services

Initialization

- ▶ *server*: node advertising a service
- ▶ *client*: node asking for a proxy on a given service

Services

Initialization

- ▶ *server*: node advertising a service
- ▶ *client*: node asking for a proxy on a given service
- ▶ require a directory

Services

Initialization

- ▶ *server*: node advertising a service
- ▶ *client*: node asking for a proxy on a given service
- ▶ require a directory

Request

- ▶ client sends a *request* to the server
- ▶ server processes and sends the *answer* back
- ▶ no need for the directory

Actions

Initialization

- ▶ *action server*: node advertising an action
- ▶ *action client*: node asking connection to an action server
- ▶ require a directory

Request and execution

- ▶ client sends a *goal*
- ▶ server starts execution (interrupting current task if needed)
- ▶ server gives goal task reference to client
- ▶ server gives continuous *feedback*
- ▶ task finished: server reports *result*
- ▶ no need for a directory

rosmaster

rosmaster

- ▶ directory
 - ▶ publishers
 - ▶ subscribers
 - ▶ services
 - ▶ actions
- ▶ provides an XML-RPC API
- ▶ *not* a central communication node
- ▶ part of **roscore**
- ▶ nodes know of it through the **ROS_MASTER_URI** shell environment variable

roscore

roscore

- ▶ executable with three roles
 - ▶ rosmaster
 - ▶ parameter server
 - ▶ log aggregator (/rosout)

roscore

roscore

- ▶ executable with three roles
 - ▶ rosmaster
 - ▶ parameter server
 - ▶ log aggregator (/rosout)

Parameter server

- ▶ centralized parameter repository
- ▶ XML-RPC data types

roscore

roscore

- ▶ executable with three roles
 - ▶ rosmaster
 - ▶ parameter server
 - ▶ log aggregator (/rosout)

Parameter server

- ▶ centralized parameter repository
- ▶ XML-RPC data types

Log aggregator

- ▶ republish log messages at lower rate

Launching

Launching a robotic system

- ▶ several/many processes
- ▶ on different computers
- ▶ with specific configuration and parameters

Launch files

- ▶ list of nodes
- ▶ arguments and parameters
- ▶ XML syntax

Transformation frames

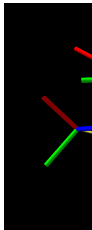
Robot

- ▶ set of rigid bodies

In ROS

- ▶ set of transformation **frames**
- ▶ linked by transformations
- ▶ arranged in a directed tree
- ▶ published on a single **/tf** topic^a
- ▶ rich API to extract information

^aand `/tf_static` too



Transformation frames

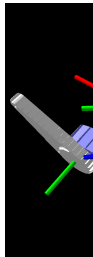
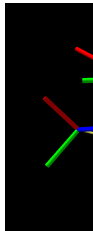
Robot

- ▶ set of rigid bodies

In ROS

- ▶ set of transformation **frames**
- ▶ linked by transformations
- ▶ arranged in a directed tree
- ▶ published on a single **/tf** topic^a
- ▶ rich API to extract information

^aand /tf_static too



Summary of concepts

Structure

- ▶ nodes, in packages

Communication

- ▶ messages
- ▶ services
- ▶ actions
- ▶ peer-to-peer

Launching

- ▶ launch files

Transformations

- ▶ /tf

02

Tools and third party

Runtime inspection

Nodes:

- ▶ list nodes
- ▶ get communication information

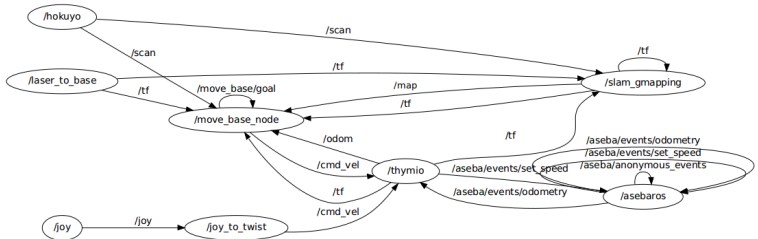
Runtime inspection

Nodes:

- ▶ list nodes
- ▶ get communication information

Connection

- ▶ `rqt_graph`



Runtime inspection

Topics

- ▶ list topics
- ▶ see messages
- ▶ get type information

Runtime inspection

Topics

- ▶ list topics
- ▶ see messages
- ▶ get type information

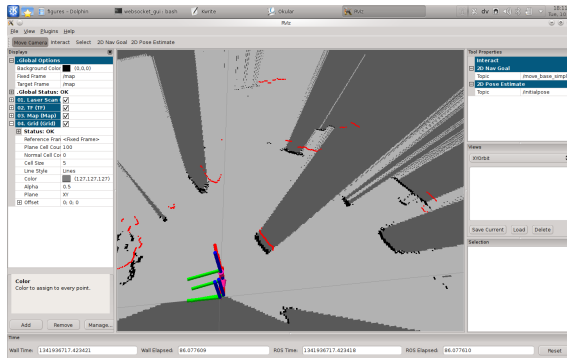
/tf

- ▶ inspect /tf tree
- ▶ compute transformations

Visualization

rviz

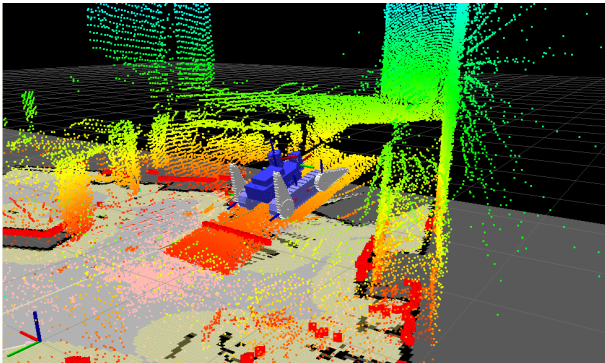
- ▶ full 3D visualization



Visualization

`rviz`

- ▶ full 3D visualization
- ▶ configurable graphical interface



Logging

Logging API

- ▶ different verbosity levels
- ▶ published on `/rosout`
- ▶ `rqt_console` for online inspection
- ▶ automatic dumping to file system for offline analysis

Message	Severity	Node
✖ Couldn't open joystick /dev/input/js0. Will retry every second.	Error	/joy
ℹ Incoming connection from ser=device=/dev/ttyACM1;baud=115200;stop=1;parity=n...	Info	/asebaros
ℹ Subscribed to Topics: scan	Info	/move_base_node
ℹ Requesting the map...\n	Info	/move_base_node
ℹ Still waiting on map...\n	Info	/move_base_node
⚠ Unknown XML node seen in .aesl file: keywords	Warn	/asebaros
ℹ Connected to device with ID: H0707634	Info	/hokuyo
ℹ Starting calibration. This will take up a few seconds.	Info	/hokuyo
ℹ Still waiting on map...\n	Info	/move_base_node
ℹ Loading general config from [/home/steph/.rviz/config]	Info	/rviz

Severity Fatal Error Warn Info Debug

Enabled Regex **From** Message Node Location Topics

Recording

Recording **messages**

- ▶ container: *bagfile*
- ▶ rosbag: generic subscriber

Replaying messages

- ▶ rosbag: generic publisher
- ▶ offline testing of perception pipeline
- ▶ handling of time

Third party tools

Hardware drivers

- ▶ plenty of common sensors
- ▶ many actuators
- ▶ some/many robots

Third party tools

Hardware drivers

- ▶ plenty of common sensors
- ▶ many actuators
- ▶ some/many robots

Software stacks

- ▶ several mapping/SLAM implementations
- ▶ navigation, motion planning
- ▶ 3D perception
- ▶ several simulators...

Third party tools

Hardware drivers

- ▶ plenty of common sensors
- ▶ many actuators
- ▶ some/many robots

Software stacks

- ▶ several mapping/SLAM implementations
- ▶ navigation, motion planning
- ▶ 3D perception
- ▶ several simulators...

Important community

- ▶ researchers
- ▶ some companies (robot/sensors manufacturers)

03

ST5: Simulation and robots

ST5: Robots

Turtlebot2 / Turtlebot3 burger

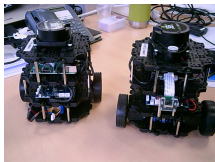
- ▶ 2D ground robot
- ▶ differential drive
- ▶ with 2D laser scanner



ST5: Robots

Turtlebot2 /
Turtlebot3 burger

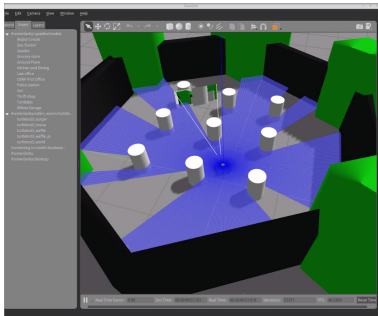
- ▶ 2D ground robot
- ▶ differential drive
- ▶ with 2D laser scanner



ST5: Simulation

Simulation

- ▶ easier and safer than a real robot
- ▶ gazebo simulator
- ▶ turtlebot3 robot



04

Conclusion

Conclusion

ROS

- ▶ open-source middleware for robotics (not the first/only)
- ▶ communication API
- ▶ build environment
- ▶ launch capabilities
- ▶ huge community and plenty of software available
- ▶ transitioning to ROS2

Conclusion

ROS

- ▶ open-source middleware for robotics (not the first/only)
- ▶ communication API
- ▶ build environment
- ▶ launch capabilities
- ▶ huge community and plenty of software available
- ▶ transitioning to ROS2

ST5

- ▶ development in ROS noetic (Python3)
- ▶ use of simulation
- ▶ final tests on real robots



Thanks for your attention
Questions?