Introduction to ROS2

IML

Cédric Buche

ENIB

September 23, 2022

1. Introduction
2. Core modules
3. Side modules
4. How ROS2 works
5. ROS2

- ROS stands for Robot Operating System.
- Collection of tools, libraries, and conventions to simplify the task of creating robot across a wide variety of robotic platforms.
- Stablishing and controlling communication between peripheral modules of a robot : sensors, cameras, physical fingers and etc.
- ROS is open source:
  - It is free
  - There is a large community of contributors.

What uses ROS at the moment?

- Almost all robots you have seen in Academic and to some extend in industry.
- Humanoid Robots
- Manipulators
- Multi-fingered graspers
- Intelligent vehicles : quadrotor helicopters, Autonomous cars ...

Peripheral units

- 1D range finders
- 2D range finders
- 3D Sensors
- Cameras
- Force/Torque/Touch Sensors
- Motion Capture
- Pose Estimation (GPS/IMU)
- RFID

What make ROS outstanding?

- ROS is completely modular :
  - Packages : A collection of Nodes, Messages , services.
  - Nodes: a process that uses ROS framework
  - Messages: Standard definition for passing information between nodes.
  - Stack: Set of multiple package
- ROS is multi-language
- Large set of tools out of box :Standard Robot Messages, Robot Description Language, pose estimation, localization in a map, building a map, and even mobile navigation.
- Integration with other libraries for: Simulation, Image processing and etc.

1. Introduction
2. Core modules
3. Side modules
4. How ROS2 works
5. ROS2

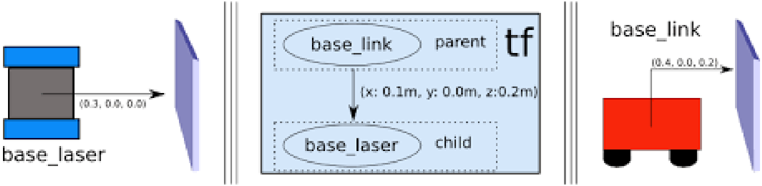
Powerful ROS libraries

Standard Message Definitions

- Example Package : geometry_msgs
-
    - Point
    - Pose
    - Transform
    - ...
- Example of a message structure: Package : sensor_msgs / Message Type : Imu
    - std_msgs/Header header
    - geometry_msgs/Quaternion orientation
    - float64[9] orientation_covariance
    - geometry_msgs/Vector3 angular_velocity
    - float64[9] angular_velocity_covariance
    - geometry_msgs/Vector3 linear_acceleration
    - float64[9] linear_acceleration_covariance

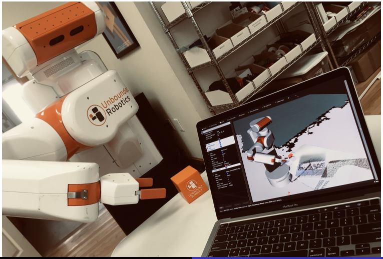
Robot Geometry Library

- Example : "where is the hand, in respect to the head ?"
- Transform library (TF) is a core library of ROS and provides a coordinate tracking system.
- TF listeners listen to the frames and provides a Tree which describes how coordinate systems are related to each other.



ROS visualizer (RVIZ)

- RVIZ is the default 3D visualization tool
- RVIZ is not a "simulator".
- RVIZ can show data that it has a plugin for displaying (DisplayTypes) and has been published by nodes:
  - Axes : Displays a set of Axes
  - Camera: Creates a new rendering window from the perspective of a camera
  - Map : Displays a map on the ground plane
  - Pose : Draws a pose as an arrow or axes.
  - ...



Robot Description Language (URDF)

- Describe a robot in a machine readable format.
- URDF is an XML file describing following physical properties:
  - Main parts: cylinder, box, length, radius, . . .
  - Joints : continuous joints, prismatic joint, planar joint, Joint Dynamics (friction, damping) , Inertia
- Used by different tools for simulation, visualization and motion planning:
  - Rviz
  - Gazebo
  - Moveit
  - Stage

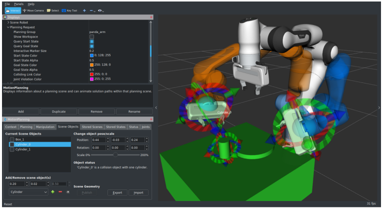
1. Introduction
2. Core modules
3. Side modules
4. How ROS2 works
5. ROS2

GAZEBO

- Simulation environment and supports many robots and sensors.
- Developing and test a node without a physical robot.

Moveit

- The most widely used open-source software for manipulation, motion planning and analyzing of robot interaction with environment.
- Collision checking
- Integrated kinematics
- Motion planning
- Integrated perceptions about environment
- Execution and monitoring
- Interactive



Scene Robot
Planning Request

Planning Group	panda_arm
Show Workspace	□
Query Start State	☑
Query Goal State	☑
Interactive Marker Size	0.2
Start State Color	0; 128; 255
Start State Alpha	0.5
Goal State Color	250; 128; 0
Goal State Alpha	0.5
Colliding Link Color	255; 0; 0
Joint Violation Color	255; 0; 255

MotionPlanning
Displays information about a planning scene and can animate solution paths within that planning scene.

Current Scene Objects
- Box_1
- Cylinder_0
- Cylinder_1

Change object pose/scale
Position: 0.44 -0.03 0.24
Rotation: 0.00 0.00 0.00
Scale:0% 200%

Object status
'Cylinder_0' is a collision object with one cylinder.

Add/Remove scene object(s)
0.20 0.02 0.50

Scene Geometry

OpenCV

- The most powerful image processing library
- ROS uses sensor_msgs/Image message and OpenCV need matrices for images : Conversion by cv_bridge stack.

1. Introduction
2. Core modules
3. Side modules
4. How ROS2 works
5. ROS2

Topics

- Node : a process that uses ROS framework.
- Message: Standard definitions for transferring data between nodes (*.msg)
- Topic: Mechanism of transferring data between nodes.
- Publisher: A node which produce message and publish them.
- Subscriber: A node which receives the messages.

Topics : Workflow

- Node A publish a message to a topic
- All nodes which are subscribed to that topic, will receive the message.

Services

- ▷ Service-Client is way to retrieve the data immediately instead of waiting for a message to be published.
- ▷ A node provides a service , the client node call the service by sending request message.
- ▷ services are defined in *.srv files : Request / Response
- ▷ one-to-one

Actions

- Similar to service calls, but provide possibility to
  - Cancel the task (preempt)
  - Receive feedback on the progress
- Best way to implement interfaces to time-extended, goal-oriented behaviors
- action are defined in *.action files : Goal / Result / Feedback

Time

- ROS uses the PC's system clock as time source (wall time)
- For simulations or playback of logged data, it is convenient to work with a simulated time (pause, slow-down etc.)
  - rosparam set use_sim_time true
  - ros::Time
  - ros::Duration
  - ros::Rate
  - ros::WallTime, ros::WallDuration, and ros::WallRate

Bag

- A bag is a format for storing message data
- Binary format with file extension *.bag
- Suited for logging and recording datasets for later visualization and analysis

1. Introduction
2. Core modules
3. Side modules
4. How ROS2 works
5. ROS2

ROS2

- ROS 1 designed mainly for research
- Never really designed for Real-time
- ROS1 - master-follower architecture
- ROS2 - replaced by Data Distribution Service (DDS)
- DDS is a distributed service that does the discovery, marshalling and transport in the background

Introduction to ROS2

IML

Cédric Buche

ENIB

September 23, 2022