IN3140 - Introduction to Robot Operating System

April 13, 2023

Jørgen Nordmoen & Justinas Mišeikis

Adel Baselizadeh

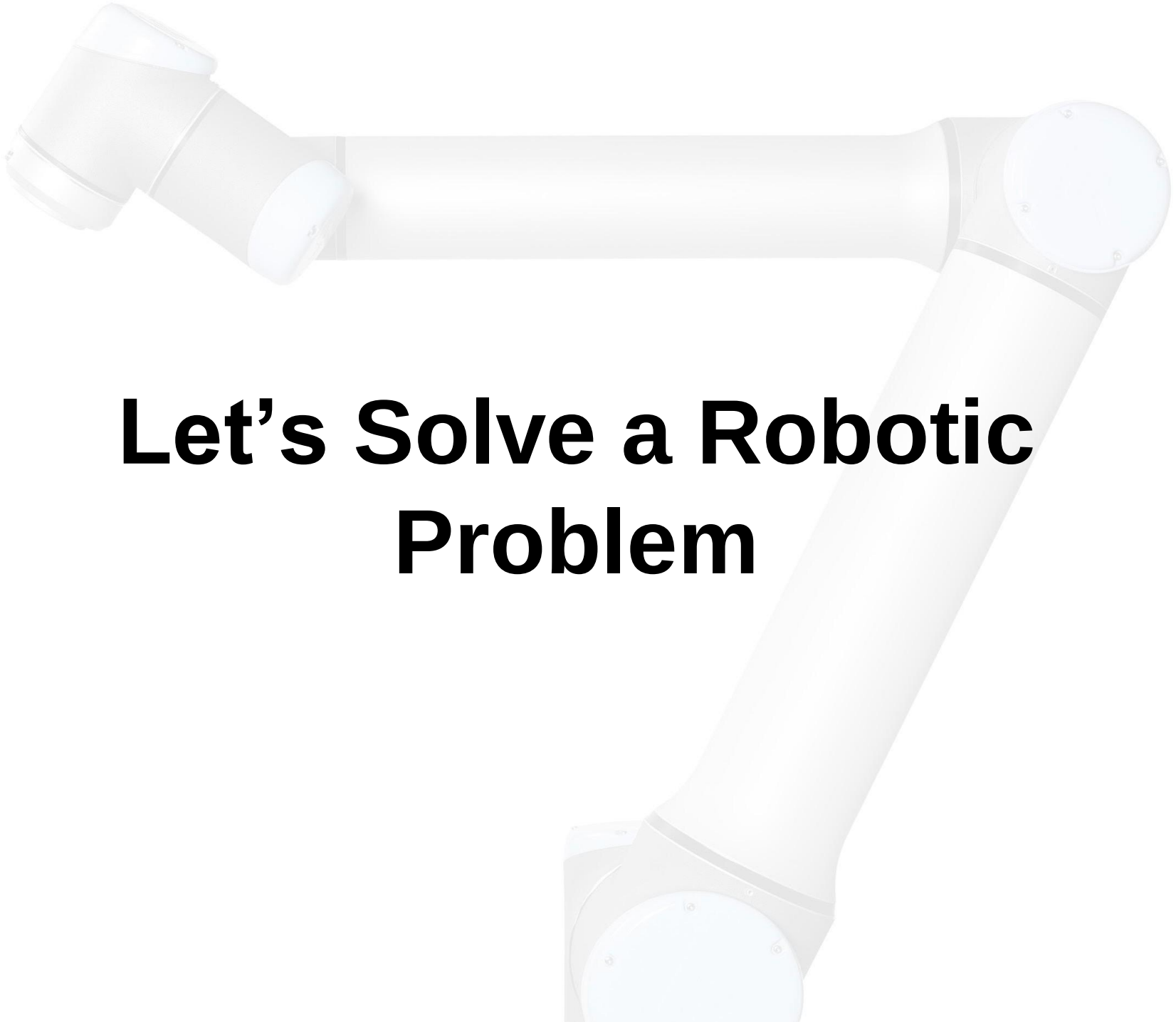
adelb@ifi.uio.no

Side Note

This is an overview lecture, but do expect exam question on ROS topics. Please pay more attention to the slides marked with “Study Material” such as below.

All practical parts are considered study material! You will not need to code in anything in the exam, but you have to understand the concepts.

Study Material



Let’s Solve a Robotic Problem

Collision-free motion planning for a robot manipulator

Encoders

Motors

Control Pendant

Processor

RGB-D Sensor

Encoders

Motors

Control Pendant

Robot Controller

Motor controllers

Inverse Kinematics

GUI

Trajectory Planner

Image Processing Algorithm

Collision Detection

What a mess!

How can we deal with it?

ROS

ROS is an open-source, meta-operating system

Built on top of the operating system and allows different processes to communicate with each other at runtime.

ROS is an open-source, meta-operating system

pointcloudlibrary

What’s so special about ROS?

- 740+ Robotic platforms officially support ROS (ROS metrics report 2022) http://wiki.ros.org/Robots
- Thousands of ready to use algorithms
- Efficient, so it can be used for actual products, not just prototyping
- Runs on Ubuntu, also ARM Processors
  - Could be run on Windows and OS X through containers like Docker and Singularity.
- Parallelisation and networking made easy, can use multiple machines simultaneously

Current Robotics Job Ads

Experience in developing robotics software, e.g., kinematics/dynamics, control of actuators/sensors, distributed systems. Provable proficiency in C, C++ and experience in at least another programming language (eg. python, java). Hands-on experience in robotics middlewares, e.g. ROS, YARP, Orocos

Must haves: Excellent general structured problem-solving and software architecture skills. Demonstrated strong software engineering and design fundamentals Fluency in C/C++. Experience with path planning and navigation. Experience in ROS and simulation environments. Experience developing in a Linux environment.

Robotics Specialist. Core tasks are the development of algorithms for grasp calculation and the improvements of existing solutions. Skills: 3+ years C++ development, Machine Learning, ROS, Ubuntu/Linux, PCL

The candidate must be a proficient user of C/C++ and ROS and any relevant computer vision library (e.g., ViSP, OpenCV, PCL). Scientific curiosity, large autonomy and ability to work independently are also expected.

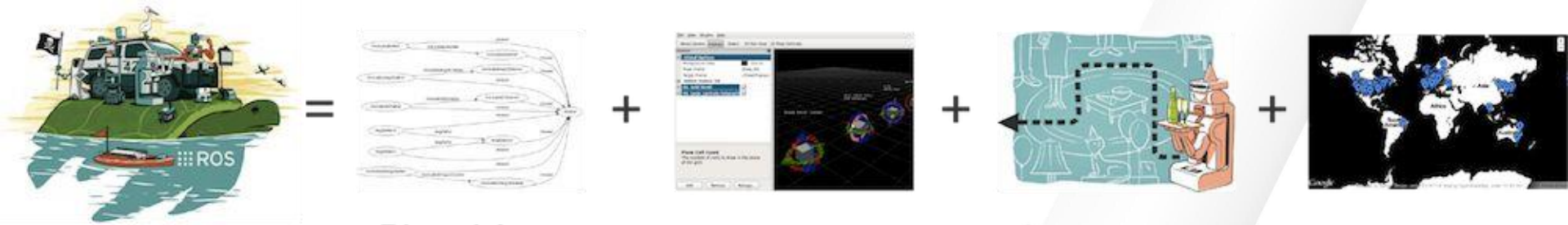
The technical Requirements: C++ or Python programming knowledge in Linux, Knowledge of ROS. You have to be able to write ROS programs, debug and find your way Knowledge of Gazebo.

Roboticist: Path-planning Specialist

- Own the navigation costmaps area and implement various data processing algorithms
- Have experience and knowledge on 2D data processing for motion planning, e.g. Fast Marching Methods
- Have experience with state-of-the-art path-planning approaches, e.g. RRT*
- Very good C++ skills (ROS, OpenCV, Linux)

One of many sources: http://www.theconstructsim.com/ros-jobs/

What is ROS?



http://www.ros.org/about-ros/

ROS

Plumbing

Tools

Capabilities

Ecosystem

Let’s see how it works!

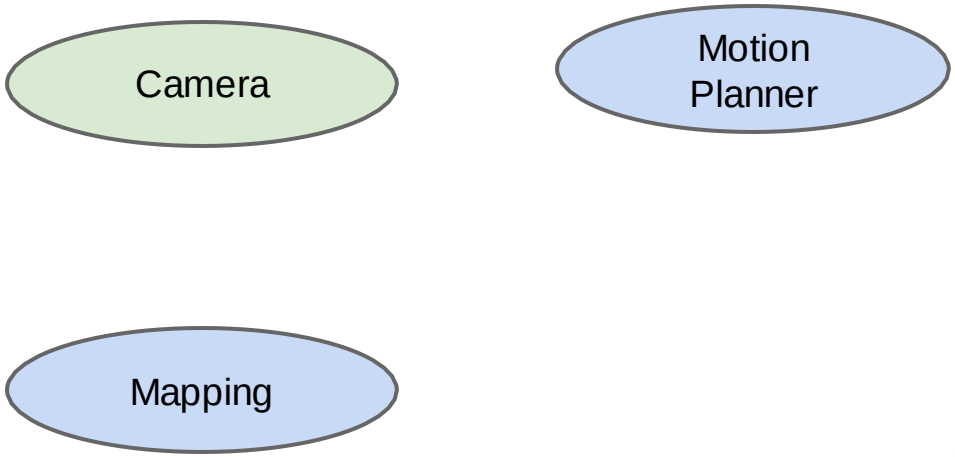
Plumbing

ROS provides publish-subscribe messaging infrastructure designed to support the quick and easy construction of distributed computing systems.

Nodes

http://wiki.ros.org/ROS/Tutorials/UnderstandingNodes

Nodes are processes that perform computation, “executables”. Each node performs a specific processing part, usually a part of the algorithm.



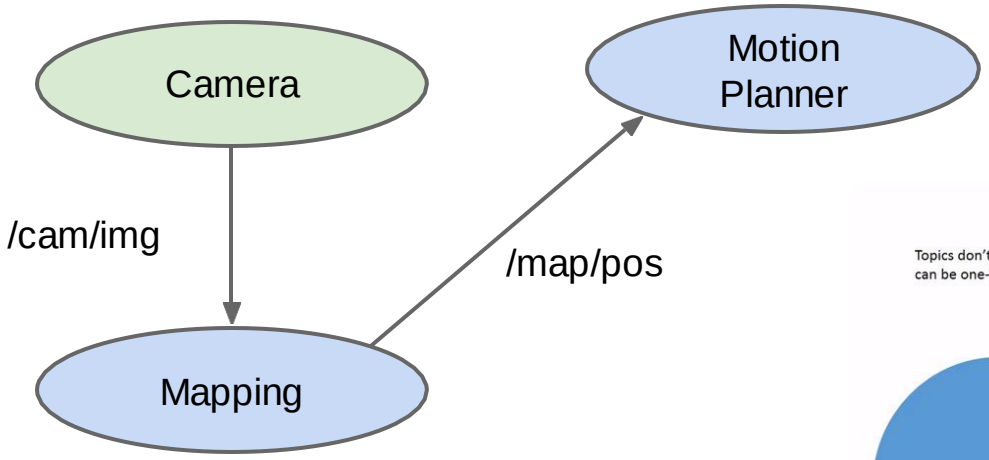
Study Material - highly recommend to check the URL

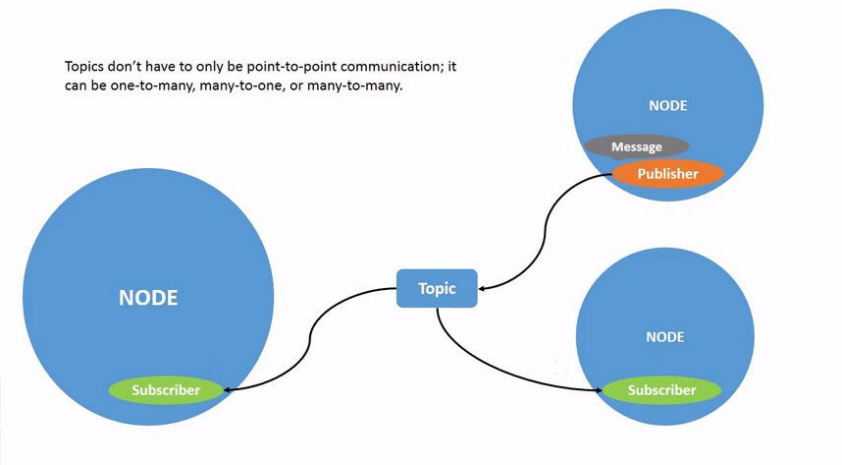
Topics

http://wiki.ros.org/ROS/Tutorials/UnderstandingTopics

Topics are streams of data with publish / subscribe semantics.

They are uniquely identifiable by its name. Nodes can publish and subscribe to topic in order to transfer data.





Study Material - highly recommend to check the URL

Messages

http://wiki.ros.org/ROS/Tutorials/CreatingMsgAndSrv

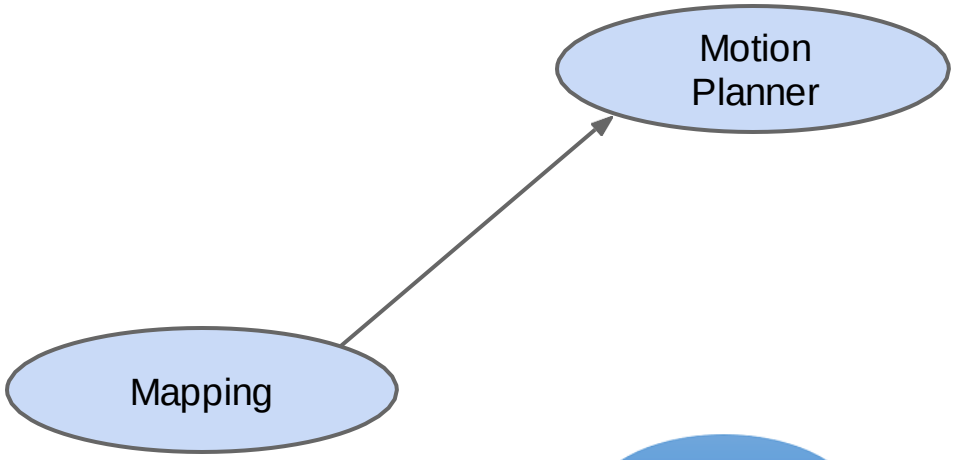
A message is simply a data structure, comprising typed fields.

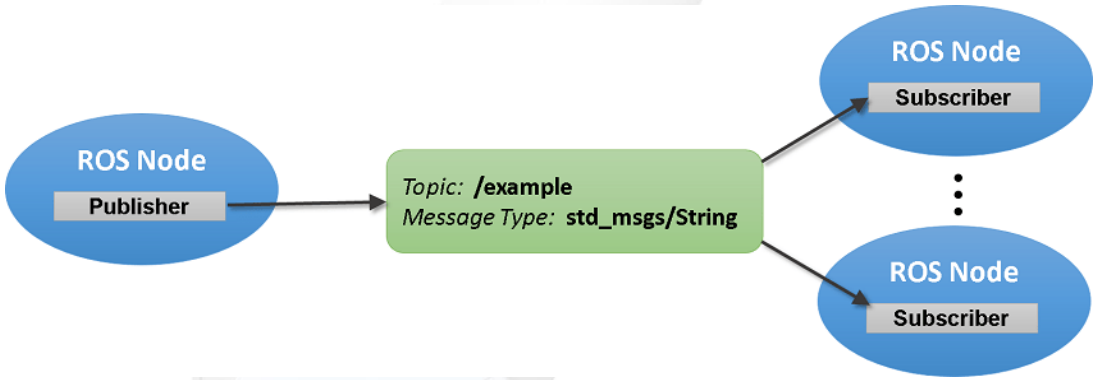
Language agnostic data representation. C++ can talk to Python.

Messages are sent on defined topics.

File: Pos.msg

```
string robotName
uint32 posX
uint32 posY
uint32 goalX
uint32 goalY
```

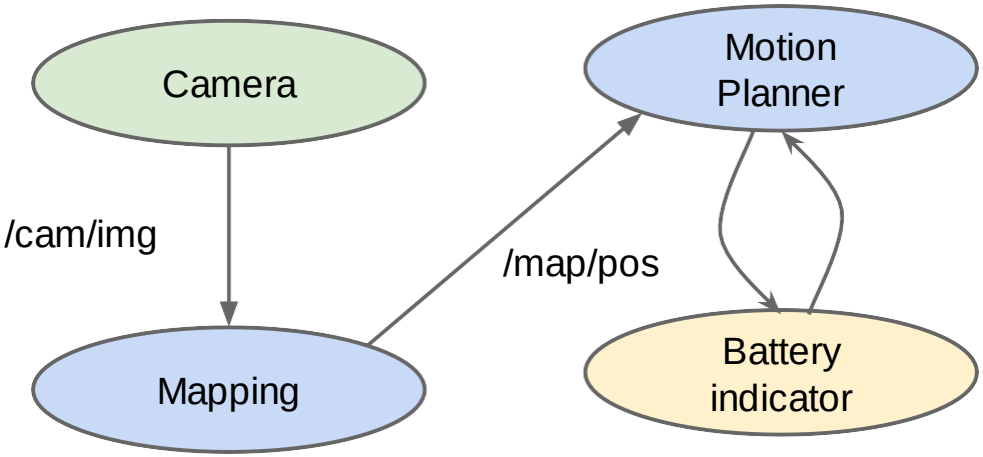


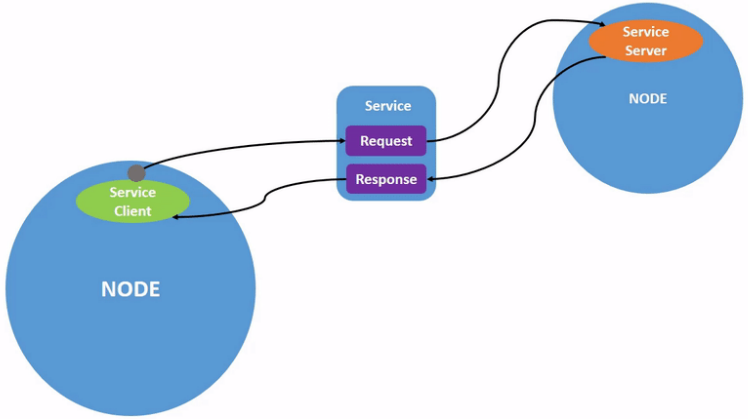


Services

http://wiki.ros.org/ROS/Tutorials/UnderstandingServicesParams

Request / reply is done via services, which are defined by a pair of message structures: one for the request and one for the reply.



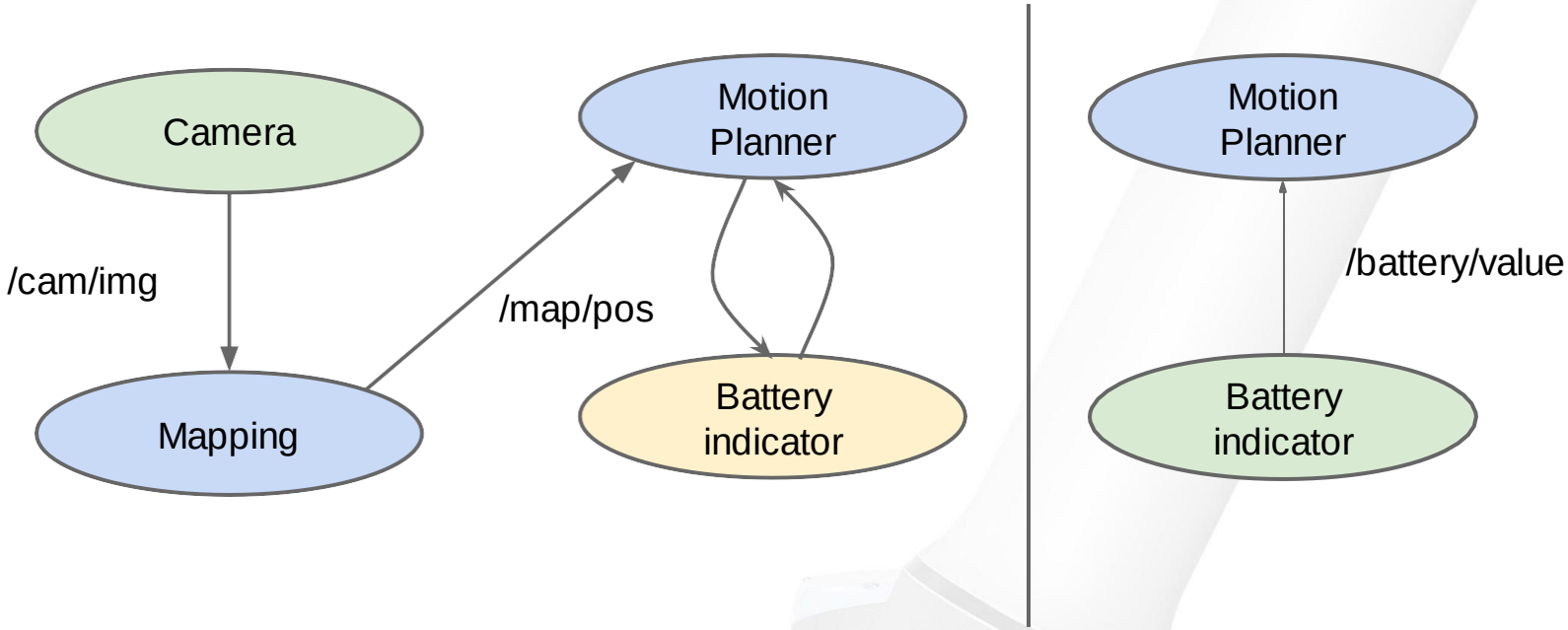


Study Material - highly recommend to check the URL

Services

http://wiki.ros.org/ROS/Tutorials/UnderstandingServicesParams

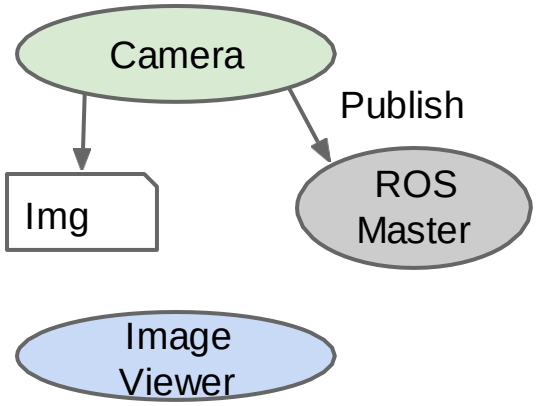
Request / reply is done via services, which are defined by a pair of message structures: one for the request and one for the reply.



Study Material - highly recommend to check the URL

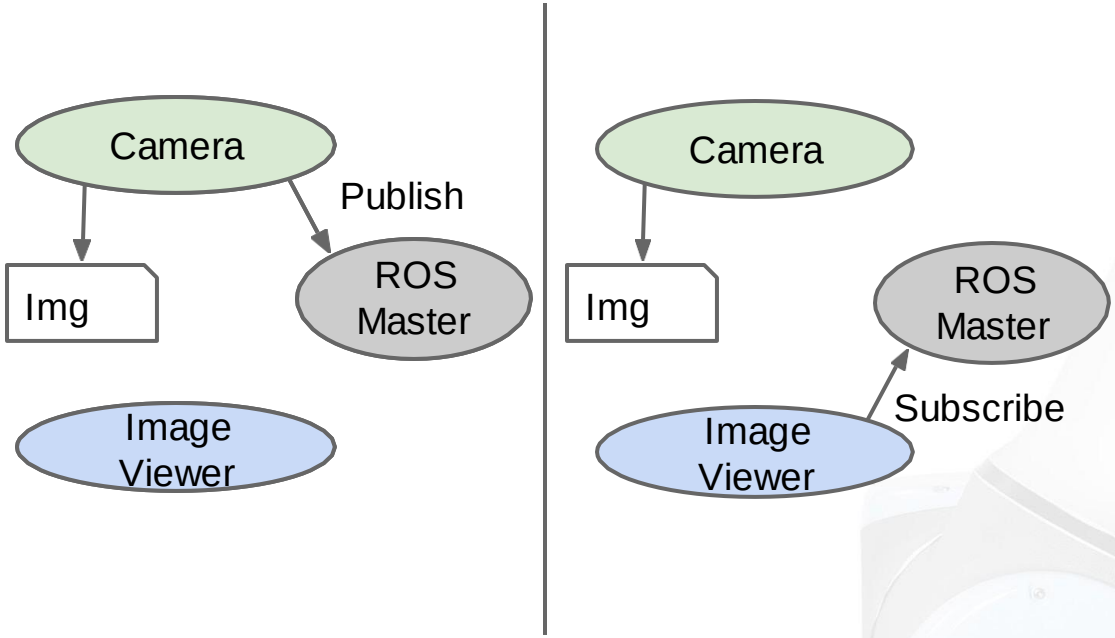
ROS Master

The ROS Master provides name registration and lookup to nodes. Without the Master, nodes would not be able to find each other, exchange messages, or invoke services.



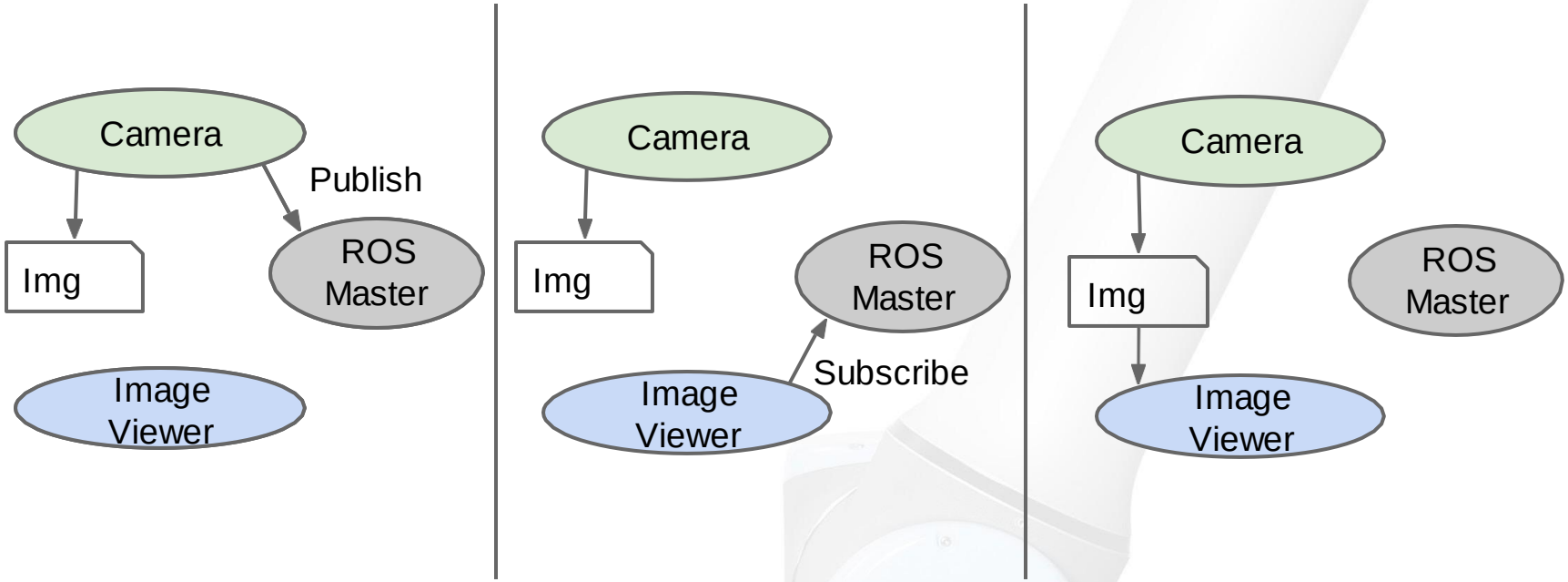
ROS Master

The ROS Master provides name registration and lookup to nodes. Without the Master, nodes would not be able to find each other, exchange messages, or invoke services.



ROS Master

The ROS Master provides name registration and lookup to nodes. Without the Master, nodes would not be able to find each other, exchange messages, or invoke services.

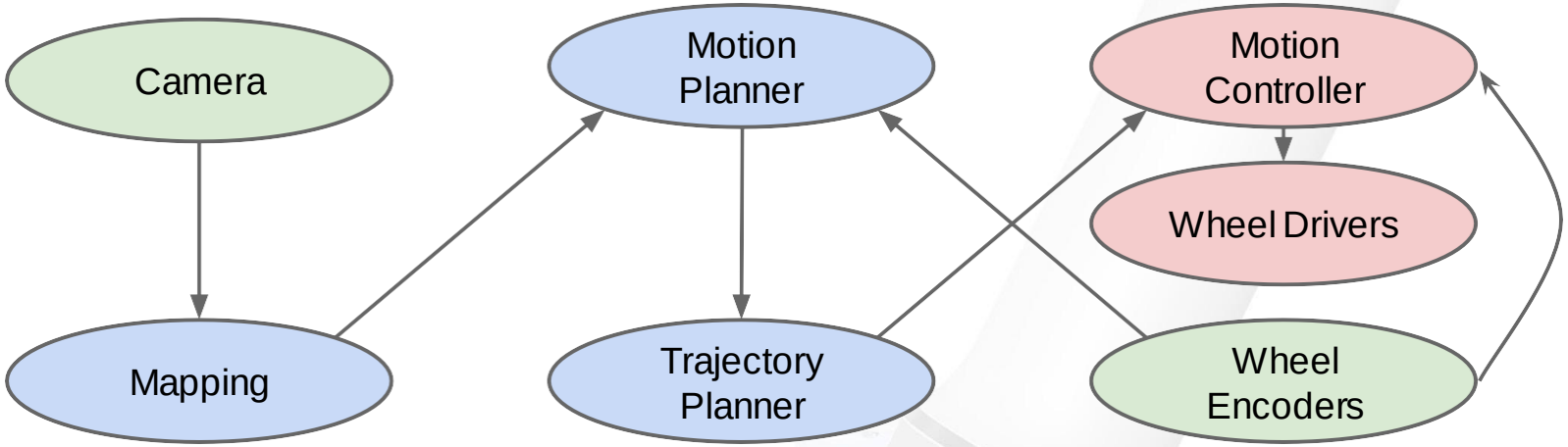


Example System - Mobile Robot

Green - Sensors

Blue - Planning algorithms

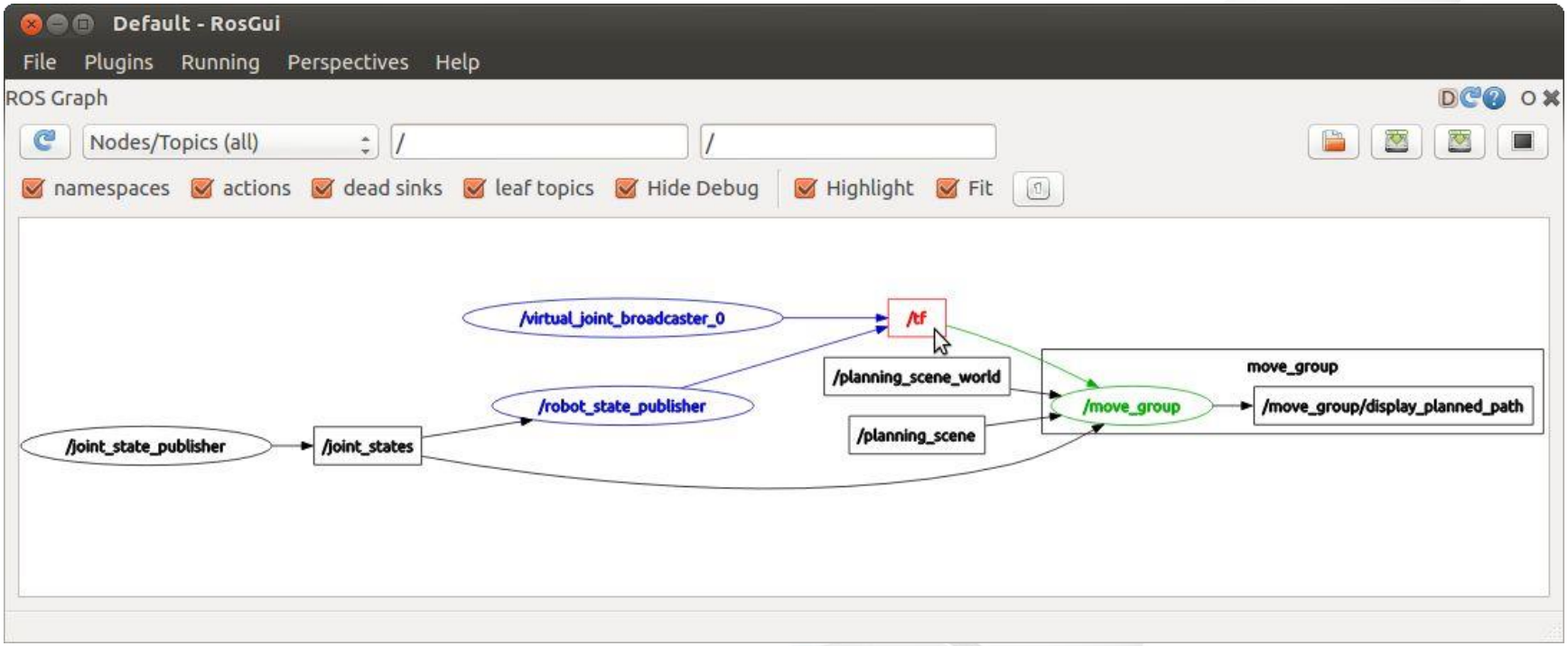
Red - Hardware integration



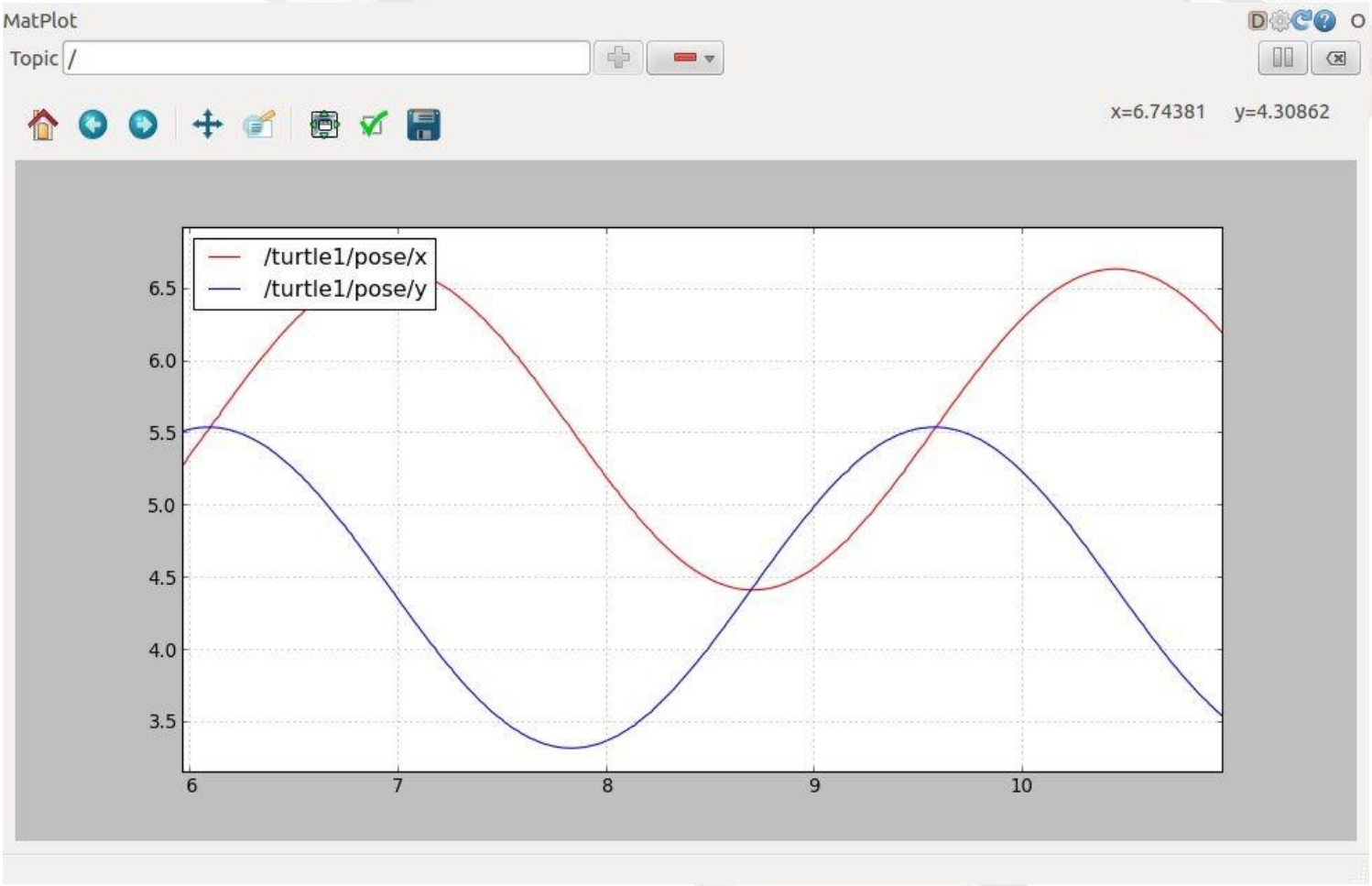
Tools

www.wiki.ros.org/Tools

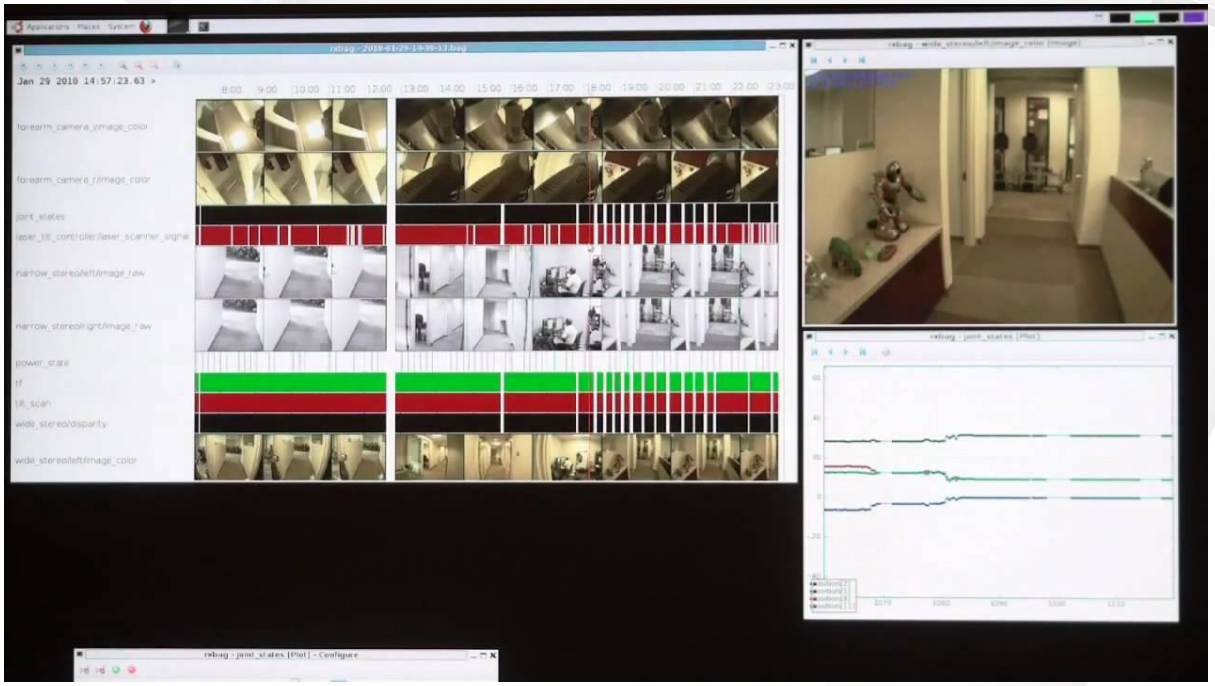
System Visualisation: rqt_graph



Live Plotting: rqt_plot



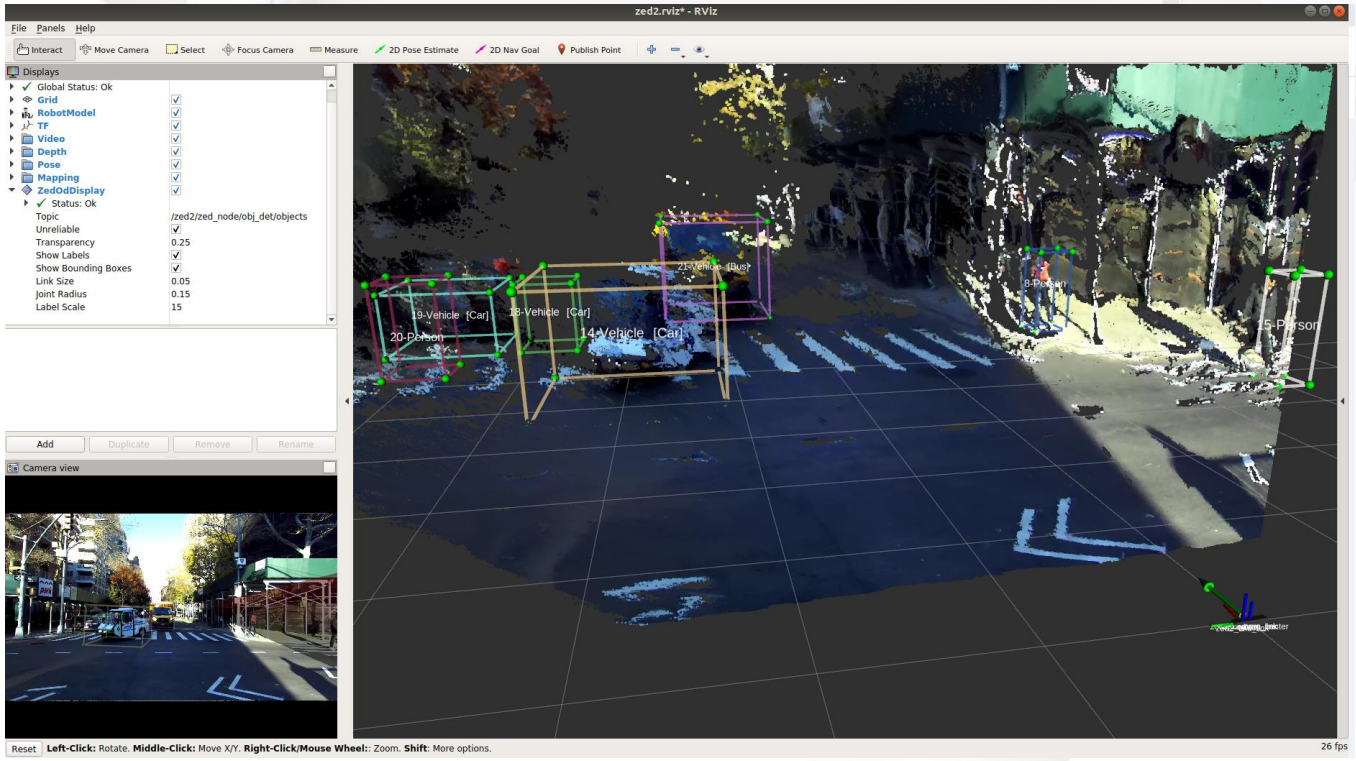
Logging and Visualization Sensor Data: rosbag and rqt_bag



This is a set of tools for recording from and playing back to ROS topics. It is intended to be high performance and avoids deserialization and reserialization of the messages.

https://www.youtube.com/watch?v=pwlbArh_neU

3D Visualisation: RVIZ



RVIZ is a ROS graphical interface that allows you to visualize a lot of information, using plugins for many kinds of available topics.

https://www.youtube.com/watch?v=i--Sd4xH9ZE

Capabilities

https://www.youtube.com/watch?v=mDwZ21Zia8s

Review of Technical Capabilities

http://gazebosim.org/

Gazebo basics, Gazebo files

To run a Gazebo simulation you need:

- **A world file**: A file with extension `.world` that contains all the elements in a simulation, including robots, lights, sensors, and static objects, formatted using the Simulation Description Format (SDF).

https://sir.upc.edu/projects/rostutorials/8-gazebo_basics_tutorial/index.html#basics-label

http://gazebosim.org/tutorials/?tut=ros_urdf#Sharingyourrobotwiththeworld

Gazebo basics, Gazebo files

To run a Gazebo simulation you need:

- **Model files**: SDF files used to describe objects and robots. Models are included in world files using the include tag:

The components of a model are:

- **Links**: A link contains the physical properties of one body of the model.
- **Joints**: A joint connects two links.

http://gazebosim.org/tutorials/?tut=ros_urdf#Sharingyourrobotwiththeworld

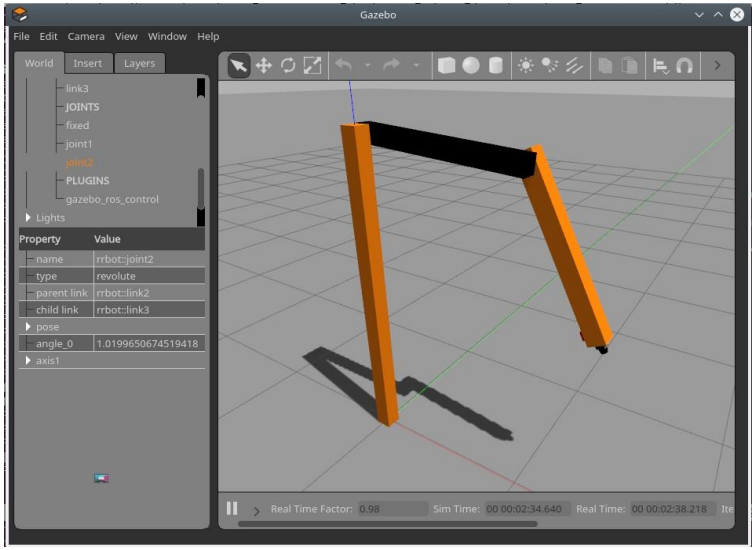
GAZEBO rrbot example

RRBot, or "Revolute-Revolute Manipulator Robot", is a simple 3-linkage, 2-joint arm.

```
cd ~/catkin_ws/src/
git clone https://github.com/ros-simulation/gazebo_ros_demos.git
cd ..
catkin_make

rosed rrbot_description rrbot.xacro

roslaunch rrbot_gazebo rrbot_world.launch
```



https://sir.upc.edu/projects/rostutorials/8-gazebo_basics_tutorial/index.html#basics-label

http://gazebosim.org/tutorials/?tut=ros_urdf#Sharingyourrobotwiththeworld

Review of Technical Capabilities

https://moveit.ros.org

https://www.youtube.com/watch?v=vAeEEoxVhAo

MoveIt Motion Planning

MoveIt! includes a variety of robust and state-of-the-art motion planners:

- Sampling-based motion planning algorithms (OMPL)
- Covariant Hamiltonian optimization for motion planning (CHOMP)
- Stochastic Trajectory Optimization for Motion Planning (STOMP)
- TrajOpt is a sequential convex optimization algorithm

MoveIt Constraints

You can specify the following kinematic constraints:

- **Position constraints** – restrict the position of a link to lie within a region of space
- **Orientation constraints** – restrict the orientation of a link to lie within specified roll, pitch or yaw limits
- **Visibility constraints** – restrict a point on a link to lie within the visibility cone for a particular sensor
- **Joint constraints** – restrict a joint to lie between two values
- **User-specified constraints** – you can also specify your own constraints with a user-defined callback.

MoveIt Scene Collision Objects

You can specify the following kinematic constraints:

- static objects (objects rigidly fixed on the robot workspace)
- dynamic objects (objects with which the robot can interact, i.g. pick, place, push ...etc)
- Moveit Collision Objects published through moveit_msgs/CollisionObject messages

MoveIt How to Use it?!

To simulate and play around with Universal Robot UR5:

- Have ROS installed.
- Create a work-space: `mkdir -p ~/ws_moveit/src`
- From ROS-Industrial GitHub Page:

```
git clone -b melodic-devel https://github.com/ros-industrial/universal_robot
```

- Install any new dependencies that may be missing:

```
rosdep install -y --from-paths . --ignore-src --rosdistro noetic
```

- Re-build and re-source the workspace and enjoy:

`catkin_make` and `source devel/setup.bash`

```
roslaunch ur5_moveit_config moveit_rviz.launch
```

ros-planning.github.io/moveit_tutorials/doc/realtime_servo/realtime_servo_tutorial.html?highlight=ur5


Ecosystem

ROS Statistics

October 2021 - October 2022

October 2022

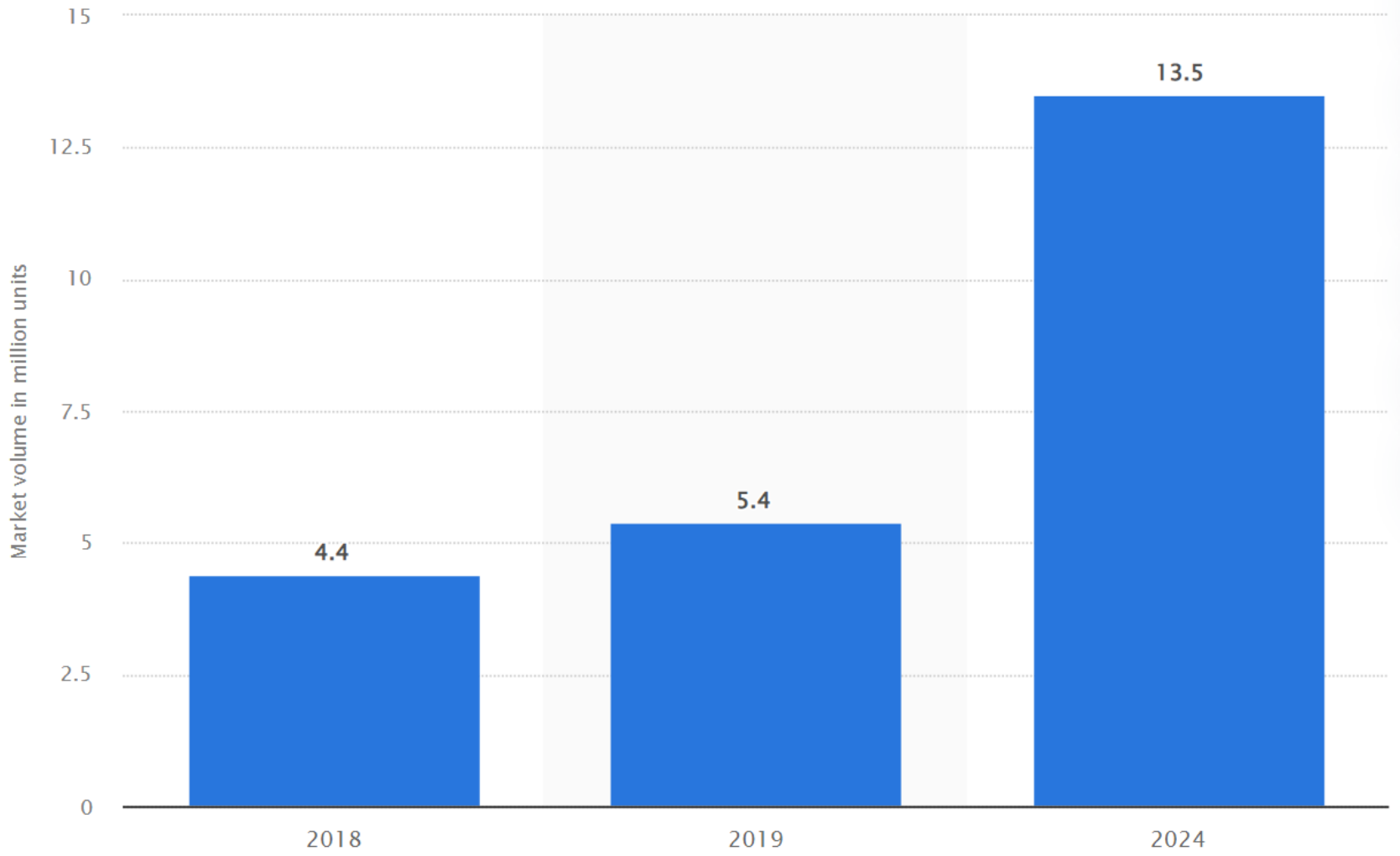
Debs Downloads, Visitors, Packages

	October 2021	October 2022	YoY Change
Total Unique Packages (packages across versions, distros, syncs, etc)	89,144	146,745	+64.62%
Different Packages	19,562	23,614	+20.71%
Deb Downloads	45,100,525	49,078,176	+8.82%
Unique Visitors	789,956	767,632	-2.82%

ROS™

Worldwide User Base

ROS-based robot market volume worldwide between 2018 and 2024 (www.statista.com)



Thank You!

Any Questions?