The Robot Operating System

Bence Magyar
Edinburgh Centre for Robotics
Heriot-Watt University
Edinburgh, UK

Winter School on Humanoid Robot Programming
Santa Margherita Ligure
6-15 February 2018

Bence Magyar

HERIOT WATT UNIVERSITY

EDINBURGH CENTRE FOR ROBOTICS

PhD topic:
“Generalising skills for robotic manipulation”

- Learning from Demonstrations for Motion Planning
- Dual-arm Dynamical Movement Primitives

ros_control

ros-planning

MoveIt!

Google Summer of Code

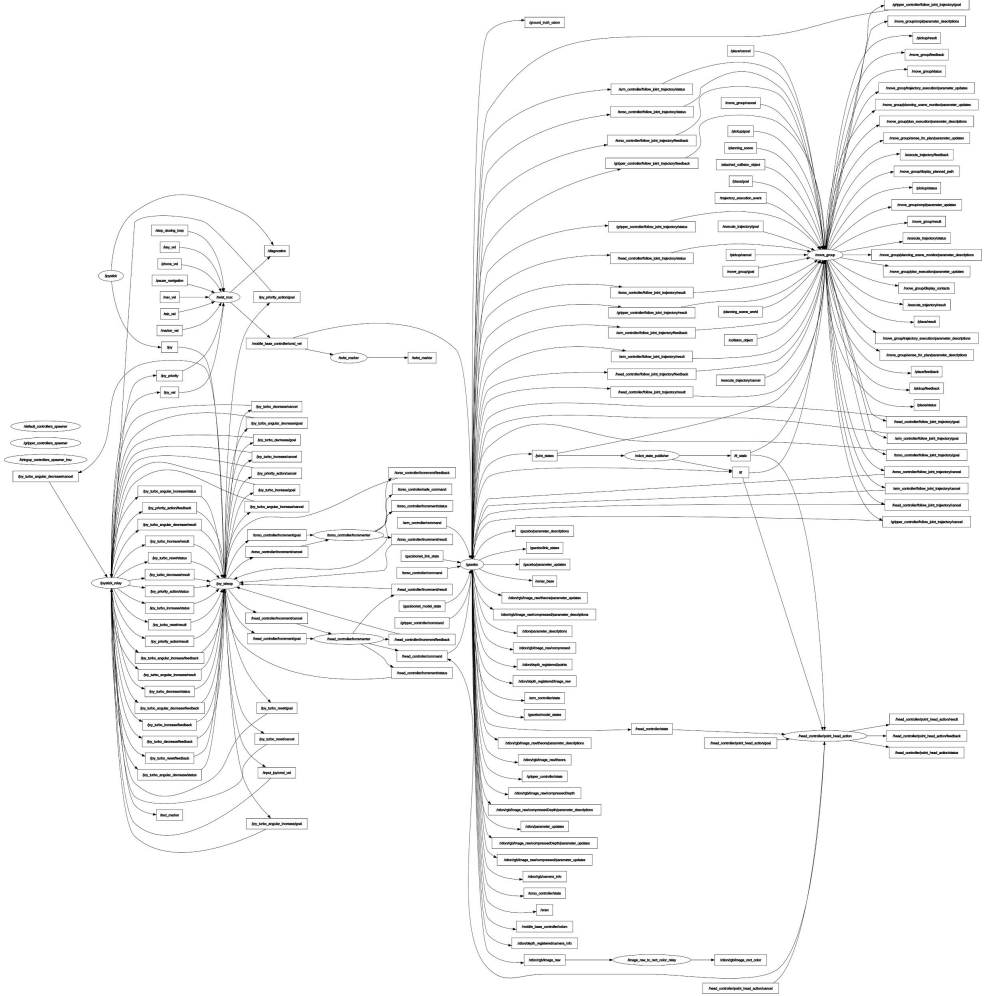
OpenCV

ros-teleop


ROS

The Robot Operating System

ROS



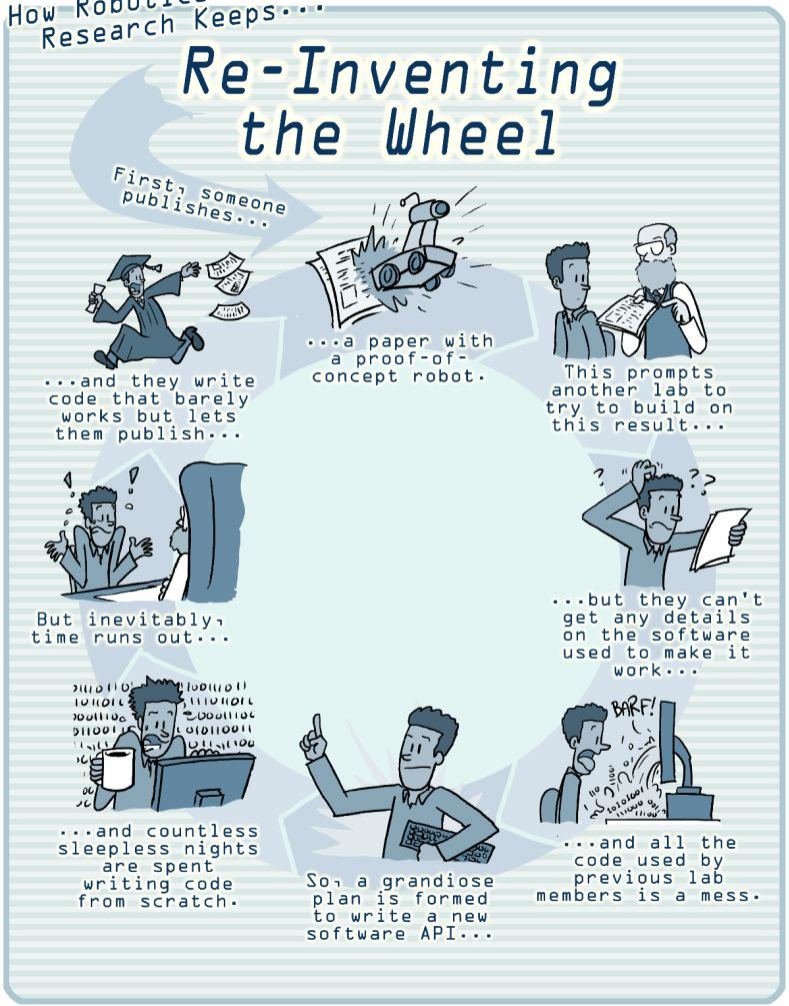
VVV18, 6-15 February, 2018

ROS

The Robot Operating System

- Neither an operating system nor only suited for robots
- A flexible software framework that works well for robotics
- Started by Willow Garage in 2007, now OSRF and community
- Aims to encourage software reuse, collaboration in academia and industry
- Includes communications infrastructure, robot-specific libraries, high-level capabilities and ecosystem.

ROS


How Robotics Research Keeps...
Re-Inventing the Wheel
First, someone publishes...
...a paper with a proof-of-concept robot.
This prompts another lab to try to build on this result...
...but they can't get any details on the software used to make it work...
...and all the code used by previous lab members is a mess.
So, a grandiose plan is formed to write a new software API...
...and countless sleepless nights are spent writing code from scratch.
But inevitably, time runs out...
...and they write code that barely works but lets them publish...
BARF!

ROS

The Robot Operating System

ROS 2017 Metrics:

- community is huge and include research labs, industrial and service robotics
- 3,000+ public packages
- 300+ developers
- wiki.ros.org annual unique visitors 500,000+
- robots.ros.org lists 100+ robots
- ROS paper cited 3704 times
- Books: http://wiki.ros.org/Books
- Annual ROSCon (475+ attendees, aligns with IROS)

50%

33%

ROS






VVV18, 6-15 February, 2018

ROS

The Robot Operating System

Industrial collaboration

ALDEBARAN
SoftBank Group
NEOBOTIX

ROS industrial

Robotnik

rethink robotics

Open Source Robotics Foundation

Fraunhofer IPA

OMRON AUTOMATION & SAFETY

YUJIN ROBOT

fetch robotics

Willow Garage

NASA

CLEARPATH ROBOTICS™

Erle Robotics

GM General Motors

KINOVA ROBOTICS

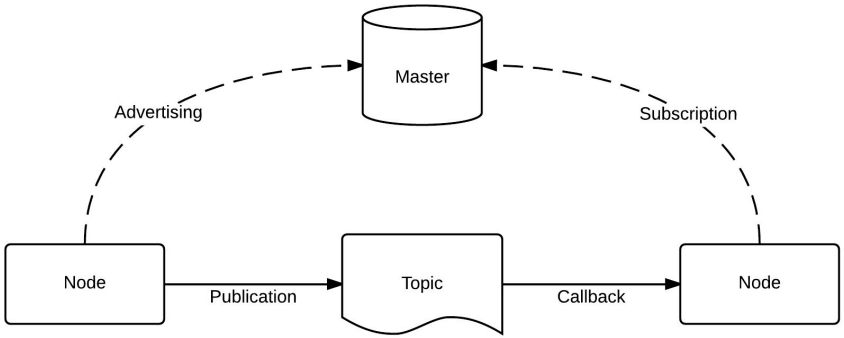
PAL ROBOTICS

CRUISE

ROS

Communications infrastructure

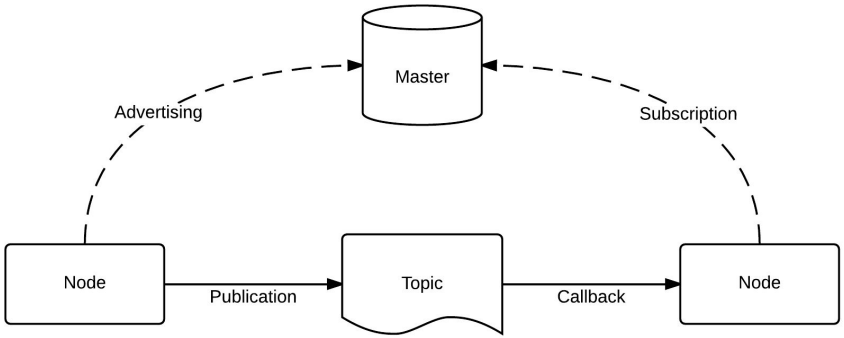
- Master
- Nodes
- Topics
- Services
- Parameters
- Multi-computer



ROS

Communications infrastructure

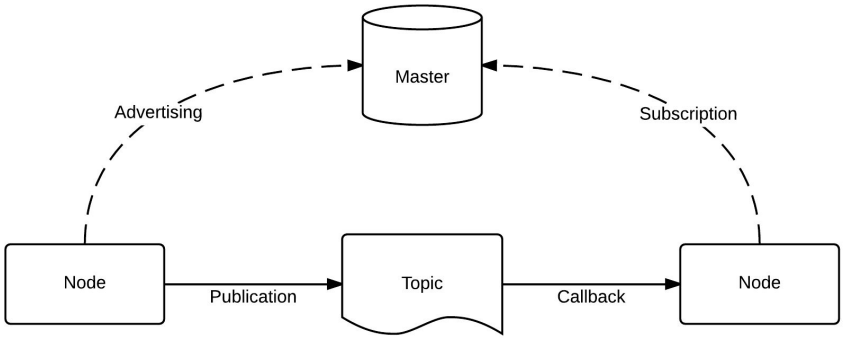
- Master
  - Global name server for
    - nodes
    - topics
    - services
  - Host for parameter server



ROS

Communications infrastructure

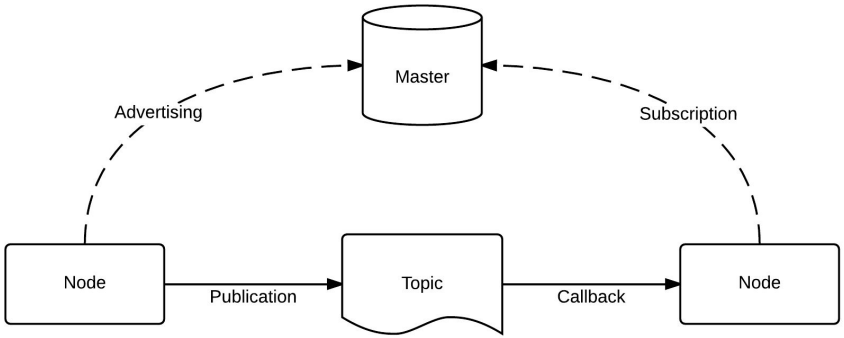
- Master
- Nodes
  - Registered names on master
  - May communicate via topics or services
  - May use parameters



ROS

Communications infrastructure

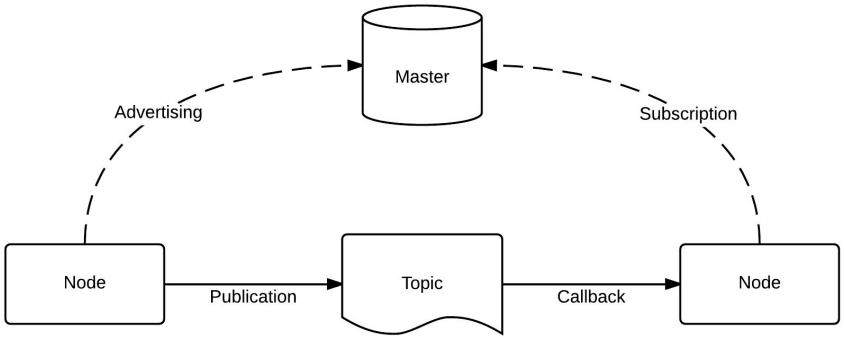
- Master
- Nodes
- Topics
  - P2P communication channels
  - Resolved via names on the master
  - Publish/Subscribe model
  - Messages are handled via callbacks



ROS

Communications infrastructure

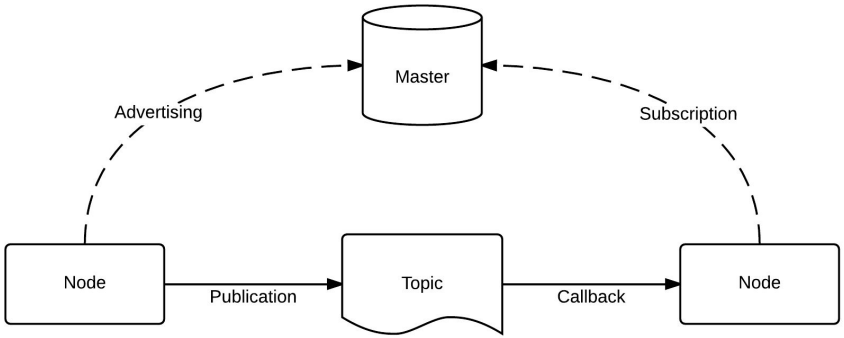
- Master
- Nodes
- Topics
- Services
  - Synchronous RPC
  - Request / Response



ROS

Communications infrastructure

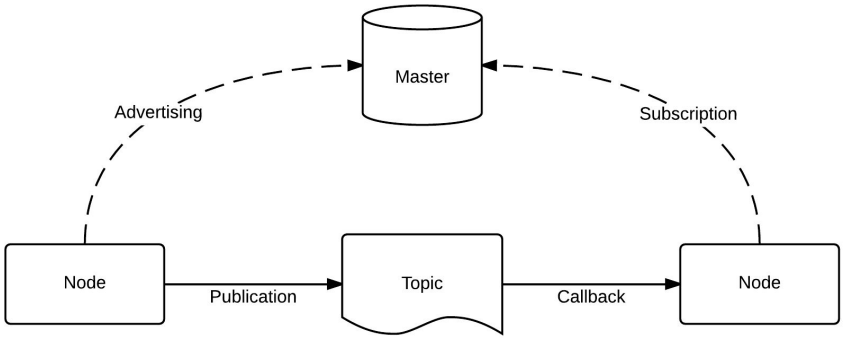
- Master
- Nodes
- Topics
- Services
- Parameters
  - Named dictionary structures
  - JSON/YAML compatible
  - Globally available



ROS

Communications infrastructure

- Master
- Nodes
- Topics
- Services
- Parameters
- Multi-computer
  - by redirecting default master IP
  - ROS_MASTER_URI env var



ROS

Messages

std_msgs/Header

uint32 seq
time stamp
string frame_id

sensor_msgs/JointState

std_msgs/Header header
string[] name
float64[] position
float64[] velocity
float64[] effort

sensor_msgs/PointCloud2

std_msgs/Header header
uint32 height
uint32 width
sensor_msgs/PointField[] fields
bool is_bigendian
uint32 point_step
uint32 row_step
uint8[] data
bool is_dense

geometry_msgs

Accel
AccelStamped
AccelWithCovariance
AccelWithCovarianceStamped
Inertia
InertiaStamped
Point
Point32
PointStamped
Polygon
PolygonStamped
Pose
Pose2D
PoseArray
PoseStamped
PoseWithCovariance
PoseWithCovarianceStamped
Quaternion
QuaternionStamped
Transform
TransformStamped
Twist
TwistStamped
TwistWithCovariance
TwistWithCovariance
Stamped
Vector3
Vector3Stamped
Wrench
WrenchStamped

Verified by MD5 sum at runtime

Ecosystem

Package-based system, where a package contains

- package.xml
- CMakeLists.txt

And may contain:

- source files
- message definitions
- launch files
- configuration files
- other

Supported languages:

C++, Python, Lisp

Node.js, Java

Ruby, C#, Julia, Matlab

Supported operating systems:
Ubuntu, Debian, OSX

(any Unix-based OS with some work)

Ecosystem

- Package-based system
- catkin
- roslaunch
- rosinstall

Supported languages:

C++, Python, Lisp

Node.js, Java

Ruby, C#, Julia, Matlab

Supported operating systems:
Ubuntu, Debian, OSX

(any Unix-based OS with some work)

ROS ⟷ YARP

roscore / ros master	yarpserver
topics	ports
services	RPC
topic / service remapping	connect
strongly typed topics/services	IDL for RPC, Bottle
catkin (cmake)	cmake
roslaunch	yarpmanager

Command line tools

Your best friends

rostopic list
echo
pub
hz
info

rosservice list
call
info

rosnode list
info

rosparam list
get
set

rosrun pkg_name node_name

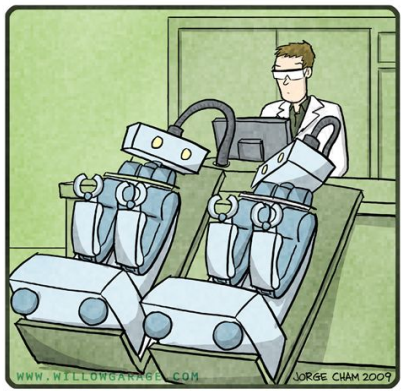
roslaunch pkg_name xyz.launch

rosbag record
play

roswtf

rqt_graph

roscd pkg_name
roscat pkg_name file_name
rosed pkg_name file_name

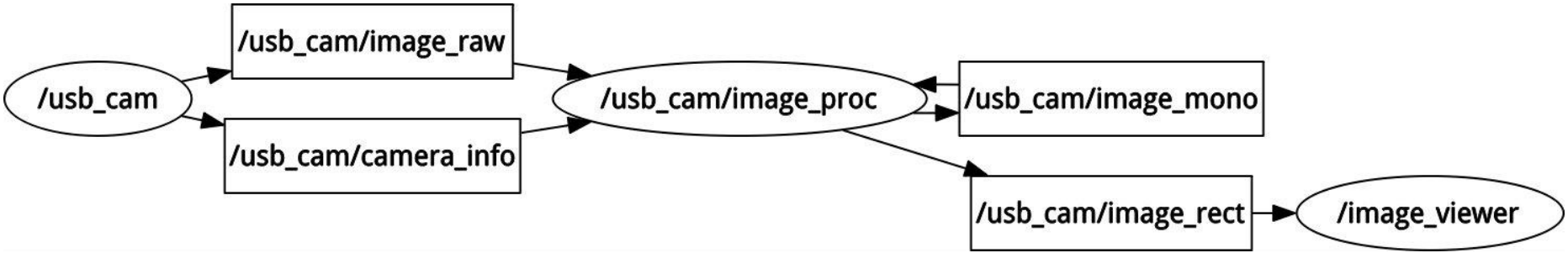


"Do you ever feel like you're in The Matrix?"

ROS

ROS node graph

rqt_graph



ROS

Talker-listener example

In a terminal do:

```
$ cd catkin_ws/src
$ git clone https://github.com/vvv-school/tutorial_ros
$ cd ..
$ catkin_make -j2
$ source devel/setup.bash

$ roscore

$ rosrun tutorial listener

$ rosrun tutorial talker
```

Things to try:

```
$ rostopic list

$ rostopic echo something

$ rqt_graph
```


Talker-listener example

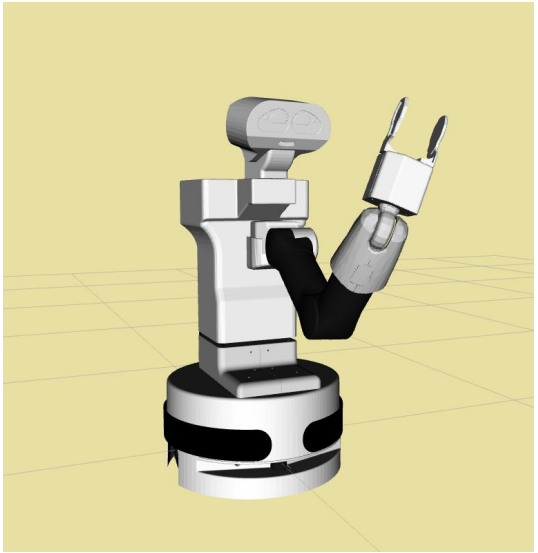
Let’s look at the code!

src/talker.cpp
src/listener.cpp

Robot-specific libraries - URDF

Universal Robot Description Format

- XML-based, supports xacro (xml macros)
- Supports specifying
  - kinematic attributes: length of limbs, size of wheels
  - dynamic attributes: weight and mass distribution of parts
  - visual appearance: fancy 3D meshes for visualisation
  - collision geometry: simplified 3D models used for calculations eg. in simulators or motion planners
- Serves as basis for
  - low level: kinematics, hardware abstraction, controllers
  - high level: planning, navigation, grasping
  - GUIs



ROS

Robot-specific libraries - TF

The transform library

Why?

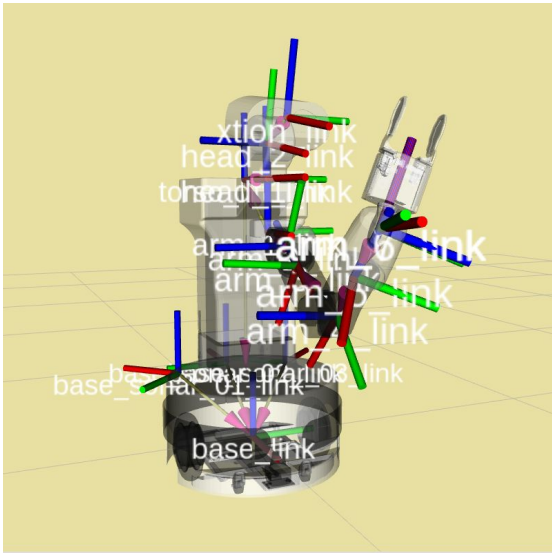
- Robots need to know where things are in the world around them.
- Handy: The URDF already defines a tree-like transformation structure.

Features:

- Lets the user keep track of multiple coordinate frames over time.
- Maintains the relationship between coordinate frames in a tree structure buffered in time.
- Lets the user transform points, vectors, etc between any two coordinate frames at any desired point in time.

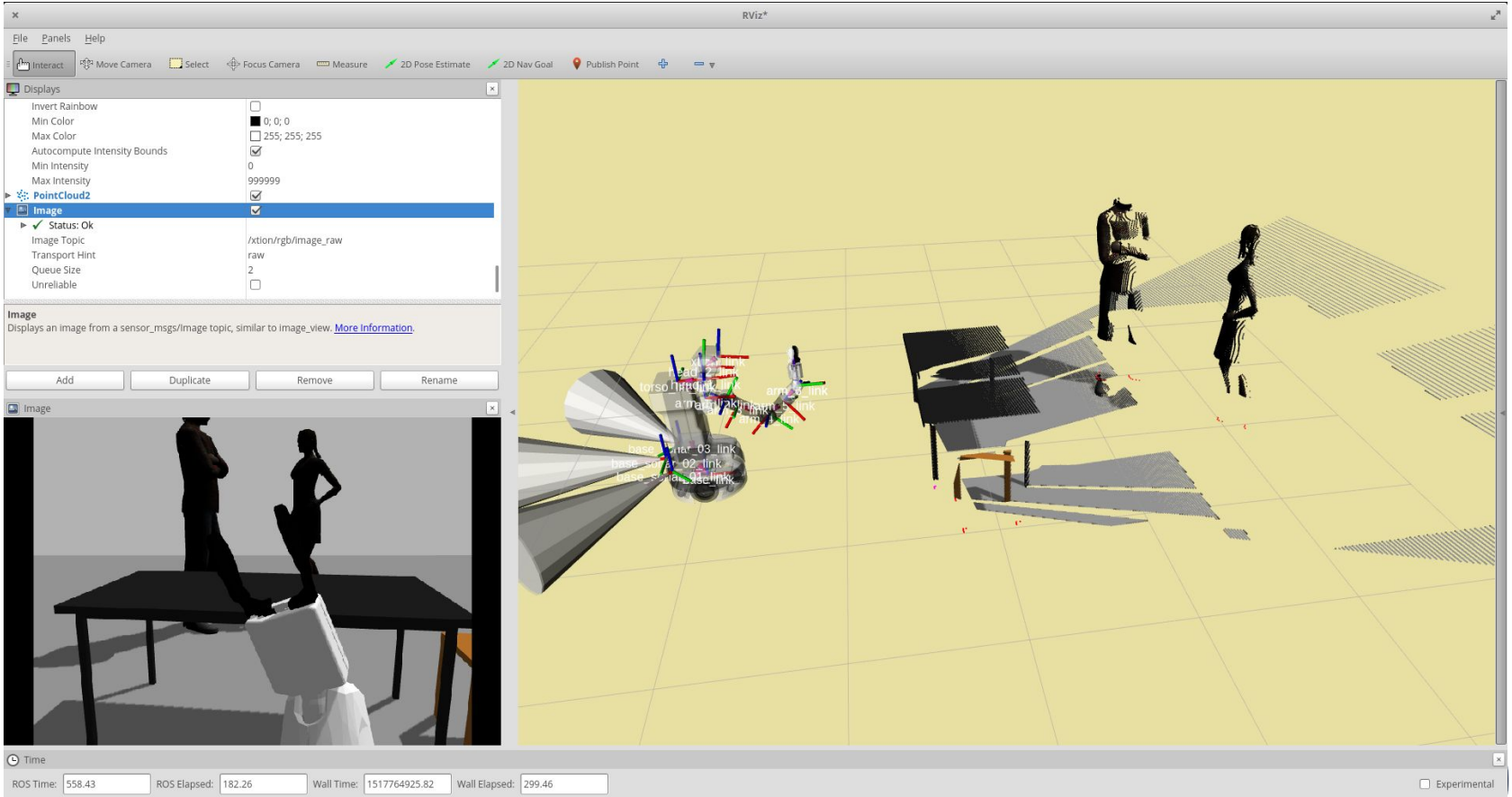
Examples:

- Robot arm moves → joint angles change → transformations are updated. Where's the detected object with respect to the hand?
- SLAM: Where's the robot on the map?



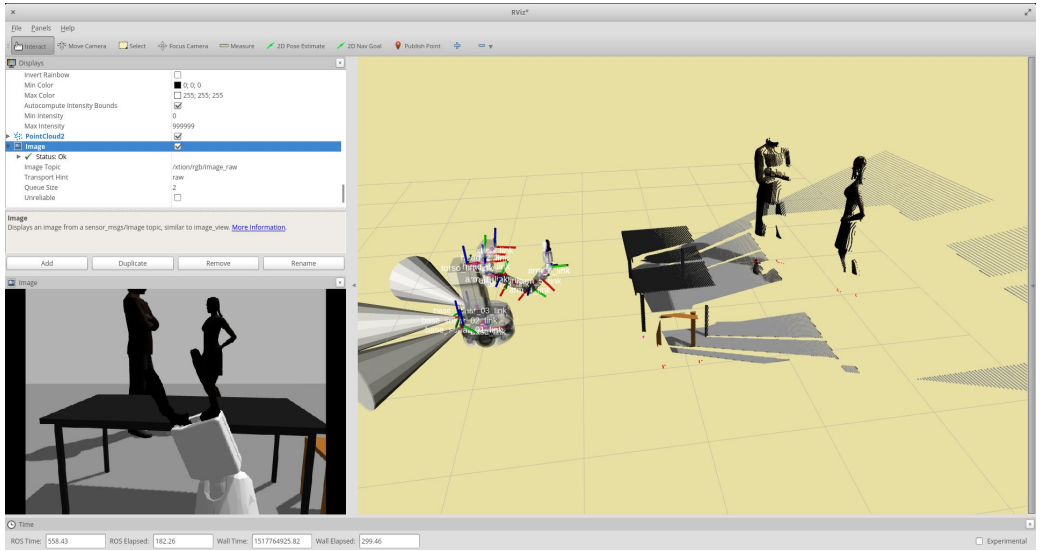
ROS

Visualisation with rviz



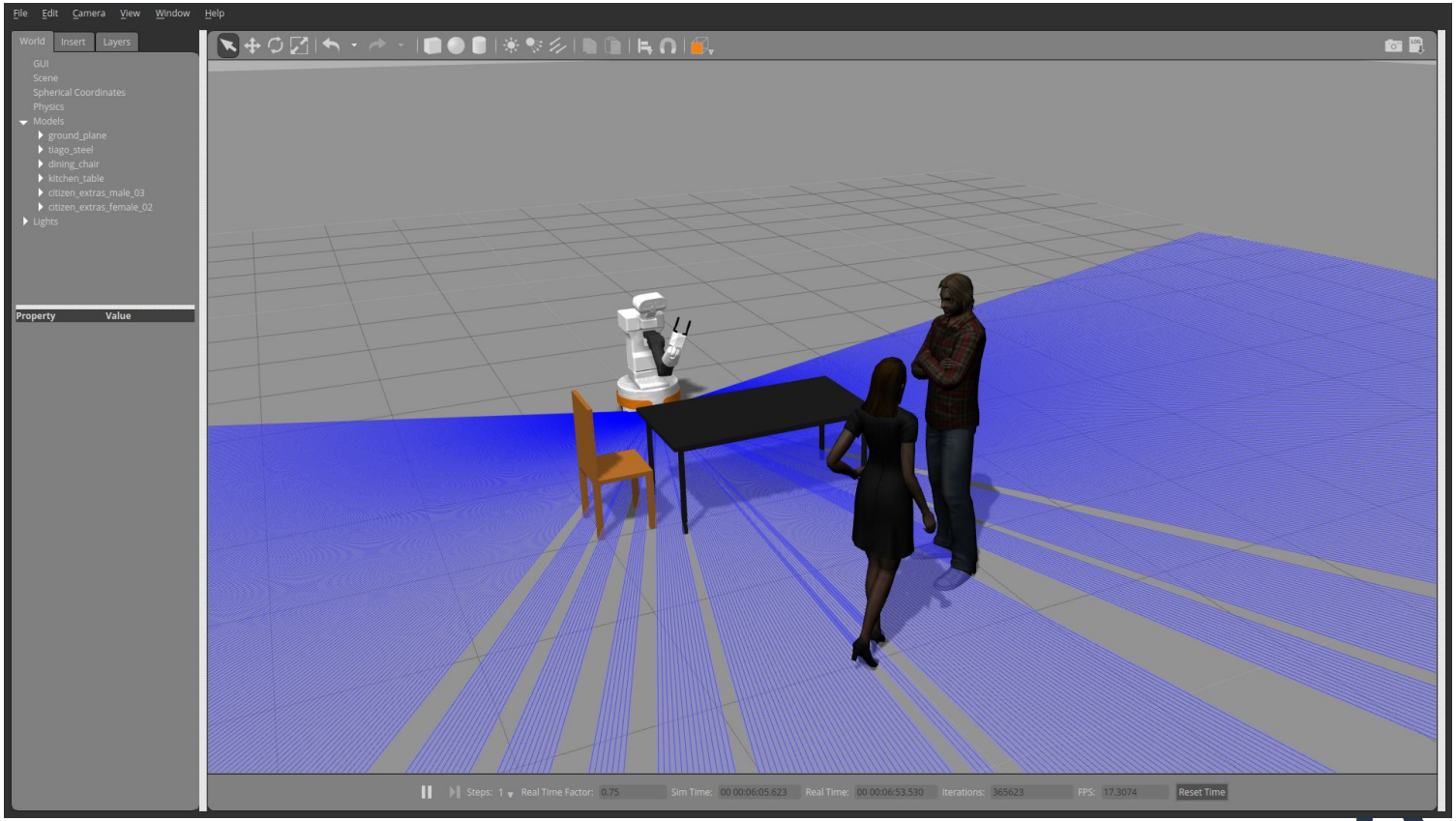
Visualisation with rviz

- RobotModel
- Sensors
- Markers
- Interactive markers
- TF
- Image
- PointCloud
- RobotTrajectory
- Map



ROS

Simulation with Gazebo

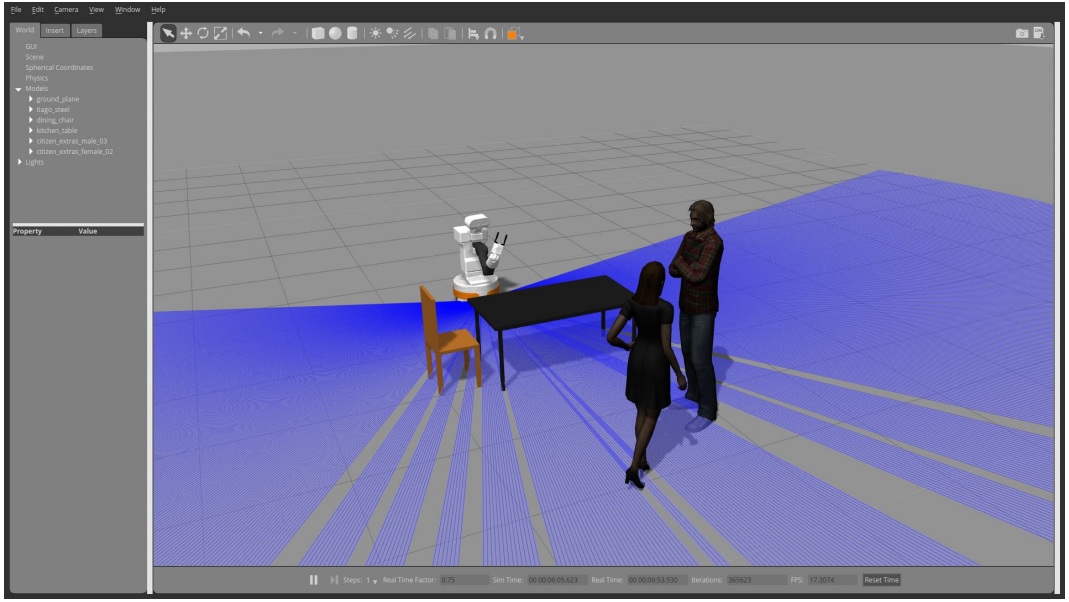


ROS

Simulation with Gazebo

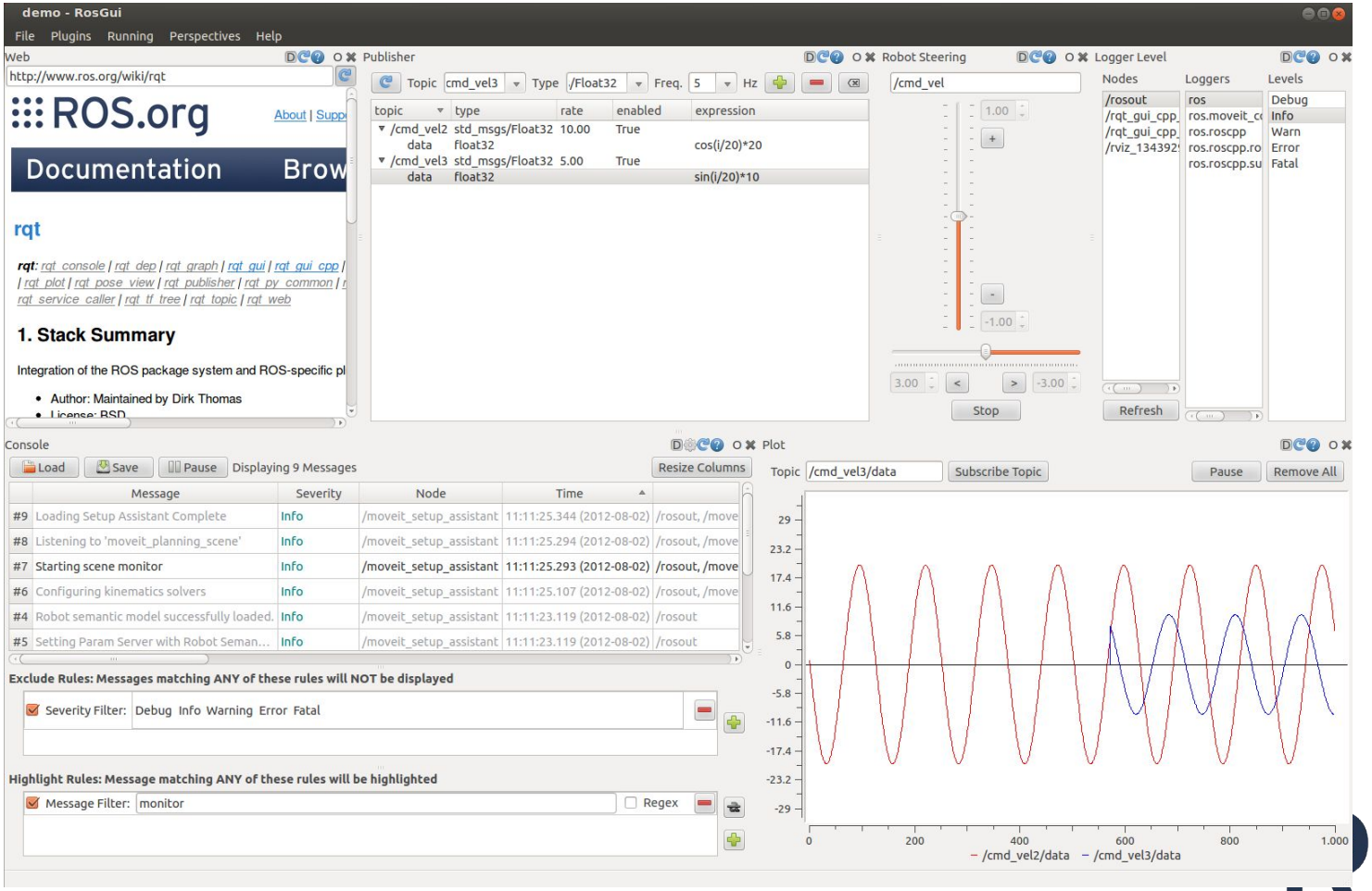
- Sensors
- Actuators
- Physical interactions

- Plugin-based
  - ROS
  - YARP
  - both?!
  - etc
- Online object database
- Interactive or offline world definition
- Darpa Robotics Challenge: many improvements for humanoids



ROS

rqt


demo - RosGui
File Plugins Running Perspectives Help
Web
http://www.ros.org/wiki/rqt
ROS.org
About | Suppo
Documentation
Brow
rqt
rqt: rqt_console | rqt_dep | rqt_graph | rqt_gui | rqt_gui_cpp |
| rqt_plot | rqt_pose_view | rqt_publisher | rqt_py_common |
rqt_service_caller | rqt_tf_tree | rqt_topic | rqt_web
1. Stack Summary
Integration of the ROS package system and ROS-specific pl
Author: Maintained by Dirk Thomas
License: BSD
Publisher
Topic cmd_vel3 Type /Float32 Freq. 5 Hz
topic type rate enabled expression
/cmd_vel2 std_msgs/Float32 10.00 True
data float32 cos(i/20)*20
/cmd_vel3 std_msgs/Float32 5.00 True
data float32 sin(i/20)*10
Robot Steering
/cmd_vel
1.00
-1.00
3.00
-3.00
Stop
Logger Level
Nodes
/rosout
/rqt_gui_cpp_
/rqt_gui_cpp_
/rviz_134392
Loggers
ros
ros.moveit_c
ros.roscpp
ros.roscpp.ro
ros.roscpp.su
Levels
Debug
Info
Warn
Error
Fatal
Refresh
Console
Load Save Pause Displaying 9 Messages
Resize Columns
Message Severity Node Time
#9 Loading Setup Assistant Complete Info /moveit_setup_assistant 11:11:25.344 (2012-08-02) /rosout, /move
#8 Listening to 'moveit_planning_scene' Info /moveit_setup_assistant 11:11:25.294 (2012-08-02) /rosout, /move
#7 Starting scene monitor Info /moveit_setup_assistant 11:11:25.293 (2012-08-02) /rosout, /move
#6 Configuring kinematics solvers Info /moveit_setup_assistant 11:11:25.107 (2012-08-02) /rosout, /move
#4 Robot semantic model successfully loaded. Info /moveit_setup_assistant 11:11:23.119 (2012-08-02) /rosout
#5 Setting Param Server with Robot Seman... Info /moveit_setup_assistant 11:11:23.119 (2012-08-02) /rosout
Exclude Rules: Messages matching ANY of these rules will NOT be displayed
Severity Filter: Debug Info Warning Error Fatal
Highlight Rules: Message matching ANY of these rules will be highlighted
Message Filter: monitor
Regex
Plot
Topic /cmd_vel3/data
Subscribe Topic
Pause
Remove All
29
23.2
17.4
11.6
5.8
0
-5.8
-11.6
-17.4
-23.2
-29
0
200
400
600
800
1.000
/cmd_vel2/data
/cmd_vel3/data

ROS

rosbag & rqt_bag

wget https://www.dropbox.com/s/j3th2xgy48wrg92/2018-01-23-09-13-51.bag
73MB

wget https://www.dropbox.com/s/w283mobbekfdk2t/2018-01-23-09-04-19.bag
170MB

$ roscore

$ rosbag play --loop 2018-01-23-09-13-51.bag

$ rostopic list // $ rostopic echo XYZ // $ rqt_graph // $ rosservice list

$ rqt_bag 2018-01-23-09-13-51.bag # image, plot

Parameters

- The parameter server is a shared, multi-variate dictionary.
- Not designed for high-performance → best used for static, non-binary data eg. configuration parameters.
- Meant to be globally viewable, makes it easy to inspect the configuration state of the system and modify if necessary.

$ rosparam list

$ rosparam get

$ rosparam set

ROS

Launch files

- XML-based format
- To start multiple nodes
- Handle parameters
- Remapping
- With extras:
  - respawn
  - output
  - remotely start

```
$ roslaunch package_name file.launch
```


Parameters and lauch files

Let’s see some code!

Files:

src/param_reader.cpp
src/param_fetcher.launch

Remapping and lauch files

- Create a “chatter.launch” file that starts up talker and listener
    - Hint: output=”screen”

- Remap the “chatter” topic published by the talker to “blabber”
    - Hint: <remap from=”....” to=”...”/>
    - $ roslaunch tutorial chatter.launch
    - rqt_graph, rostopic echo /chatter, rostopic echo /blabber

- Remap the “chatter” topic the listener subscribes to to “blabber”

Assignment: Modify talker node

Take the text to say from the parameter “to_say”

- Make a new launch file starting the talker node and pass the param in
- Read the param in talker.cpp and use it to publish

Services

- Equivalent to Yarp RPC
- Interface defined by a **pair** of messages in a single file: a request and a response
- ros::Service::wait_for_service(ros::Duration timeout);
- command line:
  - $ rosservice list
  - $ rosservice info
  - $ rosservice call

AddTwoInts.srv

```
int64 A
int64 B
---
int64 Sum
```


ROS

Services

Let’s have a look at the code!

src/service_server.cpp
src/service_client.cpp

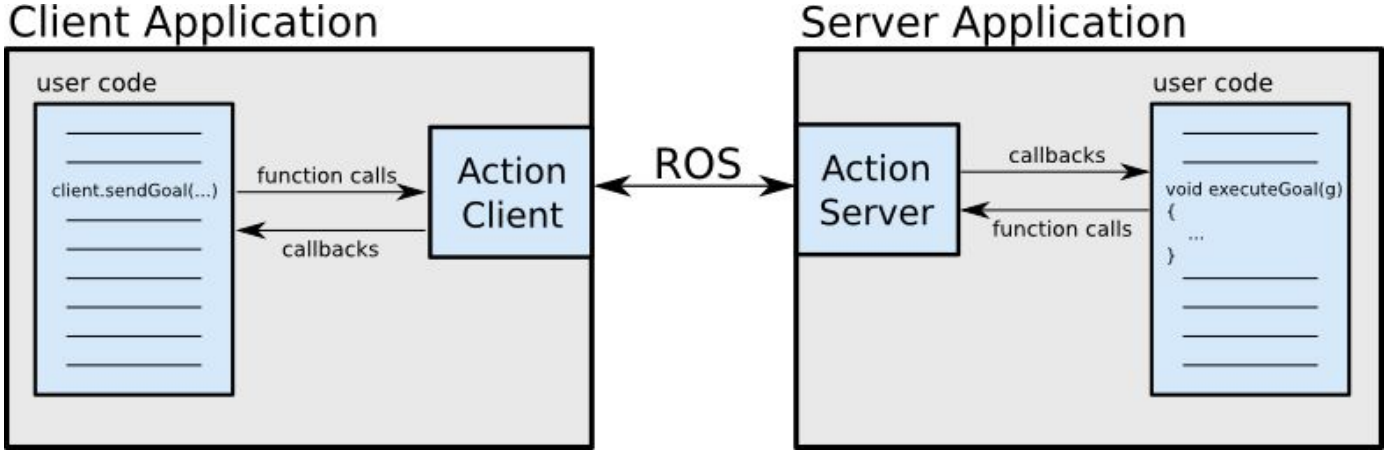
Assignment: Services

- Define a new service type “ToSay.srv”
- Add service client to talker node that allows to change “to_say”
  .
- Service should be successful when “to_say” is less than 10 characters long, reject longer ones.
- Hint: std::string::size()

Actions

- A standardized interface for interfacing with preemptable tasks
- Can be used as asynchronous RPC
- When task execution
  - takes a longer time
  - we wish to have periodic feedback
  - we wish to allow cancelling
- Example: navigation goals, behaviour-based systems, controllers

Actions



Fibonacci.action

```
#goal definition
int32 order
---
#result definition
int32[] sequence
---
#feedback
int32[] sequence
```


ROS

Actions

Let’s look at the code

src/fibonacci_server.cpp
src/fibonacci_client.cpp

Start up rviz with Tiago!

$ roslaunch tiago_description show.launch robot:=steel

$ rostopic list

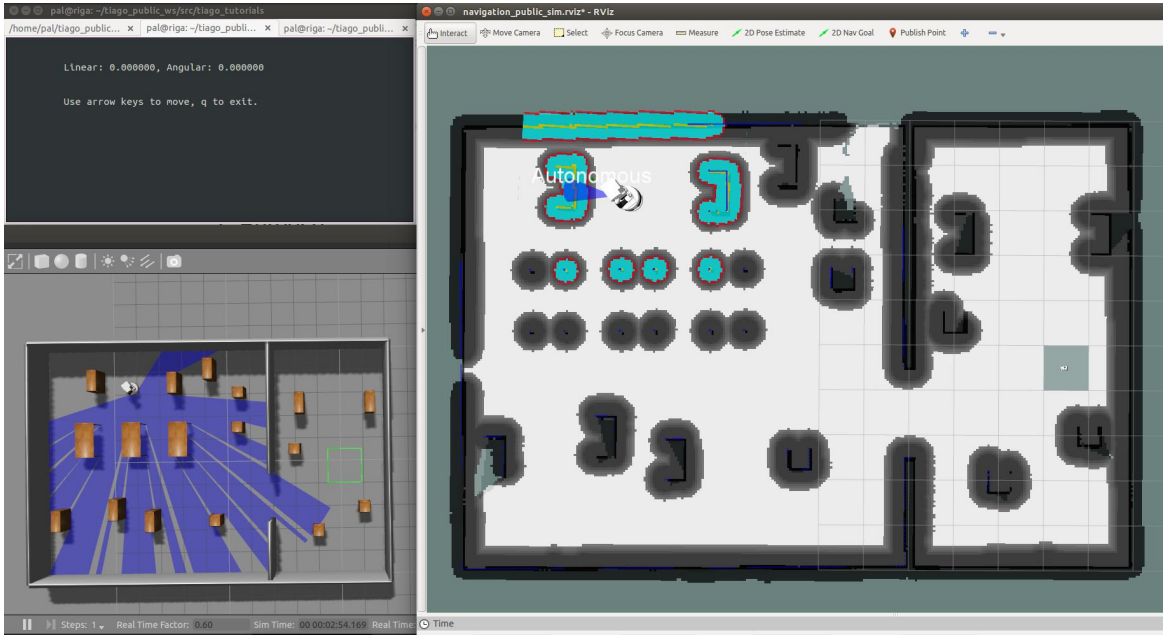
$ rostopic echo /joint_states

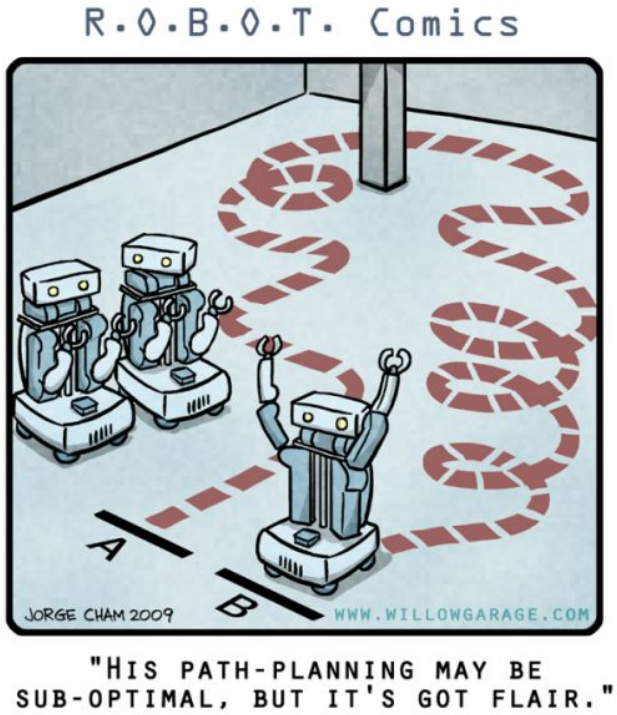
$ rqt_graph

$ rostopic echo /tf

$ rosrun tf view_frames && evince frames.pdf

Navigation



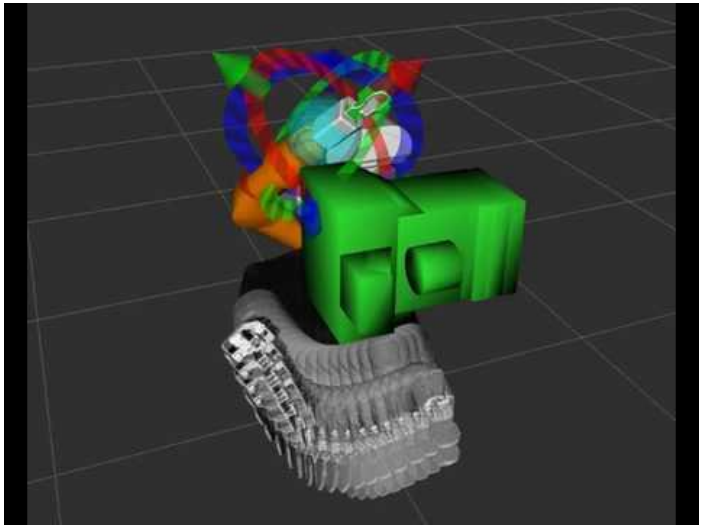


If interested: http://wiki.ros.org/Robots/TIAGo/Tutorials/Navigation/Mapping

ROS

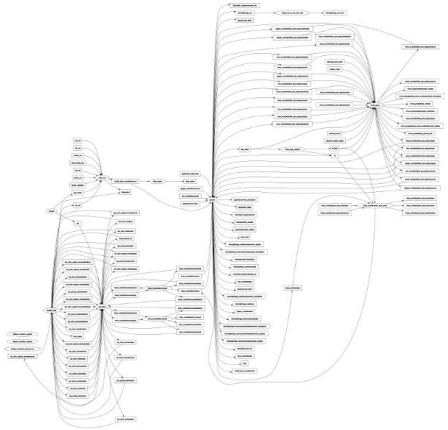
MoveIt demo with Tiago

$ roslaunch tiago_moveit_config demo.launch robot:=steel



ROS

Start up Gazebo with Tiago!



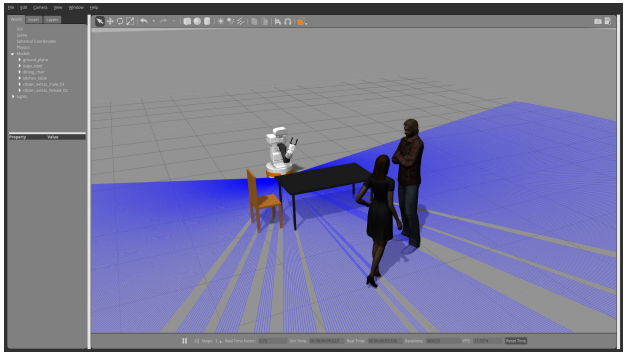
$ roslaunch tiago_gazebo tiago_gazebo.launch robot:=steel

optional: gui:=False

$ rqt_graph

$ rostopic echo /joint_states

$ rosrun rqt_joint_trajectory_controller rqt_joint_trajectory_controller



ROS

Assignment: Move Tiago's arm

… using the arm_controller through the FollowJointTrajectory action interface.

File: tiago_bend_elbow.cpp

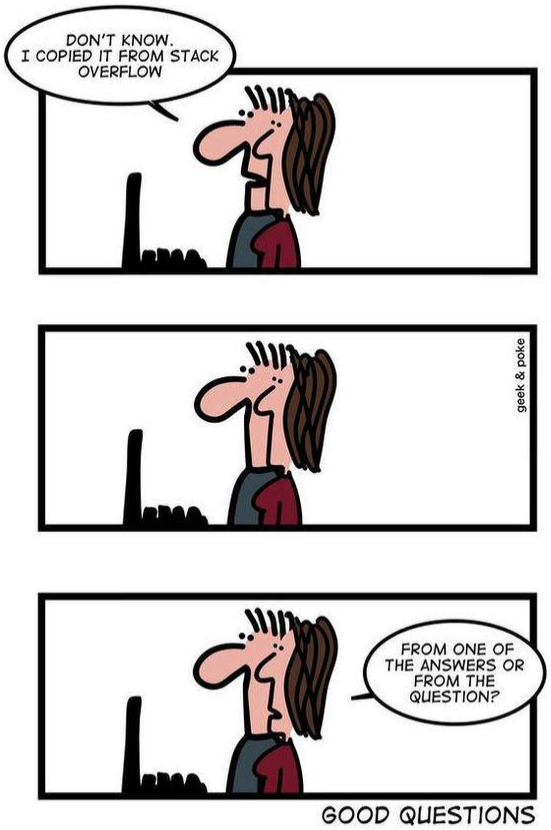
For general info:
http://wiki.ros.org/joint_trajectory_controller

$ roslaunch tiago_gazebo tiago_gazebo.launch robot:=steel #optional: gui:=False

$ rosrun tutorial tiago_bend_elbow

ROS

Questions!



ROS