



# Programming for Robotics

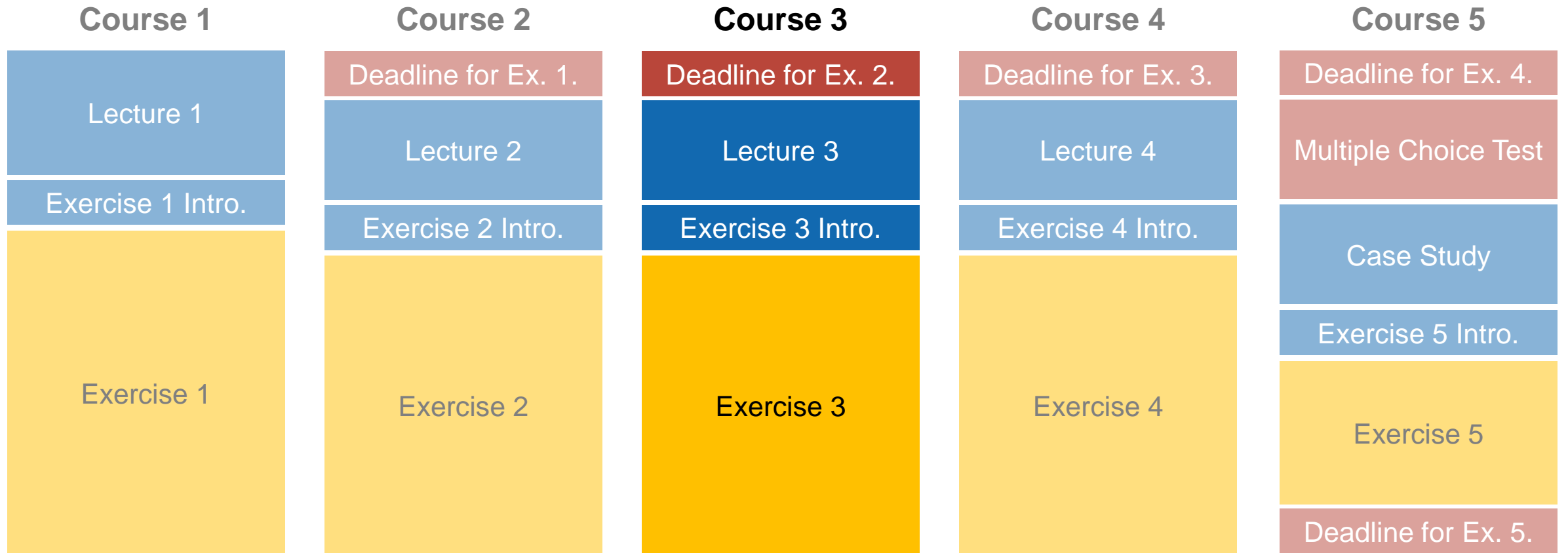
## Introduction to ROS

Course 3

Marko Bjelonic, Dominic Jud, Martin Wermelinger, Péter Fankhauser  
Prof. Dr. Marco Hutter



# Course Structure

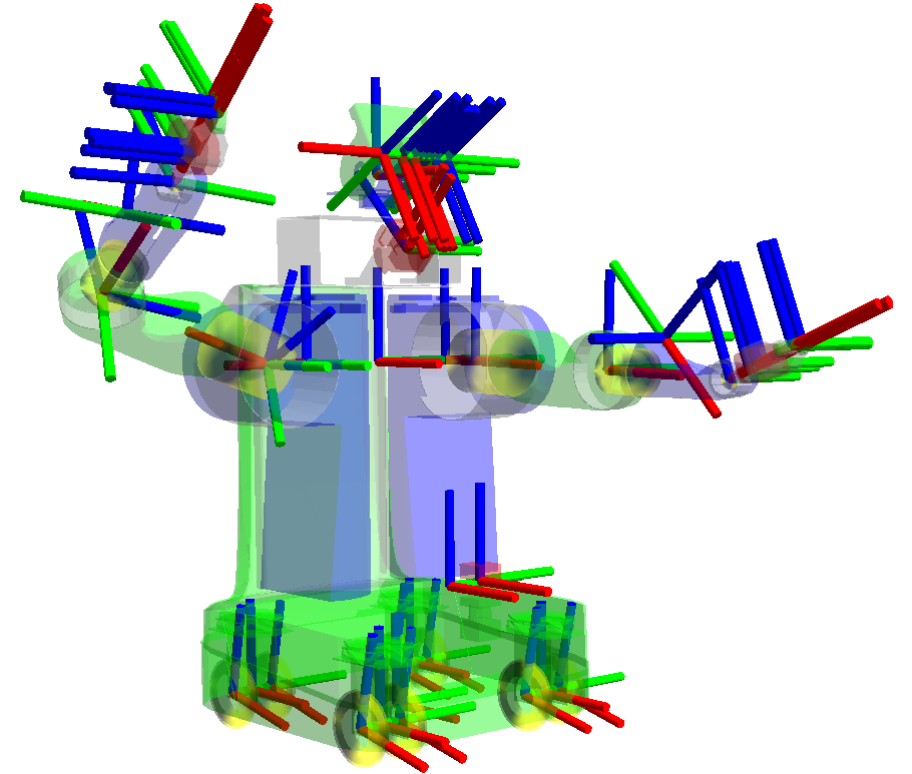
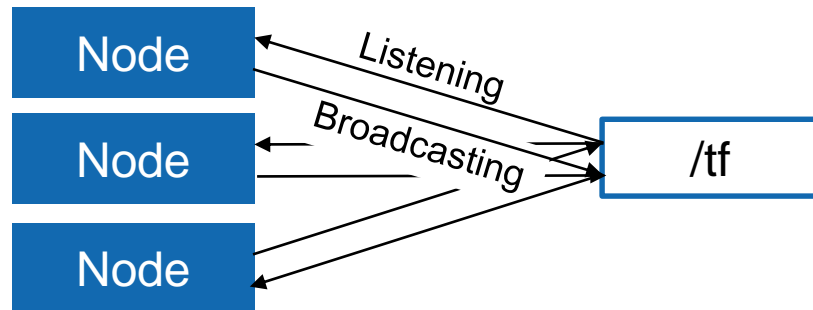


## Overview Course 3

- TF Transformation System
- rqt User Interface
- Robot models (URDF)
- Simulation descriptions (SDF)

# TF Transformation System

- Tool for keeping track of coordinate frames over time
- Maintains relationship between coordinate frames in a tree structure buffered in time
- Lets the user transform points, vectors, etc. between coordinate frames at desired time
- Implemented as publisher/subscriber model on the topics `/tf` and `/tf_static`



**More info**  
<http://wiki.ros.org/tf2>

# TF Transformation System

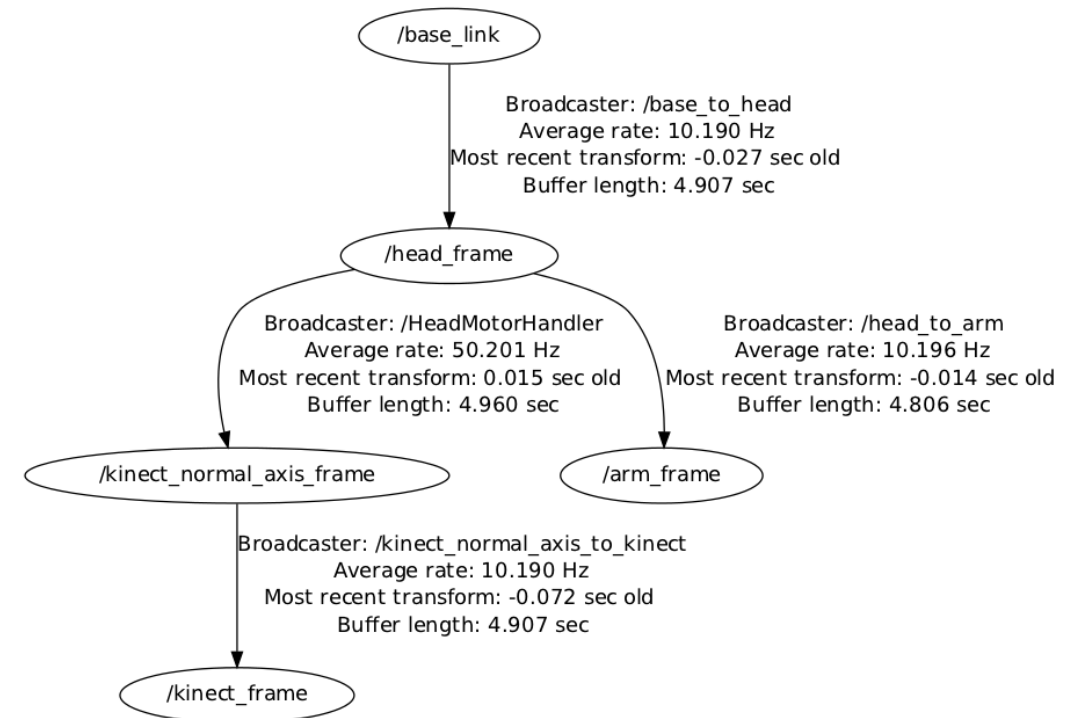
## Transform Tree

- TF listeners use a buffer to listen to all broadcasted transforms
- Query for specific transforms from the transform tree

### tf2\_msgs/TFMessage.msg

```

geometry_msgs/TransformStamped[] transforms
std_msgs/Header header
uint32 seqtime stamp
string frame_id
string child_frame_id
geometry_msgs/Transform transform
geometry_msgs/Vector3 translation
geometry_msgs/Quaternion rotation
  
```



# TF Transformation System Tools

## Command line

Print information about the current transform tree

```
> rosrun tf tf_monitor
```

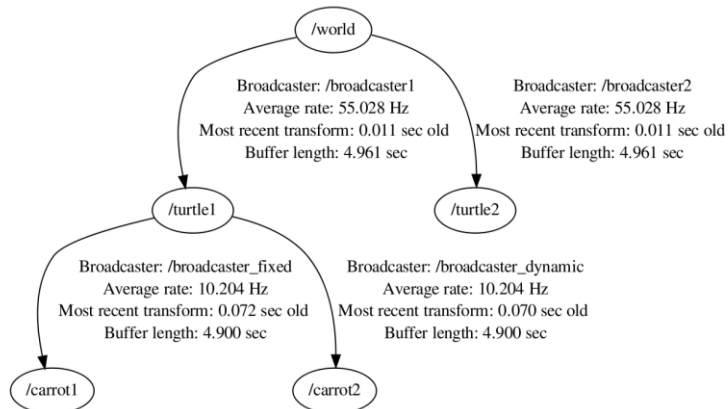
Print information about the transform between two frames

```
> rosrun tf tf_echo
  source_frame target_frame
```

## View Frames

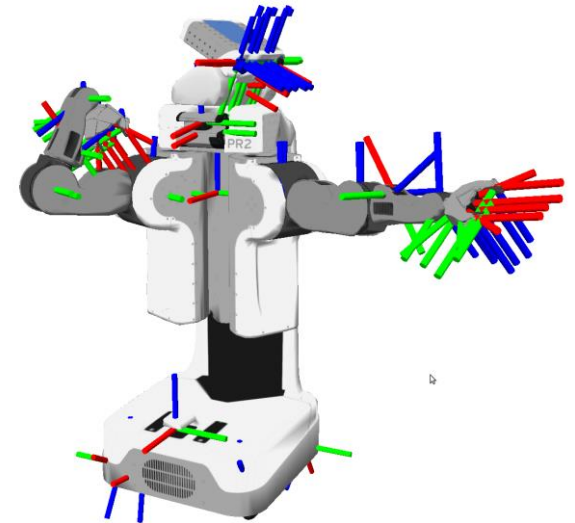
Creates a visual graph (PDF) of the transform tree

```
> rosrun tf view_frames
```

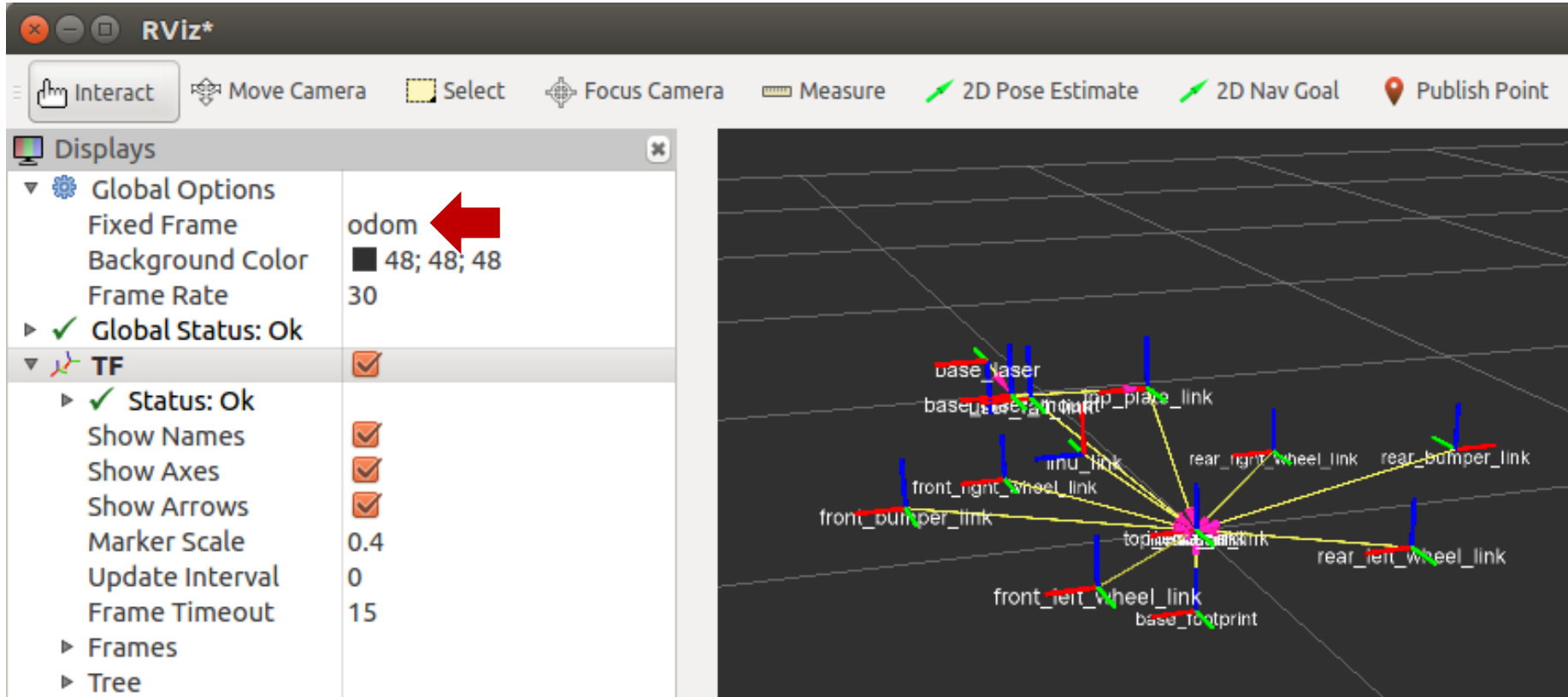


## RViz

3D visualization of the transforms



# TF Transformation System RViz Plugin



# TF Transformation System

## Transform Listener C++ API

- Create a TF listener to fill up a buffer

```
tf2_ros::Buffer tfBuffer;
tf2_ros::TransformListener tfListener(tfBuffer);
```

- Make sure, that the listener does not run out of scope!

- To lookup transformations, use

```
geometry_msgs::TransformStamped transformStamped =
tfBuffer.lookupTransform(target_frame_id,
                        source_frame_id, time);
```

- For time, use `ros::Time(0)` to get the latest available transform

```
#include <ros/ros.h>
#include <tf2_ros/transform_listener.h>
#include <geometry_msgs/TransformStamped.h>

int main(int argc, char** argv) {
  ros::init(argc, argv, "tf2_listener");
  ros::NodeHandle nodeHandle;
  tf2_ros::Buffer tfBuffer;
  tf2_ros::TransformListener tfListener(tfBuffer);

  ros::Rate rate(10.0);
  while (nodeHandle.ok()) {
    geometry_msgs::TransformStamped transformStamped;
    try {
      transformStamped = tfBuffer.lookupTransform("base",
                                                "odom", ros::Time(0));
    } catch (tf2::TransformException &exception) {
      ROS_WARN("%s", exception.what());
      ros::Duration(1.0).sleep();
      continue;
    }
    rate.sleep();
  }
  return 0;
};
```

**More info**

<http://wiki.ros.org/tf2/Tutorials/Writing%20a%20tf2%20listener%20%28C%2B%2B%29>



# rqt User Interface

- User interface based on Qt
- Custom interfaces can be setup
- Lots of plugins exist
- Simple to write own plugins

Run RQT with

```
> rosrun rqt_gui rqt_gui
```

or

```
> rqt
```

The screenshot displays the ROS GUI (rqt) interface. The top window shows the ROS.org website with the 'rqt' section highlighted. Below the website, there is a '1. Stack Summary' section. The middle window shows a topic monitor with columns for topic, type, rate, enabled, and expression. Two topics are listed: `/cmd_vel2` (std\_msgs/Float32, rate 10.00) and `/cmd_vel3` (std\_msgs/Float32, rate 5.00). The bottom window shows a plot of `/cmd_vel3/data` with two sine waves (red and blue) over time. The console window at the bottom shows a list of messages with columns for message, severity, node, and time.

Message	Severity	Node	Time
#9 Loading Setup Assistant Complete	Info	/moveit_setup_assistant	11:11:25.344 (2012-08-02)
#8 Listening to 'moveit_planning_scene'	Info	/moveit_setup_assistant	11:11:25.294 (2012-08-02)
#7 Starting scene monitor	Info	/moveit_setup_assistant	11:11:25.293 (2012-08-02)
#6 Configuring kinematics solvers	Info	/moveit_setup_assistant	11:11:25.107 (2012-08-02)
#4 Robot semantic model successfully loaded.	Info	/rosout	11:11:23.119 (2012-08-02)
#5 Setting Param Server with Robot Seman...	Info	/moveit_setup_assistant	11:11:23.119 (2012-08-02)

More info

<http://wiki.ros.org/rqt/Plugins>

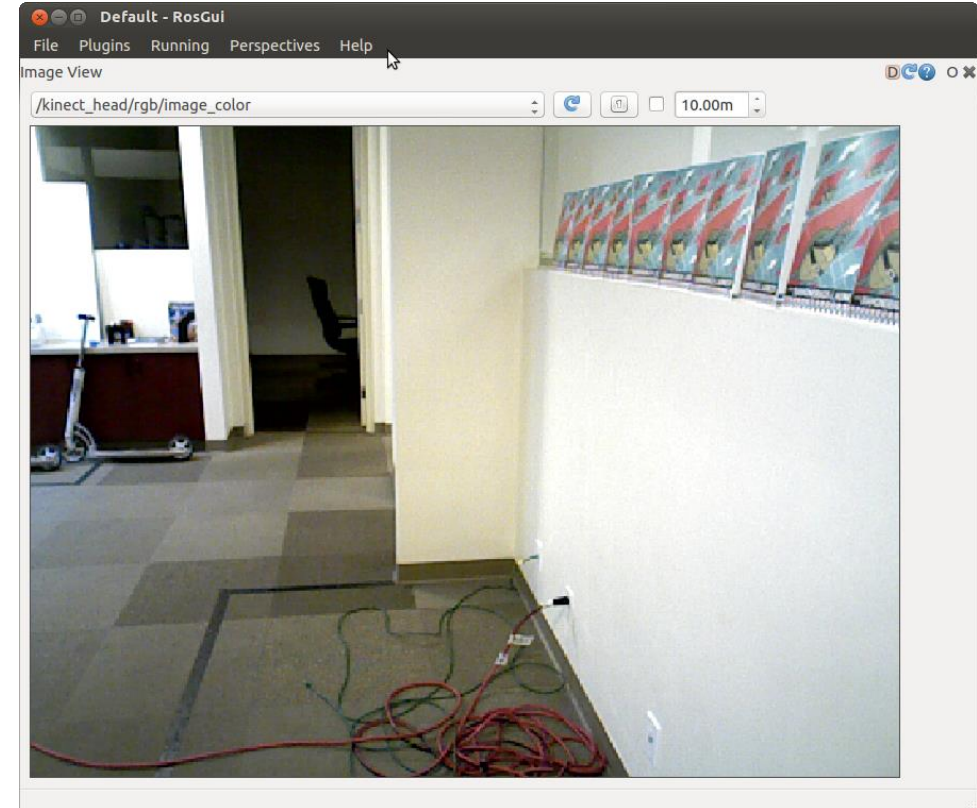
# rqt User Interface

## rqt\_image\_view

- Visualizing images

Run *rqt\_graph* with

```
> rosrun rqt_image_view rqt_image_view
```



**More info**

[http://wiki.ros.org/rqt\\_image\\_view](http://wiki.ros.org/rqt_image_view)

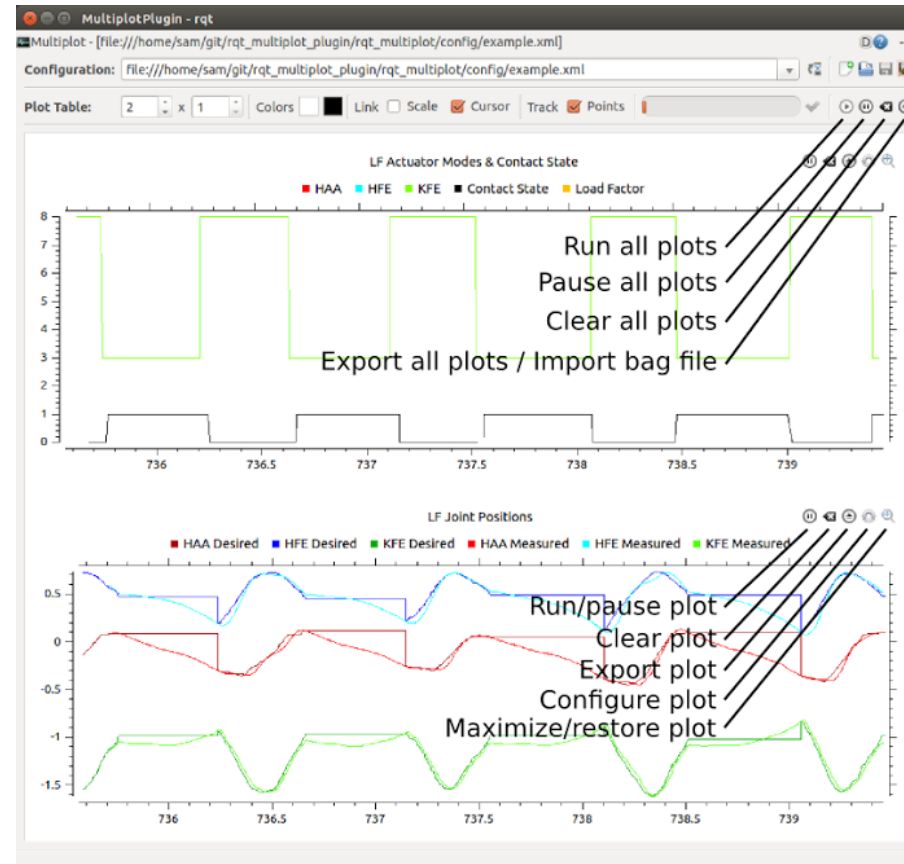
# rqt User Interface

## rqt\_multiplot

- Visualizing numeric values in 2D plots

Run *rqt\_multiplot* with

```
> rosrun rqt_multiplot rqt_multiplot
```



More info

[http://wiki.ros.org/rqt\\_multiplot](http://wiki.ros.org/rqt_multiplot)

# rqt User Interface

## rqt\_graph

- Visualizing the ROS computation graph

Run *rqt\_graph* with

```
> rosrun rqt_graph rqt_graph
```



More info

[http://wiki.ros.org/rqt\\_graph](http://wiki.ros.org/rqt_graph)

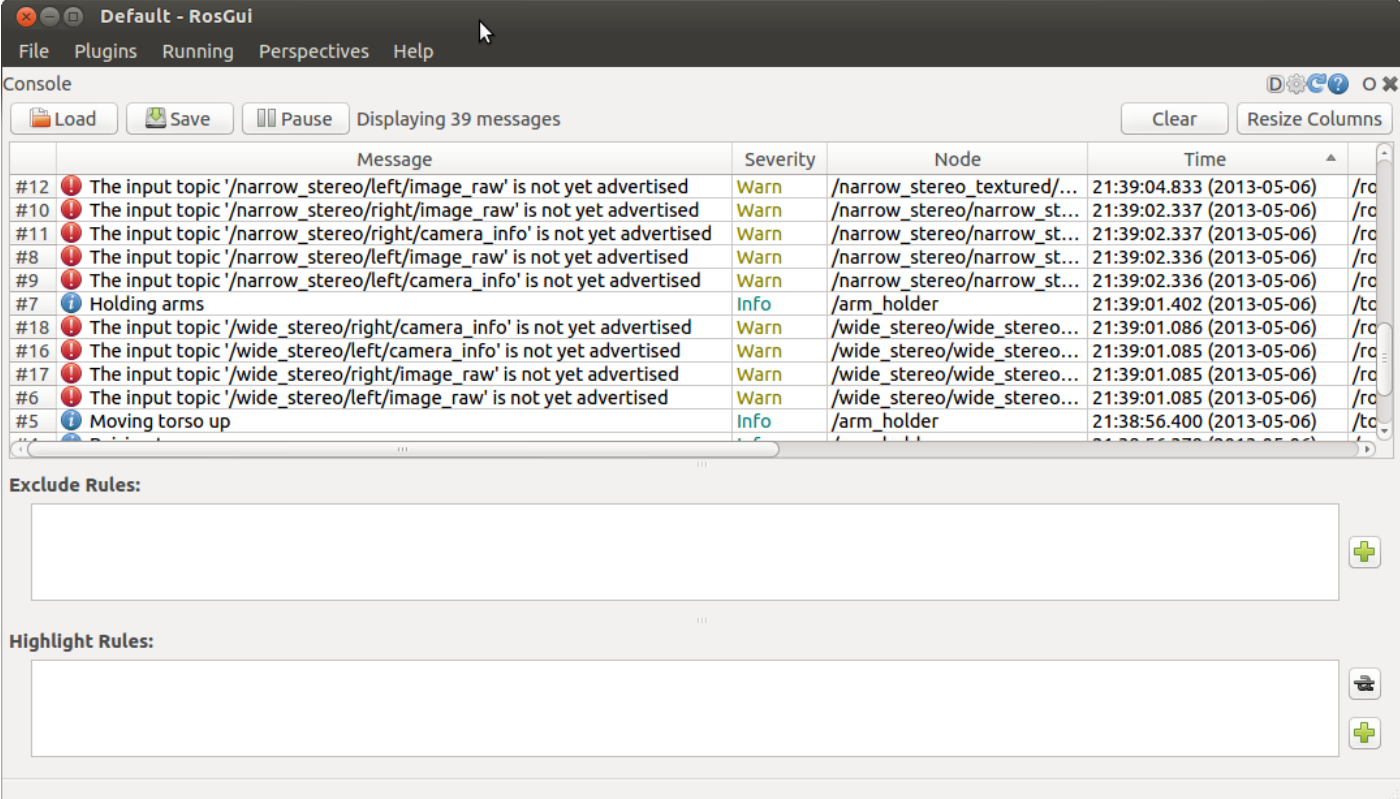
# rqt User Interface

## rqt\_console

- Displaying and filtering ROS messages

Run *rqt\_console* with

```
> rosrn rqt_console rqt_console
```



The screenshot shows the rqt\_console window with a menu bar (File, Plugins, Running, Perspectives, Help) and a toolbar (Load, Save, Pause, Clear, Resize Columns). The main area displays a table of 39 messages. Below the table are sections for 'Exclude Rules' and 'Highlight Rules'.

	Message	Severity	Node	Time
#12	The input topic '/narrow_stereo/left/image_raw' is not yet advertised	Warn	/narrow_stereo_textured/...	21:39:04.833 (2013-05-06)
#10	The input topic '/narrow_stereo/right/image_raw' is not yet advertised	Warn	/narrow_stereo/narrow_st...	21:39:02.337 (2013-05-06)
#11	The input topic '/narrow_stereo/right/camera_info' is not yet advertised	Warn	/narrow_stereo/narrow_st...	21:39:02.337 (2013-05-06)
#8	The input topic '/narrow_stereo/left/image_raw' is not yet advertised	Warn	/narrow_stereo/narrow_st...	21:39:02.336 (2013-05-06)
#9	The input topic '/narrow_stereo/left/camera_info' is not yet advertised	Warn	/narrow_stereo/narrow_st...	21:39:02.336 (2013-05-06)
#7	Holding arms	Info	/arm_holder	21:39:01.402 (2013-05-06)
#18	The input topic '/wide_stereo/right/camera_info' is not yet advertised	Warn	/wide_stereo/wide_stereo...	21:39:01.086 (2013-05-06)
#16	The input topic '/wide_stereo/left/camera_info' is not yet advertised	Warn	/wide_stereo/wide_stereo...	21:39:01.085 (2013-05-06)
#17	The input topic '/wide_stereo/right/image_raw' is not yet advertised	Warn	/wide_stereo/wide_stereo...	21:39:01.085 (2013-05-06)
#6	The input topic '/wide_stereo/left/image_raw' is not yet advertised	Warn	/wide_stereo/wide_stereo...	21:39:01.085 (2013-05-06)
#5	Moving torso up	Info	/arm_holder	21:38:56.400 (2013-05-06)

More info

[http://wiki.ros.org/rqt\\_console](http://wiki.ros.org/rqt_console)

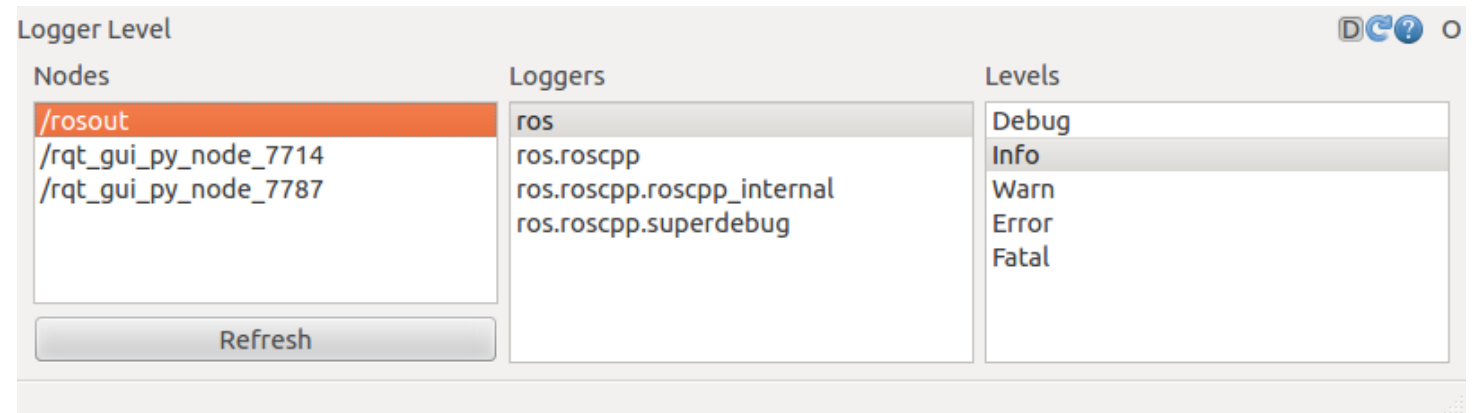
# rqt User Interface

## rqt\_logger\_level

- Configuring the logger level of ROS nodes

Run *rqt\_logger\_level* with

```
> rosrun rqt_logger_level  
rqt_logger_level
```



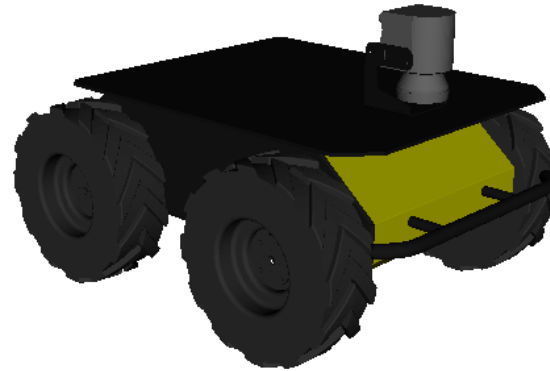
**More info**

[http://wiki.ros.org/rqt\\_logger\\_level](http://wiki.ros.org/rqt_logger_level)

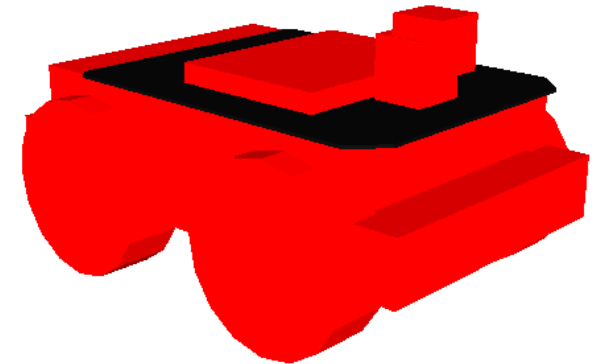
# Robot Models

## Unified Robot Description Format (URDF)

- Defines an XML format for representing a robot model
  - Kinematic and dynamic description
  - Visual representation
  - Collision model
- URDF generation can be scripted with *XACRO*



Mesh for visuals



Primitives for collision

### More info

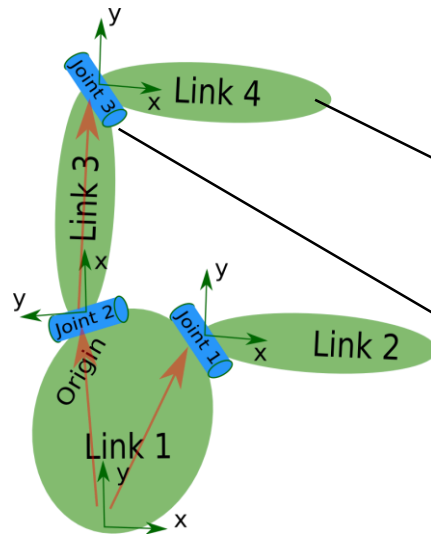
<http://wiki.ros.org/urdf>

<http://wiki.ros.org/xacro>

# Robot Models

## Unified Robot Description Format (URDF)

- Description consists of a set of *link* elements and a set of *joint* elements
- Joints connect the links together



*robot.urdf*

```
<robot name="robot">
  <link> ... </link>
  <link> ... </link>
  <link> ... </link>

  <joint> .... </joint>
  <joint> .... </joint>
  <joint> .... </joint>
</robot>
```

```
<link name="Link_name">
  <visual>
    <geometry>
      <mesh filename="mesh.dae"/>
    </geometry>
  </visual>
  <collision>
    <geometry>
      <cylinder length="0.6" radius="0.2"/>
    </geometry>
  </collision>
  <inertial>
    <mass value="10"/>
    <inertia ixx="0.4" ixy="0.0" .../>
  </inertial>
</link>
```

```
<joint name="joint_name" type="revolute">
  <axis xyz="0 0 1"/>
  <limit effort="1000.0" upper="0.548" ... />
  <origin rpy="0 0 0" xyz="0.2 0.01 0"/>
  <parent link="parent_link_name"/>
  <child link="child_link_name"/>
</joint>
```


### More info

<http://wiki.ros.org/urdf/XML/model>



# Robot Models

## Usage in ROS

- The robot description (URDF) is stored on the parameter server (typically) under `/robot_description`
- You can visualize the robot model in Rviz with the  *RobotModel* plugin

### *spawn\_husky.launch*

```
...
<include file="$(find
husky_description)/launch/description.launch" >
  <arg name="robot_namespace" value="$(arg robot_namespace)"/>
  <arg name="laser_enabled" default="$(arg laser_enabled)"/>
  <arg name="kinect_enabled" default="$(arg kinect_enabled)"/>
  <arg name="urdf_extras" default="$(arg urdf_extras)"/>
</include>
...
```

### *description.launch*

```
...
<param name="robot_description" command="$(find xacro)/xacro
'$(find husky_description)/urdf/husky.urdf.xacro'
  --inorder
  robot_namespace:=$(arg robot_namespace)
  laser_enabled:=$(arg laser_enabled)
  kinect_enabled:=$(arg kinect_enabled)
  urdf_extras:=$(arg urdf_extras)" />
...
```

# Simulation Descriptions

## Simulation Description Format (SDF)

- Defines an XML format to describe
  - Environments (lighting, gravity etc.)
  - Objects (static and dynamic)
  - Sensors
  - Robots
- SDF is the standard format for Gazebo
- Gazebo converts a URDF to SDF automatically



### More info

<http://sdformat.org>

## Further References

- **ROS Wiki**
  - <http://wiki.ros.org/>
- **Installation**
  - <http://wiki.ros.org/ROS/Installation>
- **Tutorials**
  - <http://wiki.ros.org/ROS/Tutorials>
- **Available packages**
  - <http://www.ros.org/browse/>
- **ROS Cheat Sheet**
  - <https://www.clearpathrobotics.com/ros-robot-operating-system-cheat-sheet/>
  - [https://kapeli.com/cheat\\_sheets/ROS.docset/Contents/Resources/Documents/index](https://kapeli.com/cheat_sheets/ROS.docset/Contents/Resources/Documents/index)
- **ROS Best Practices**
  - [https://github.com/leggedrobotics/ros\\_best\\_practices/wiki](https://github.com/leggedrobotics/ros_best_practices/wiki)
- **ROS Package Template**
  - [https://github.com/leggedrobotics/ros\\_best\\_practices/tree/master/ros\\_package\\_template](https://github.com/leggedrobotics/ros_best_practices/tree/master/ros_package_template)

# Contact Information

## ETH Zurich

Robotic Systems Lab  
Prof. Dr. Marco Hutter  
LEE H 303  
Leonhardstrasse 21  
8092 Zurich  
Switzerland

<http://www.rsl.ethz.ch>

## Lecturers

Marko Bjelonic (marko.bjelonic@mavt.ethz.ch)  
Dominic Jud (dominic.jud@mavt.ethz.ch)  
Martin Wermelinger (martin.wermelinger@mavt.ethz.ch)  
Péter Fankhauser (pfankhauser@anybotics.com)

Course website: <http://www.rsl.ethz.ch/education-students/lectures/ros.html>