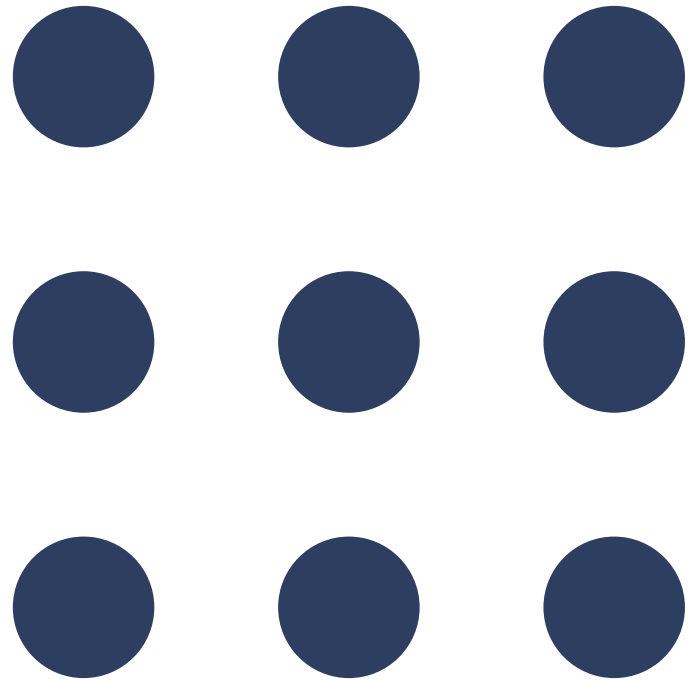
WITH PLYMOUTH UNIVERSITY

The Robot Operating System

High-Altitude Overview of ROS

December 8, 2015



Séverin Lemaignan

Centre for Robotics and Neural Systems
Plymouth University

ROS IS NOT AN OPERATING SYSTEM

INSTEAD, ROS IS...

- A fairly simple peer-to-peer message passing system designed with robotics in mind

INSTEAD, ROS IS...

- A fairly simple peer-to-peer message passing system designed with robotics in mind
- An API to this system (in several languages – C++ and Python are 1st tier)

INSTEAD, ROS IS...

- A fairly simple peer-to-peer message passing system designed with robotics in mind
- An API to this system (in several languages – C++ and Python are 1st tier)
- A set of standard message types that facilitate interoperability between modules

INSTEAD, ROS IS...

- A fairly simple peer-to-peer message passing system designed with robotics in mind
- An API to this system (in several languages – C++ and Python are 1st tier)
- A set of standard message types that facilitate interoperability between modules
- **A middleware?**

INSTEAD, ROS IS...

- A fairly simple peer-to-peer message passing system designed with robotics in mind
- An API to this system (in several languages – C++ and Python are 1st tier)
- A set of standard message types that facilitate interoperability between modules

- A set of conventions to write and package robotic softwares

INSTEAD, ROS IS...

- A fairly simple peer-to-peer message passing system designed with robotics in mind
- An API to this system (in several languages – C++ and Python are 1st tier)
- A set of standard message types that facilitate interoperability between modules

- A set of conventions to write and package robotic softwares
- Deep integration of a few key open-source libraries (OpenCV, PCL, tf)

INSTEAD, ROS IS...

- A fairly simple peer-to-peer message passing system designed with robotics in mind
- An API to this system (in several languages – C++ and Python are 1st tier)
- A set of standard message types that facilitate interoperability between modules

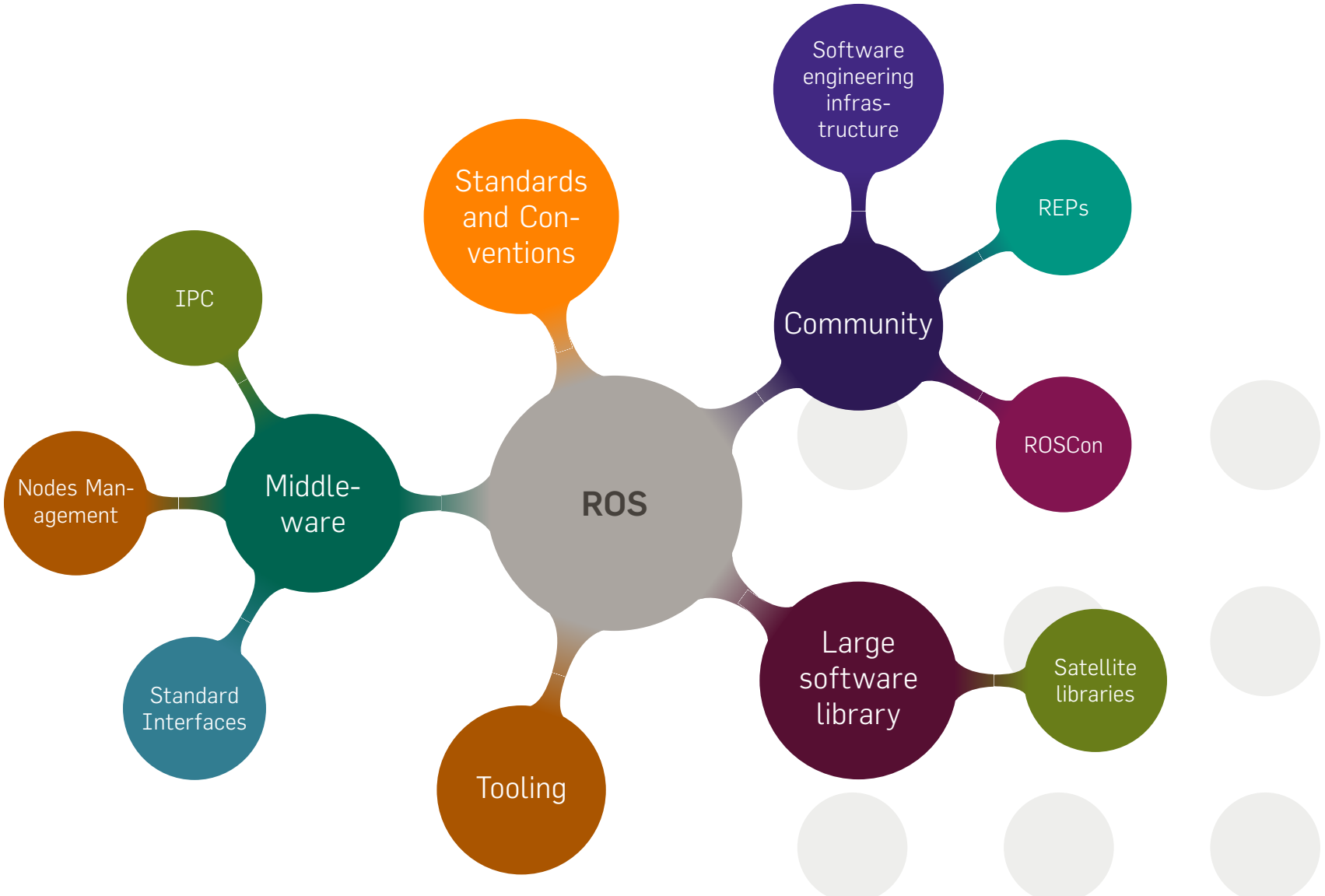
- A set of conventions to write and package robotic softwares
- Deep integration of a few key open-source libraries (OpenCV, PCL, tf)
- A set of tools to run and monitor the nodes

INSTEAD, ROS IS...

- A fairly simple peer-to-peer message passing system designed with robotics in mind
- An API to this system (in several languages – C++ and Python are 1st tier)
- A set of standard message types that facilitate interoperability between modules

- A set of conventions to write and package robotic softwares
- Deep integration of a few key open-source libraries (OpenCV, PCL, tf)
- A set of tools to run and monitor the nodes
- Engagement of a large academic community, leading to a library of thousands of nodes

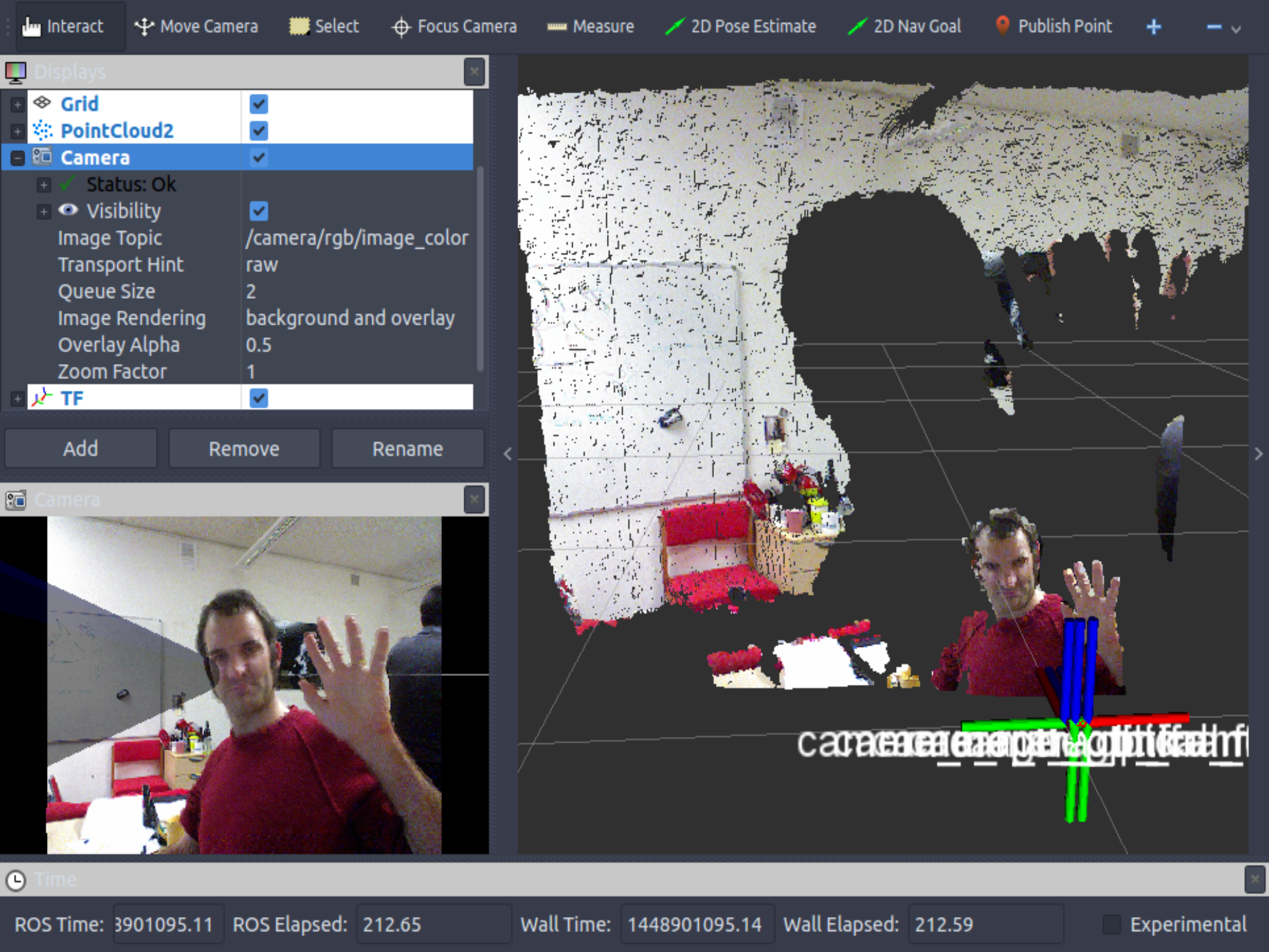
ROS ECOSYSTEM



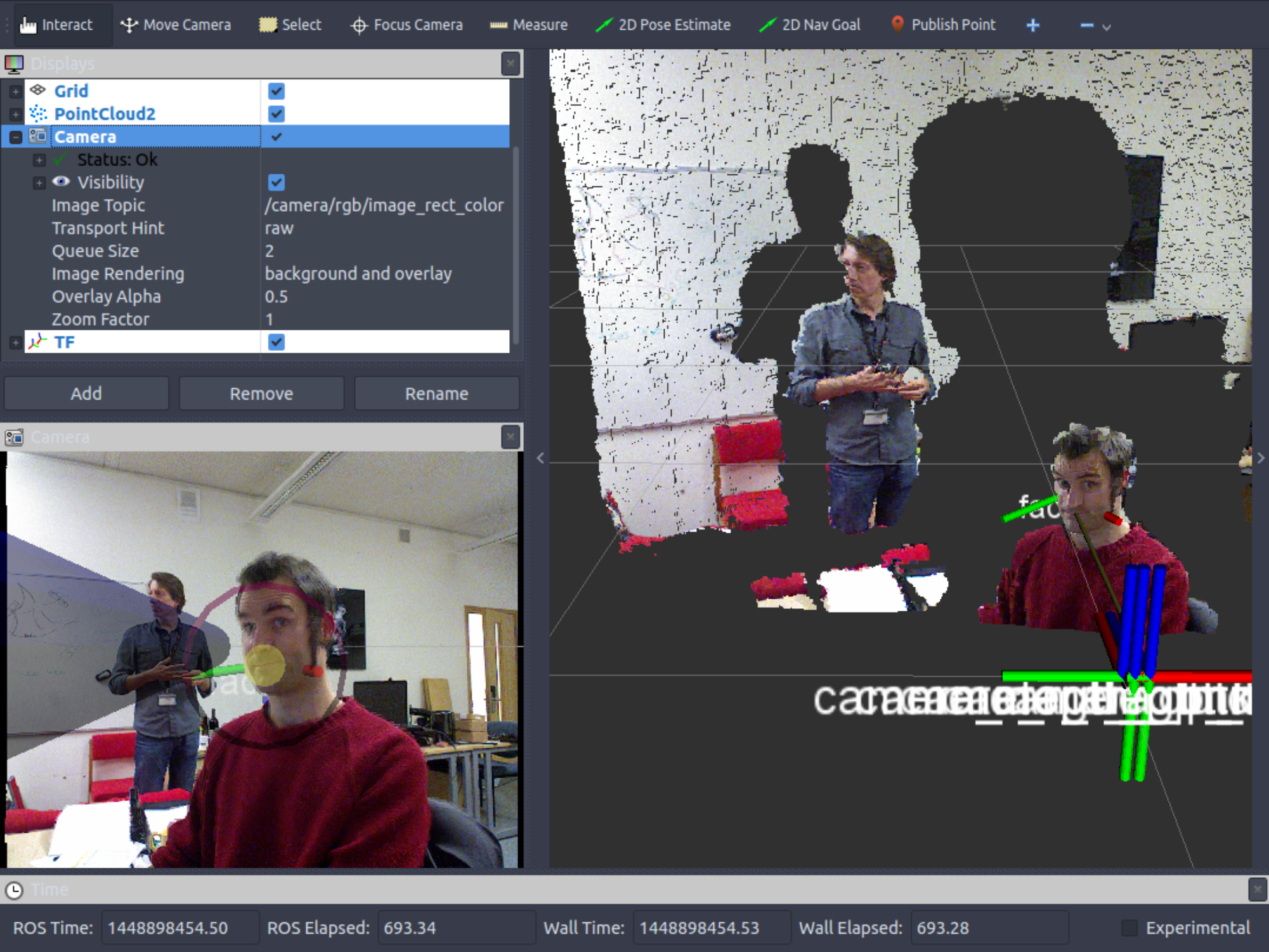
This being clarified...

HOW DOES IT LOOK LIKE?

```
$ roscore
$ rosrun rviz rviz
$ roslaunch openni_launch openni.launch
```


Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
Grid
PointCloud2
Camera
Status: Ok
Visibility
Image Topic
/camera/rgb/image_color
Transport Hint
raw
Queue Size
2
Image Rendering
background and overlay
Overlay Alpha
0.5
Zoom Factor
1
TF
Add
Remove
Rename
Camera
Time
ROS Time:
3901095.11
ROS Elapsed:
212.65
Wall Time:
1448901095.14
Wall Elapsed:
212.59
Experimental

```
$ rosrun attention_tracker estimate
                              image:=/camera/rgb/image_color
```

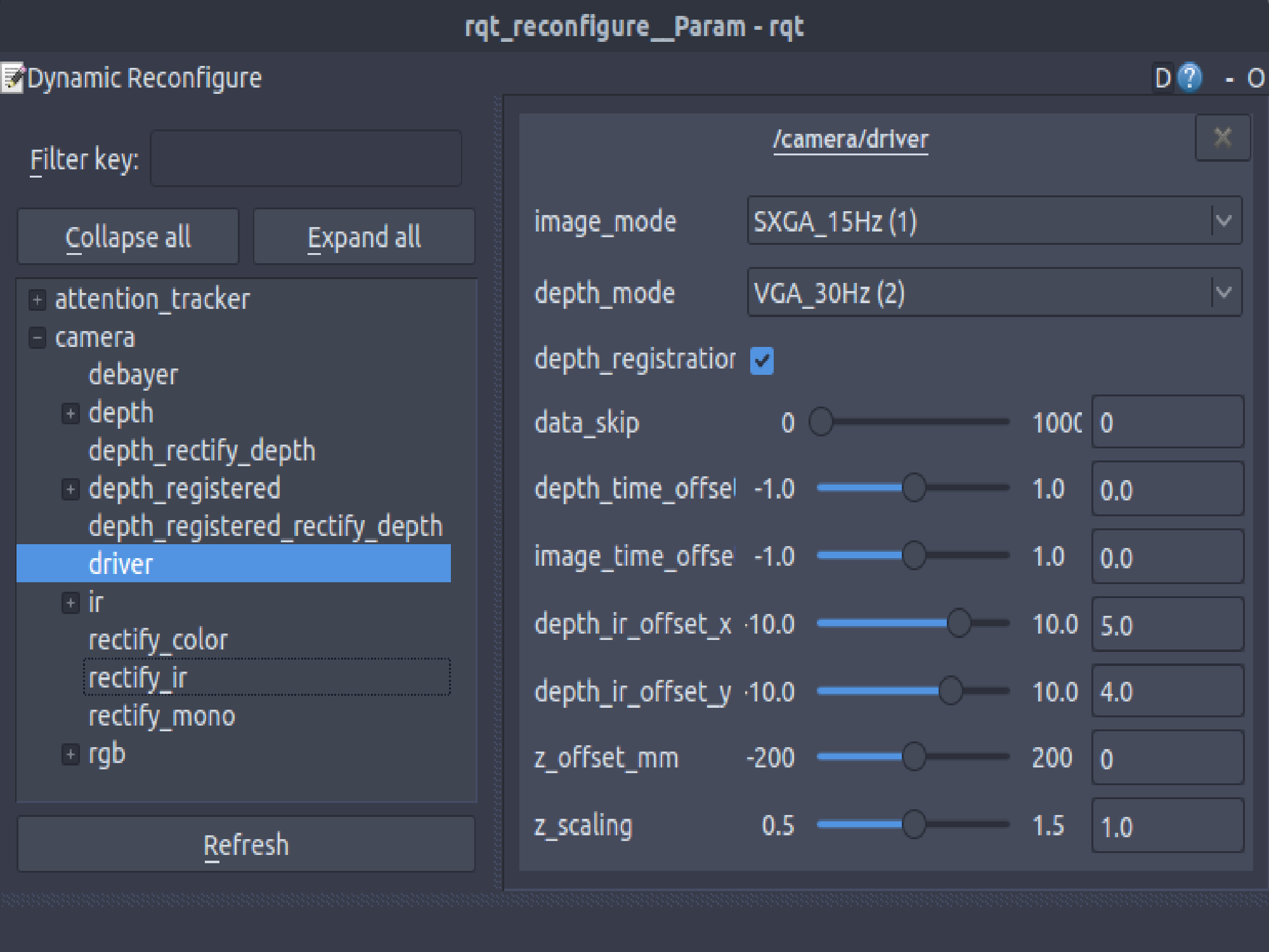

Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
Grid
PointCloud2
Camera
Status: Ok
Visibility
Image Topic
/camera/rgb/image_rect_color
Transport Hint
raw
Queue Size
2
Image Rendering
background and overlay
Overlay Alpha
0.5
Zoom Factor
1
TF
Add
Remove
Rename
Camera
Time
ROS Time:
1448898454.50
ROS Elapsed:
693.34
Wall Time:
1448898454.53
Wall Elapsed:
693.28
Experimental

```
$ rosnode list
/camera_base_link
/camera_base_link1
/camera_base_link2
/camera_base_link3
/ros_attention_tracker
/rosout
/camera/camera_nodelet_manager
/camera/debayer
/camera/depth_metric
/camera/depth_metric_rect
/camera/depth_points
/camera/depth_rectify_depth
/camera/depth_registered_hw_metric_rect
/camera/depth_registered_metric
/camera/depth_registered_rectify_depth
/camera/depth_registered_sw_metric_rect
/camera/disparity_depth
/camera/disparity_registered_hw
/camera/disparity_registered_sw
/camera/driver
```

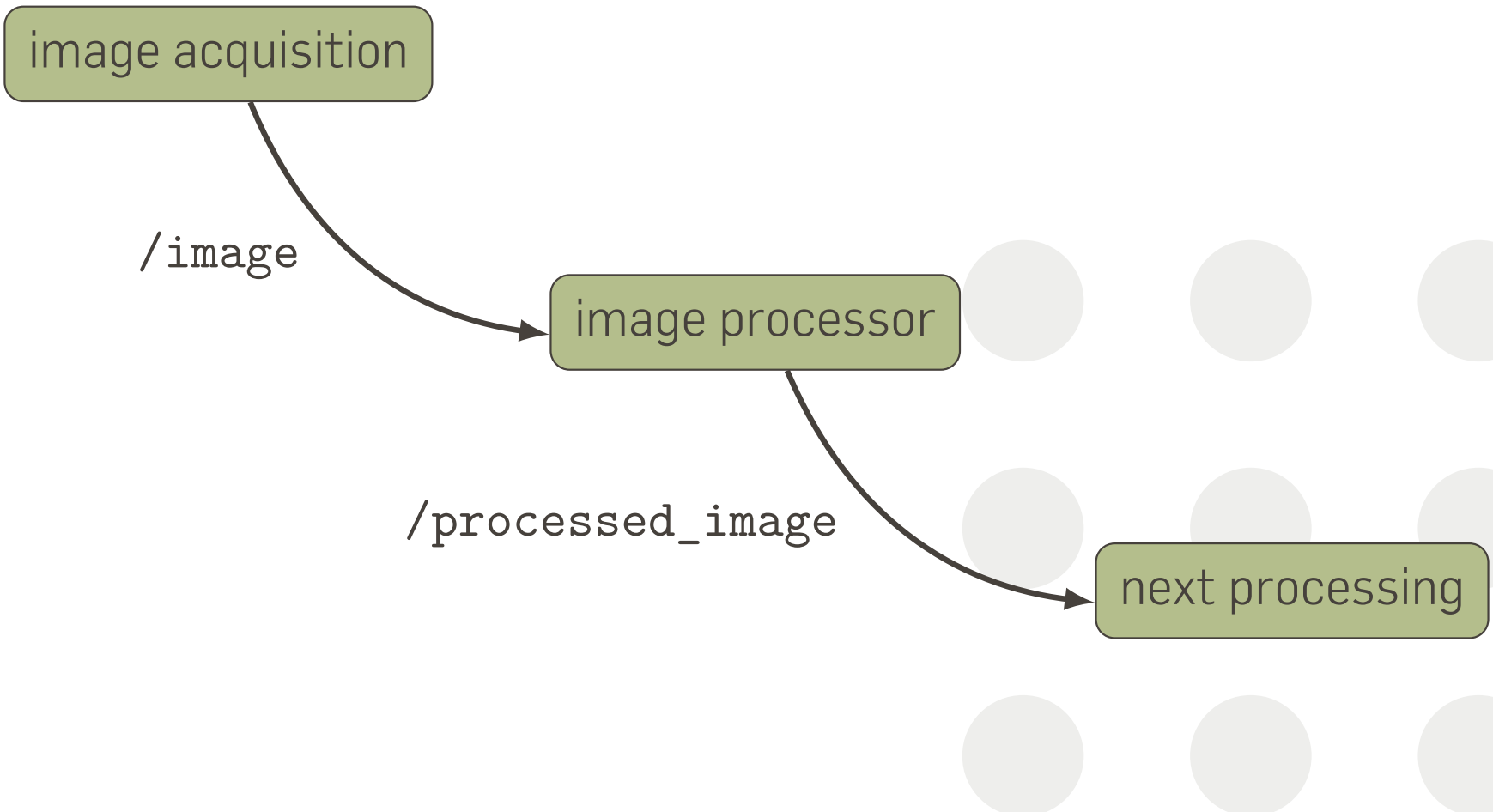
```
$ rostopic list
/camera_info
/image
/nb_detected_faces
/rosout
/rosout_agg
/tf
/camera/depth/image_rect_raw
/camera/depth/image_rect_raw/compressed
/camera/depth/image_rect_raw/compressed/parameter_descriptions
/camera/depth/image_rect_raw/compressed/parameter_updates
/camera/depth/image_rect_raw/compressedDepth
/camera/depth/image_rect_raw/compressedDepth/parameter_descripti
/camera/depth/image_rect_raw/compressedDepth/parameter_updates
/camera/depth/image_rect_raw/theora
/camera/depth/image_rect_raw/theora/parameter_descriptions
/camera/depth/image_rect_raw/theora/parameter_updates
/camera/depth_rectify_depth/parameter_descriptions
/camera/depth_rectify_depth/parameter_updates
```

```
$ rostopic echo tf
transforms:
  -
    header:
      seq: 0
      stamp:
        secs: 1449222890
        nsecs: 396561780
      frame_id: /camera_link
    child_frame_id: /camera_rgb_frame
    transform:
      translation:
        x: 0.0
        y: -0.045
        z: 0.0
      rotation:
        x: 0.0
        y: 0.0
        z: 0.0
        w: 1.0
---
```

```
$ rosrun rqt_reconfigure rqt_reconfigure
```


rqt_reconfigure__Param - rqt
Dynamic Reconfigure
Filter key:
Collapse all
Expand all
attention_tracker
camera
debayer
depth
depth_rectify_depth
depth_registered
depth_registered_rectify_depth
driver
ir
rectify_color
rectify_ir
rectify_mono
rgb
Refresh
/camera/driver
image_mode
SXGA_15Hz (1)
depth_mode
VGA_30Hz (2)
depth_registratior
data_skip
0
100(
0
depth_time_offse
-1.0
1.0
0.0
image_time_offse
-1.0
1.0
0.0
depth_ir_offset_x
-10.0
10.0
5.0
depth_ir_offset_y
-10.0
10.0
4.0
z_offset_mm
-200
200
0
z_scaling
0.5
1.5
1.0

EXAMPLE: A SIMPLE IMAGE PROCESSING PIPELINE



```python
import sys, cv2, rospy
from sensor_msgs.msg import Image
from cv_bridge import CvBridge

def on_image(image):
    cv_image = bridge.imgmsg_to_cv2(image, "bgr8")
    (rows,cols,channels) = cv_image.shape
    cv2.circle(cv_image, (cols/2,rows/2), 50,(0,0,255), -1)
    cv2.imshow("Image window", cv_image)
    cv2.waitKey(3)
    image_pub.publish(bridge.cv2_to_imgmsg(cv_image, "bgr8"))

rospy.init_node('image_processor')
bridge = CvBridge()
image_sub = rospy.Subscriber("image",Image, on_image)
image_pub = rospy.Publisher("processed_image",Image)

while not rospy.is_shutdown():
    rospy.spin()
```

```
$ roslaunch gscam v4l.launch
$ python image_processor.py image:=/v4l/camera/image_raw
$ rosrun image_view image_view image:=/processed_image
```

THE KEY CONCEPTS

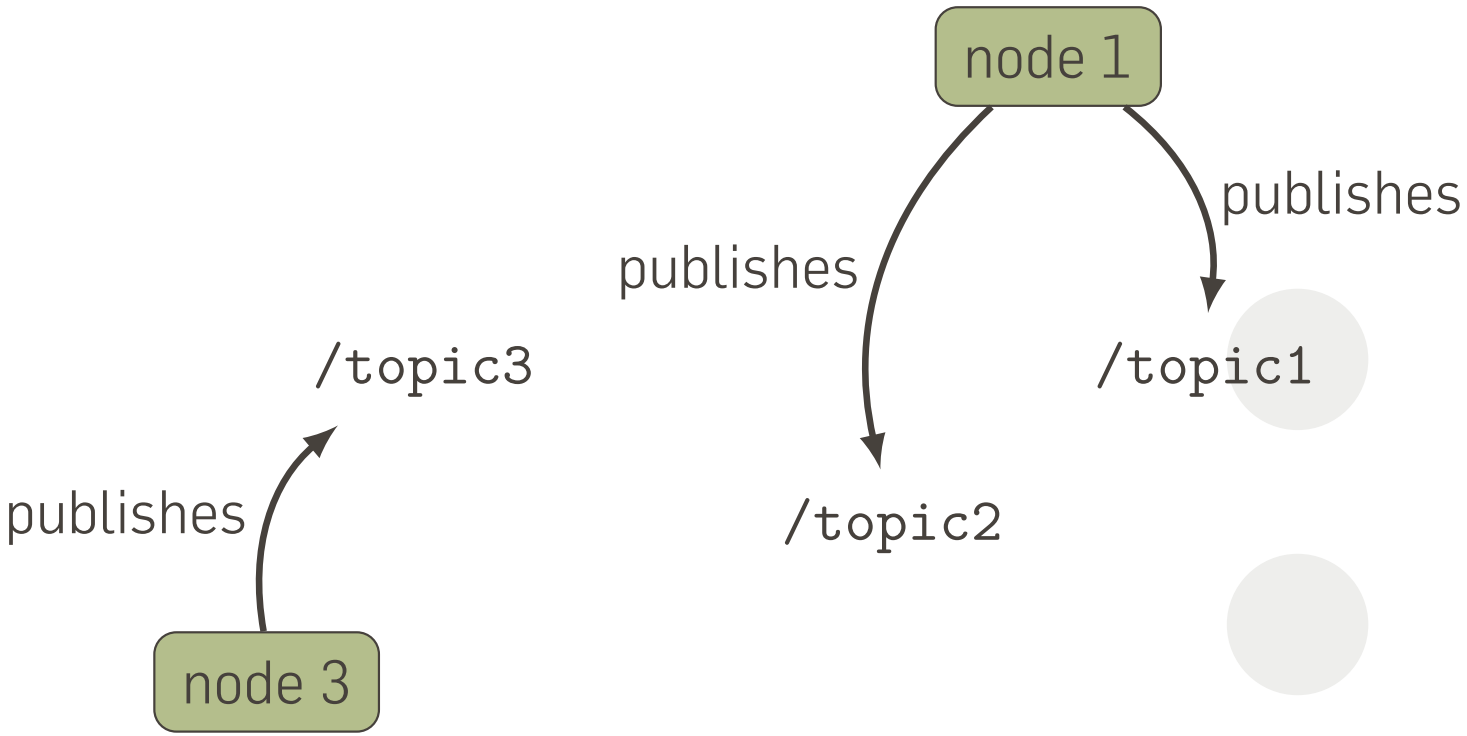
TALKING NODES

node 1

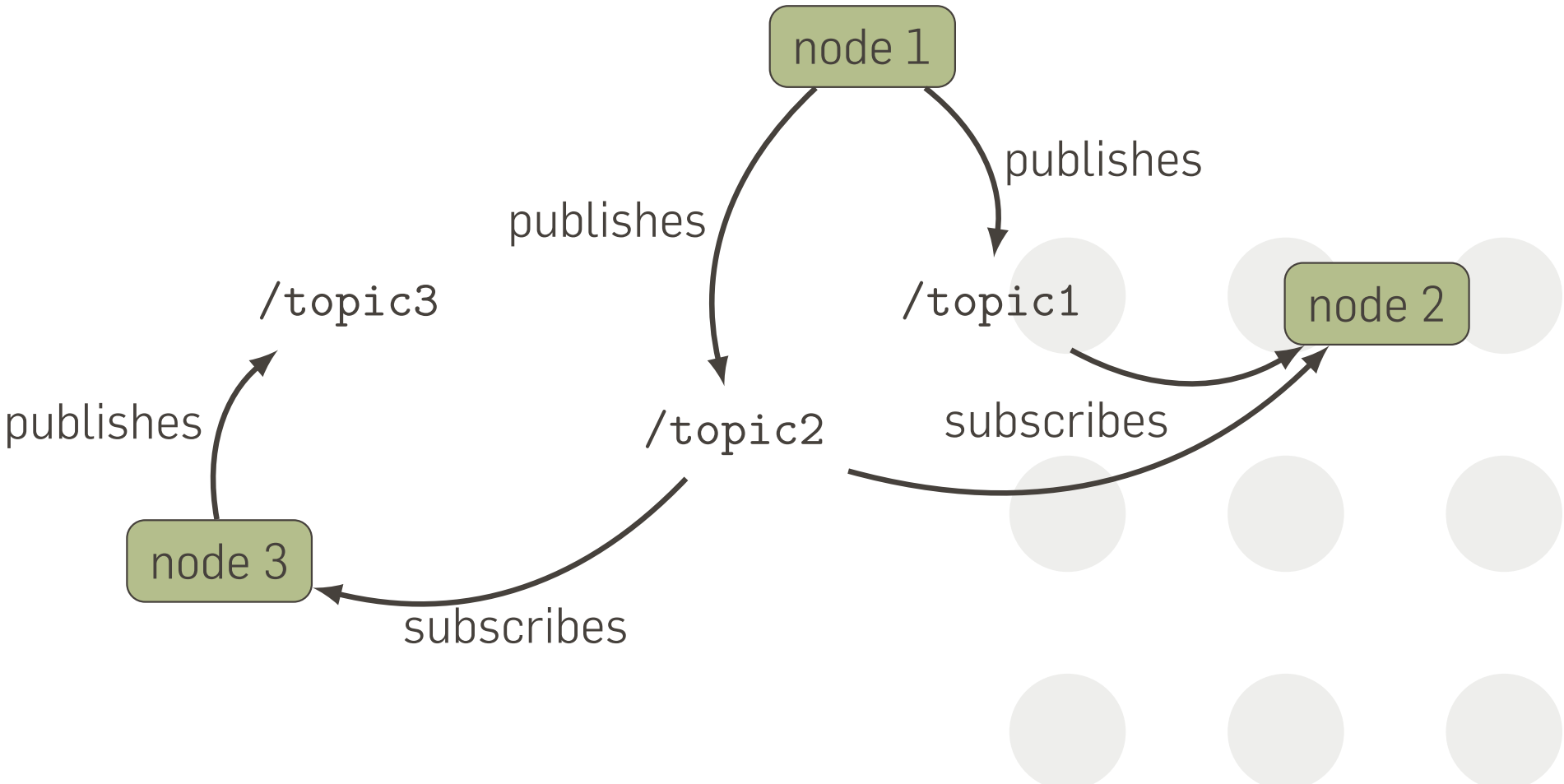
node 2

node 3

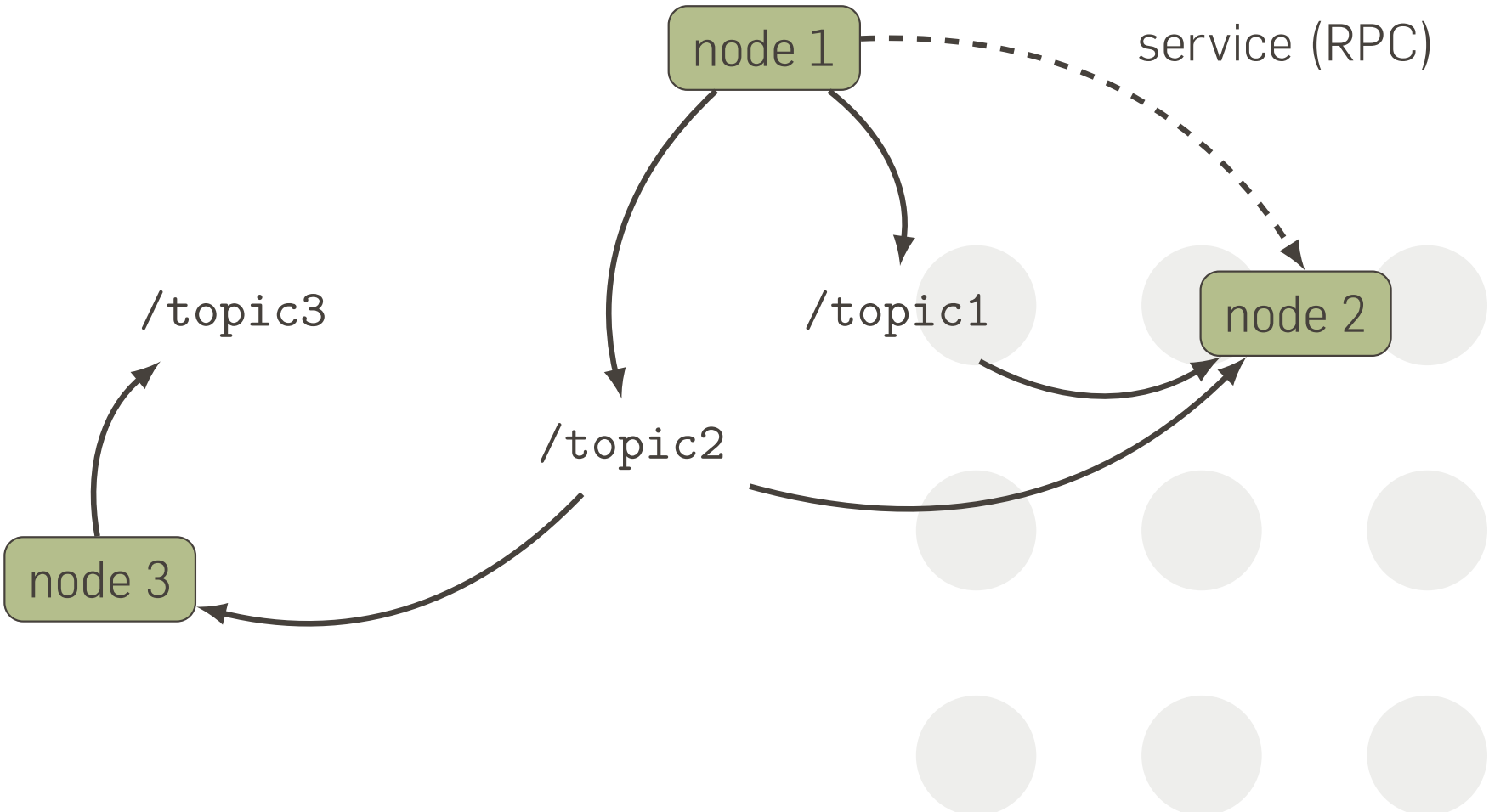
TALKING NODES



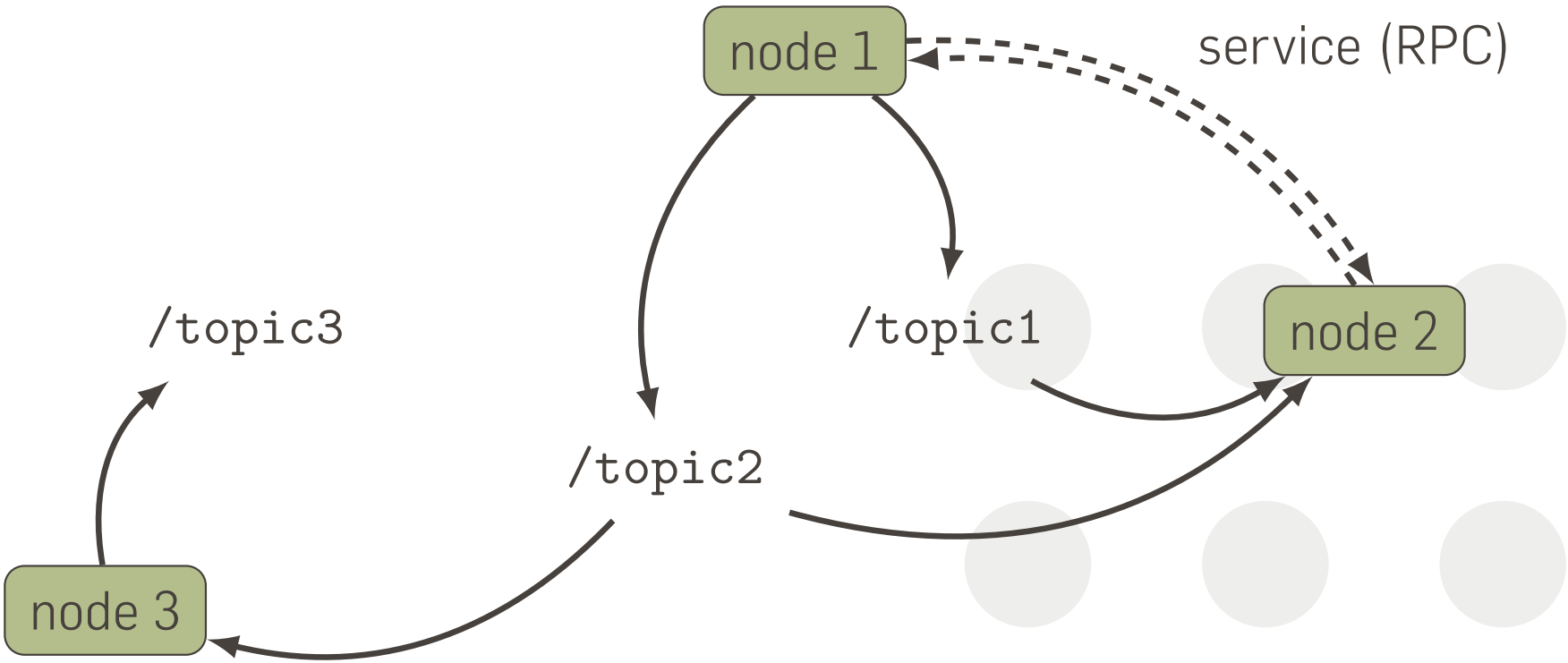
TALKING NODES



TALKING NODES

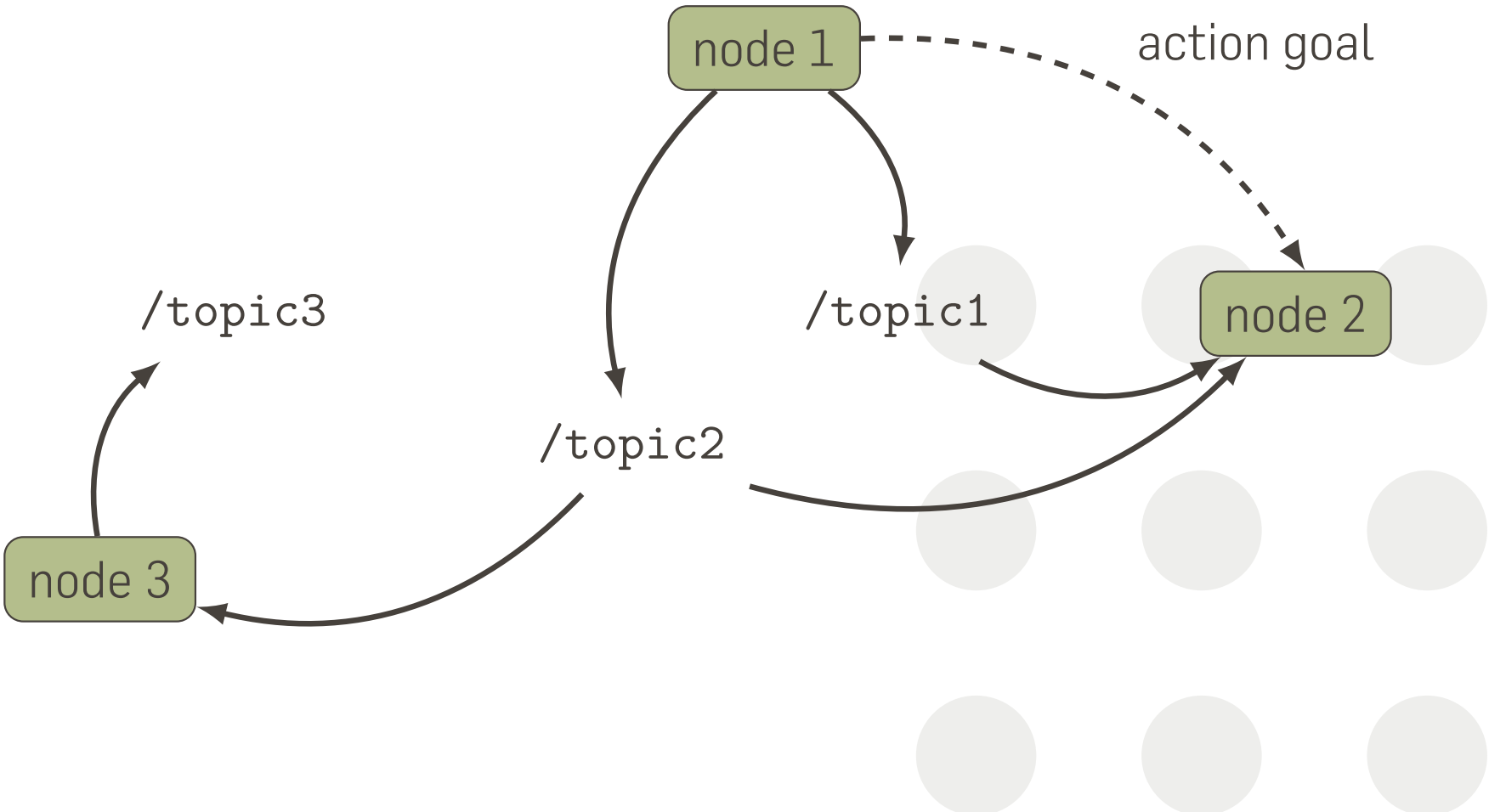


TALKING NODES

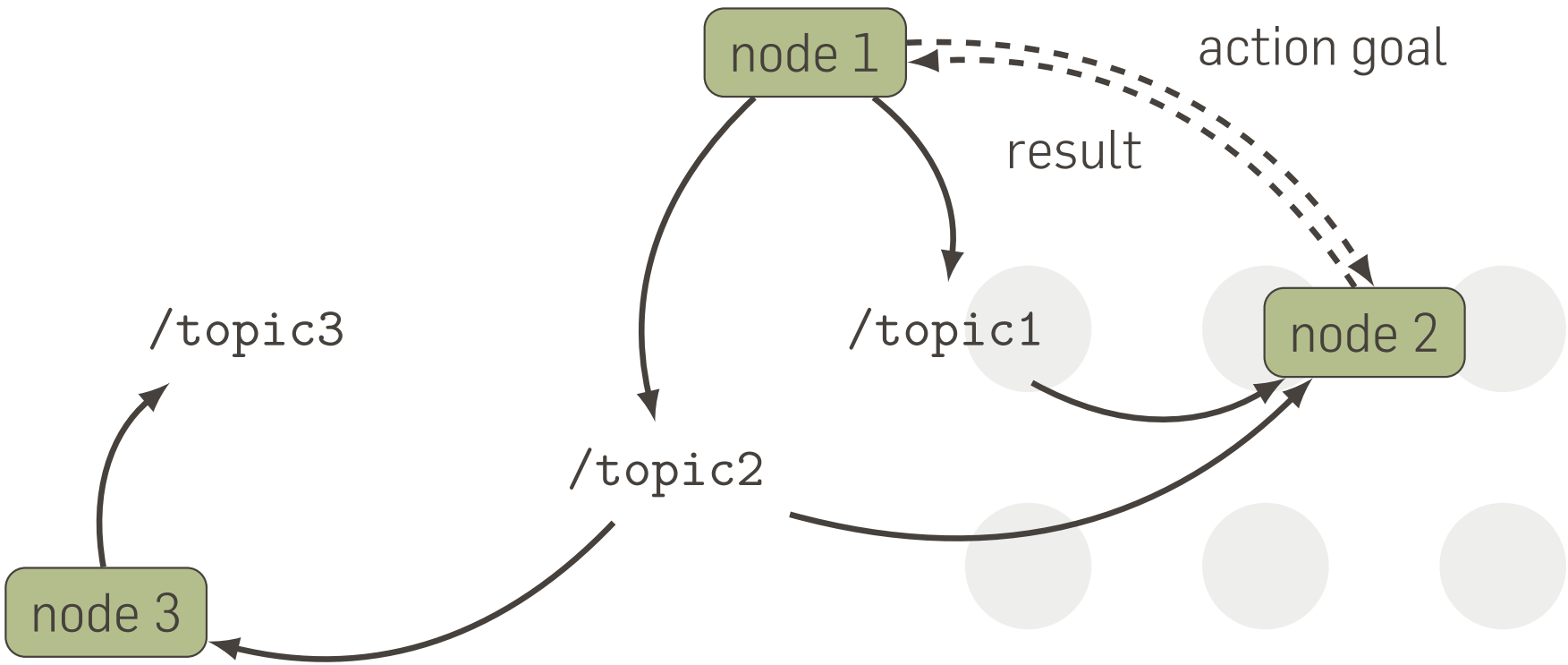


Services: **synchronous**

TALKING NODES

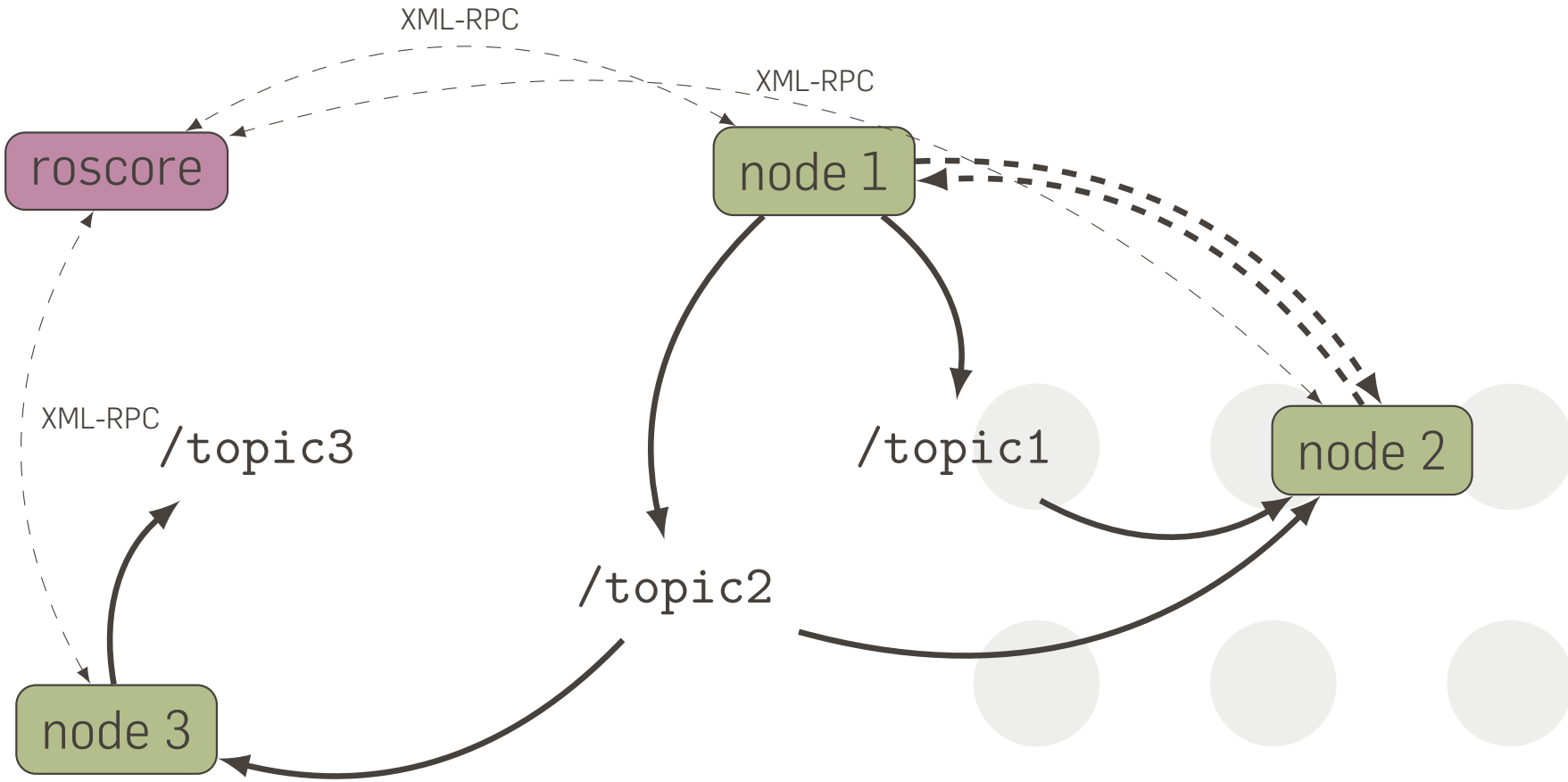


TALKING NODES



Actions: **asynchronous**

TALKING NODES



`ROS_MASTER_URI=http://<host>:<port>`

NODELETS

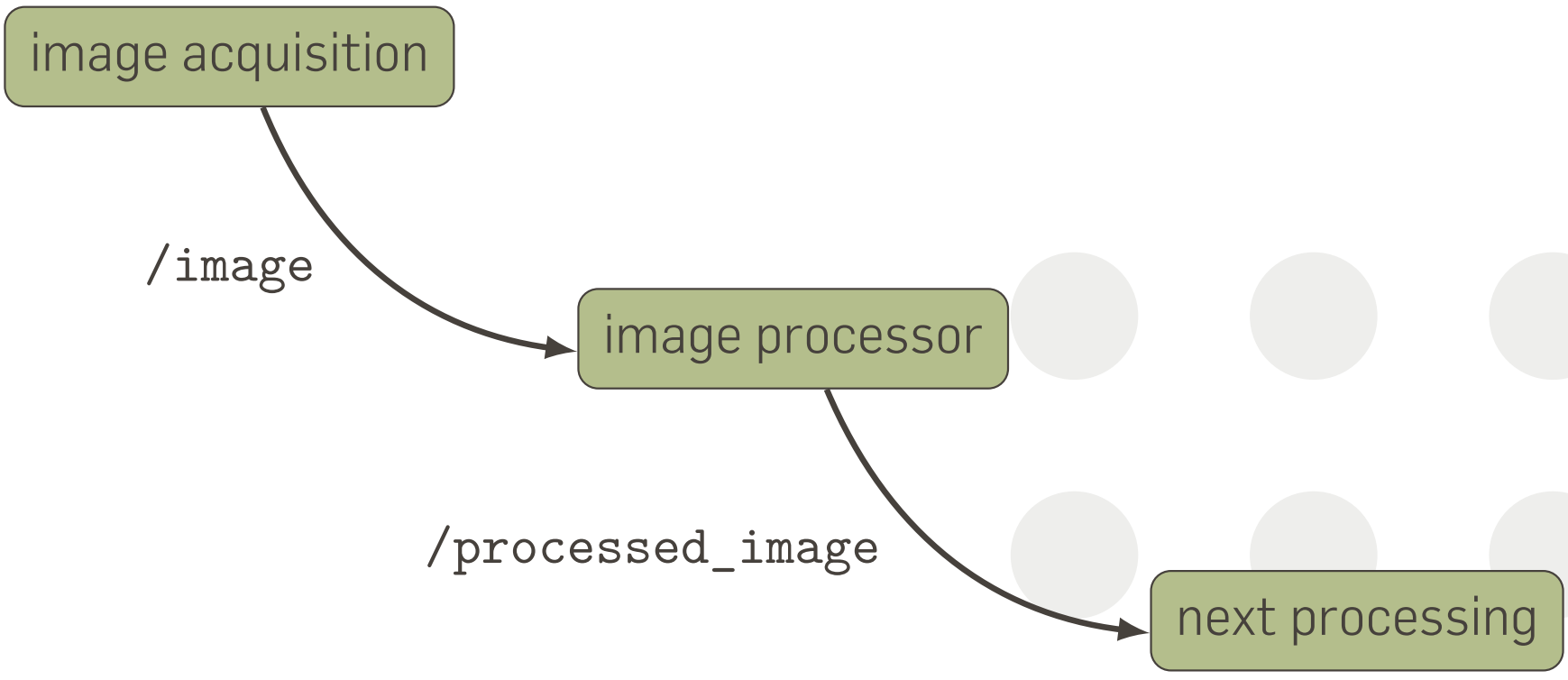


image acquisition
/image
image processor
/processed_image
next processing

PACKAGES, NODES AND LAUNCH FILES

```
$ cd my_package
$ ls
CMakeLists.txt
cfg/
include/
launch/
msgs/
nodes/
package.xml
src/
$ rosrun my_package my_node
```

PACKAGES, NODES AND LAUNCH FILES

```
<launch>

  <arg name="head_frame" default="head_camera"/>
  <arg name="image" default="/camera/rgb/image_raw"/>

  <node pkg="attention_tracker" type="focus" />

  <node pkg="attention_tracker" type="estimate">
    <param name="face_model"
        value="$(find attention_tracker)/model.dat" />
    <remap from="/image" to="$(arg image)"/>
  </node>

</launch>
```

```
$ roslaunch my_package my_system.launch
```

MESSAGES

```
$ rosmsg show geometry_msgs/Pose
geometry_msgs/Point position
  float64 x
  float64 y
  float64 z
geometry_msgs/Quaternion orientation
  float64 x
  float64 y
  float64 z
  float64 w
```

MESSAGES

```
$ rosmsg show sensor_msgs/Image
std_msgs/Header header
    uint32 seq
    time stamp
    string frame_id
uint32 height
uint32 width
string encoding
uint8 is_bigendian
uint32 step
uint8[] data
```

MESSAGES

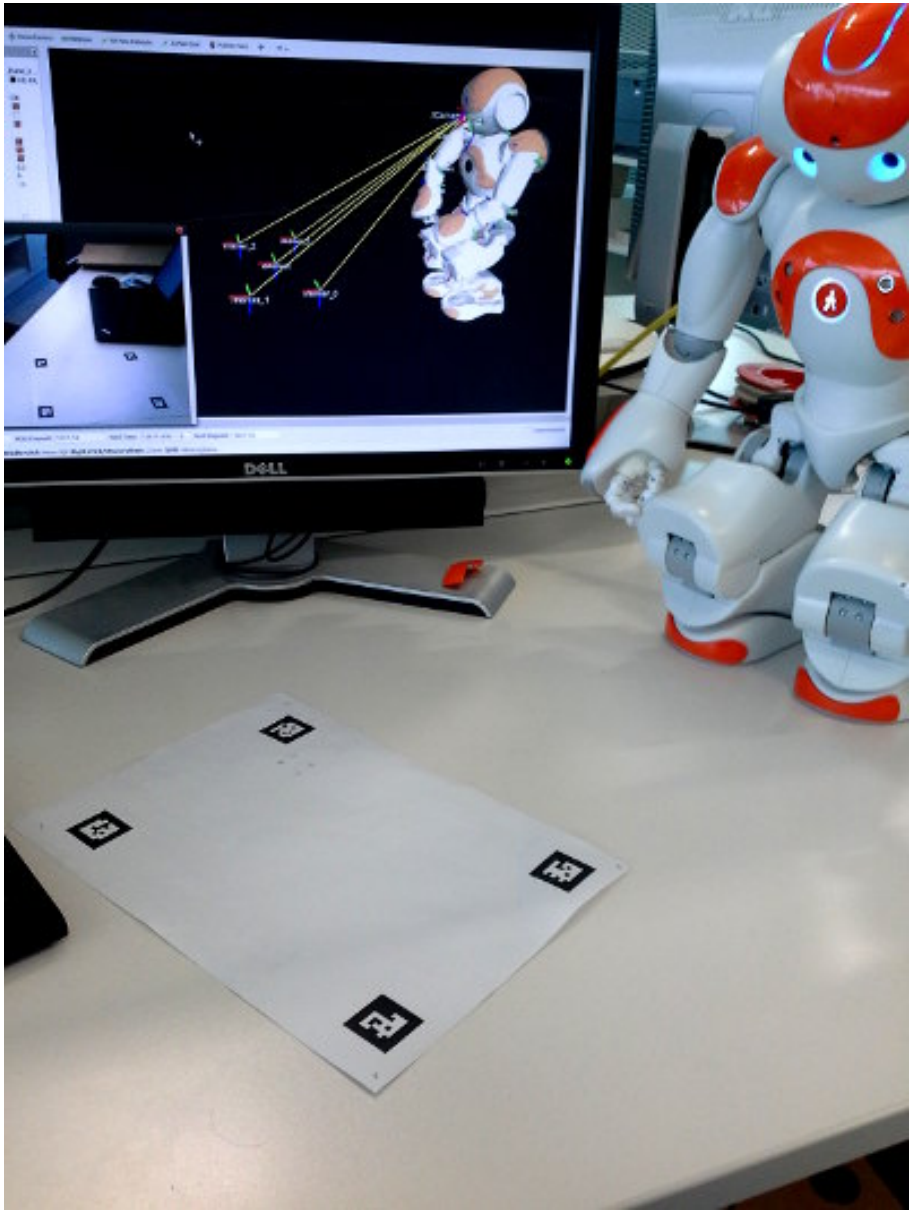
```
$ rostopic echo /camera/image_raw
header:
    seq: 56
    stamp:
      secs: 1449243166
      nsecs: 415330019
    frame_id: /camera_frame
height: 720
width: 1280
encoding: rgb8
is_bigendian: 0
step: 3840
data: [32, 57, 51, 36, 61, 55, 41, 63, 60,...
```

ROS VS YARP TERMINOLOGY

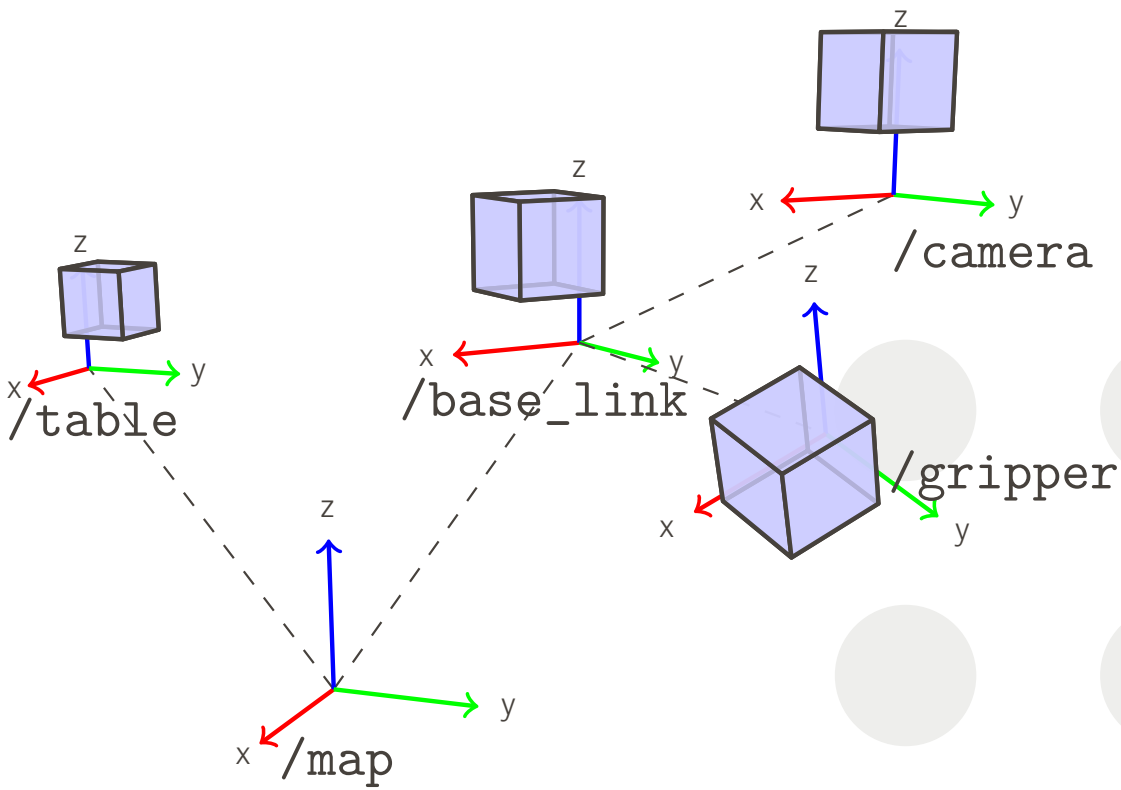
YARP	ROS
`Port`	`topic`
`RpcClient/RpcServer`	`action` (async) or `service` (sync)
`Bottle`	`message`
`yarpserver`	`roscore`

Connections of ports: explicit with YARP, implicit with ROS

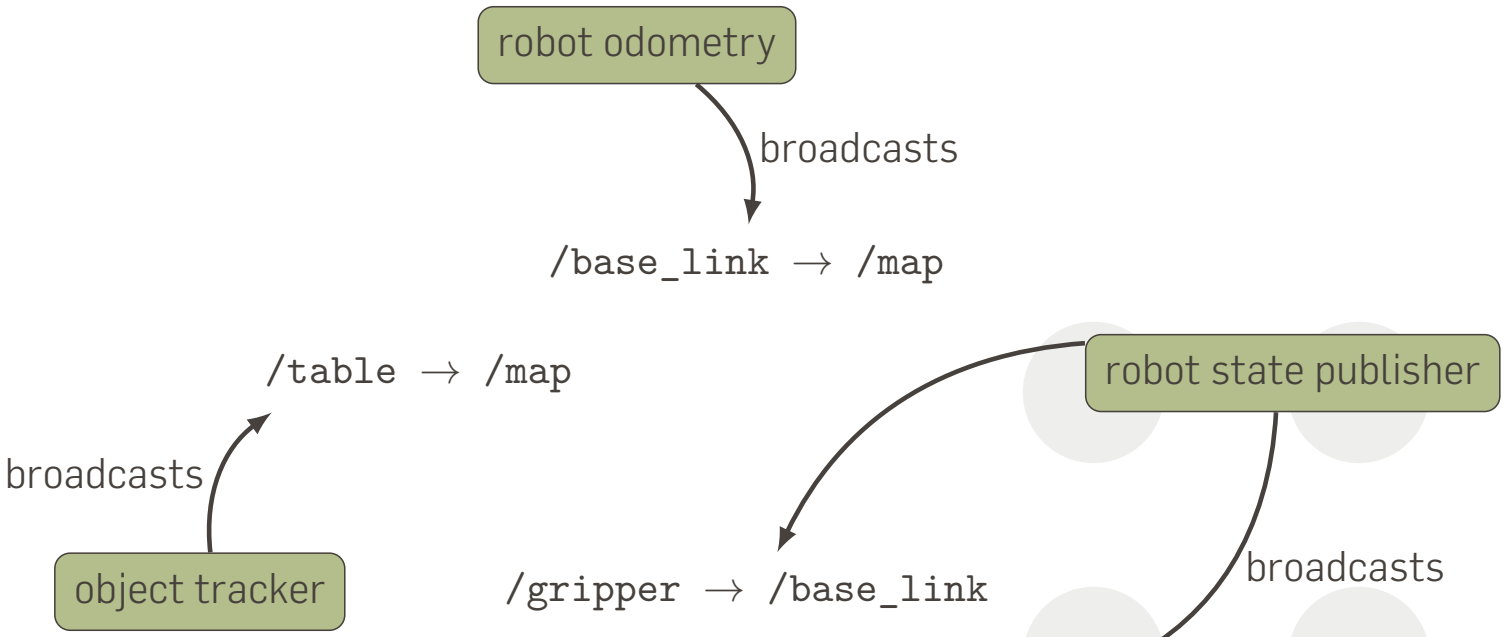
TRANSFORMATIONS



TRANSFORMATIONS



FRAMES


robot odometry
broadcasts
/base_link → /map
/table → /map
broadcasts
object tracker
robot state publisher
/gripper → /base_link
broadcasts
/camera → /base_link

```
1  #include <ros/ros.h>
2  #include <tf/transform_broadcaster.h>
3
4  int main(int argc, char** argv){
5    float x=0.f,y=0.f,theta=0.f;
6
7    ros::init(argc, argv, "my_tf_broadcaster");
8    tf::TransformBroadcaster br;
9    ros::Rate rate(10); // 10 hz
10
11   while (ros::ok()) {
12     tf::Transform transform(tf::Quaternion(0, 0, theta),
13                             tf::Vector3(x, y, 0.0));
14     br.sendTransform(tf::StampedTransform(transform,
15                                           ros::Time::now(),
16                                           "map", "my_robot"));
17     x++;
18     rate.sleep();
19   }
20   return 0;
21 };
```

```
$ g++ tf.cpp -o tf -lroscpp -lrostime -lboost_system -ltf
$ ./tf
```

```
$ rostopic echo tf
transforms:
  -
    header:
      seq: 0
      stamp:
        secs: 1449488936
        nsecs: 480597909
      frame_id: map
    child_frame_id: my_robot
    transform:
      translation:
        x: 239.0
        y: 0.0
        z: 0.0
      rotation:
        x: 0.0
        y: 0.0
        z: 0.0
        w: 1.0
---
```

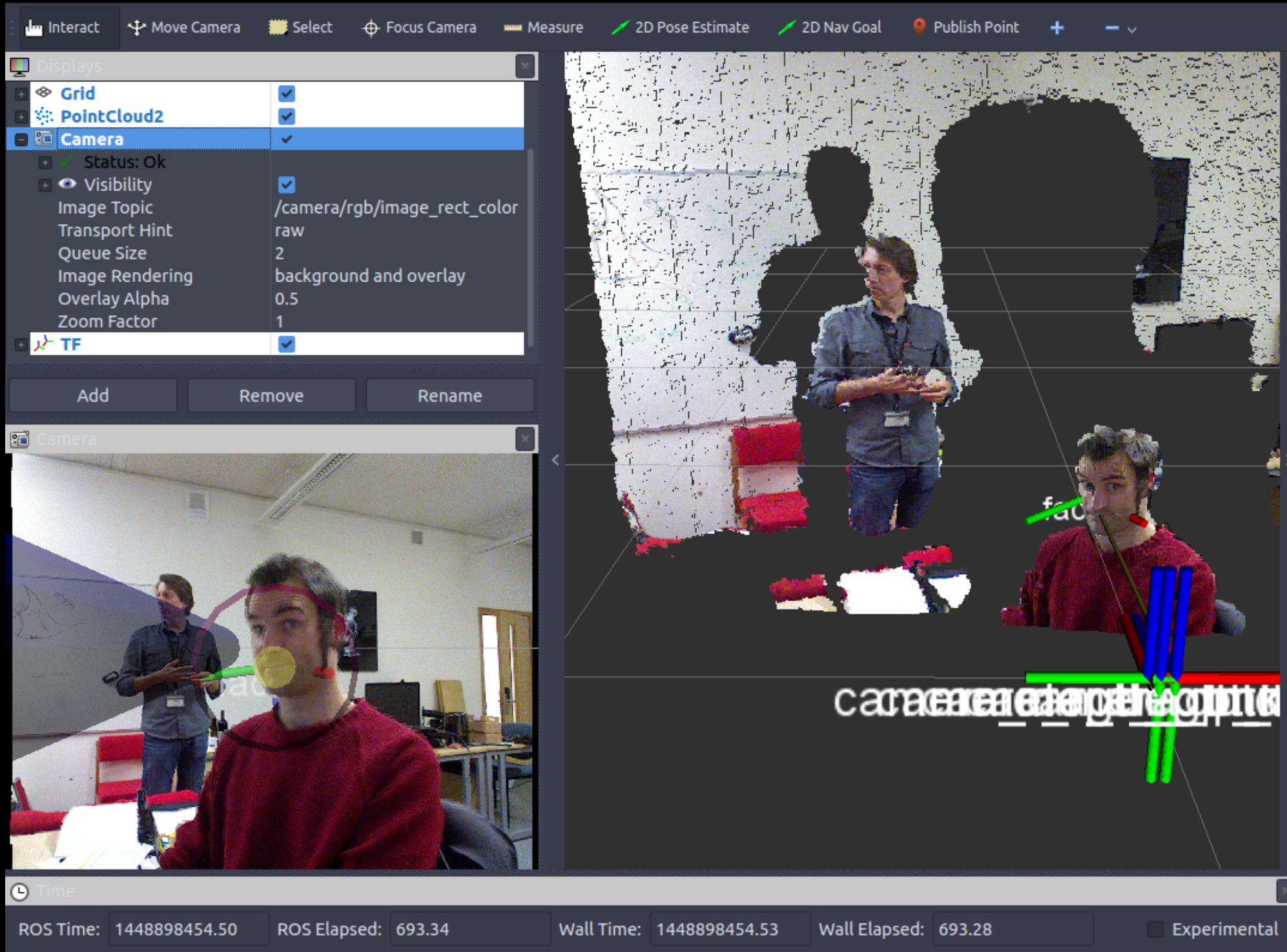
TO SUMMARIZE: KEY CONCEPTS

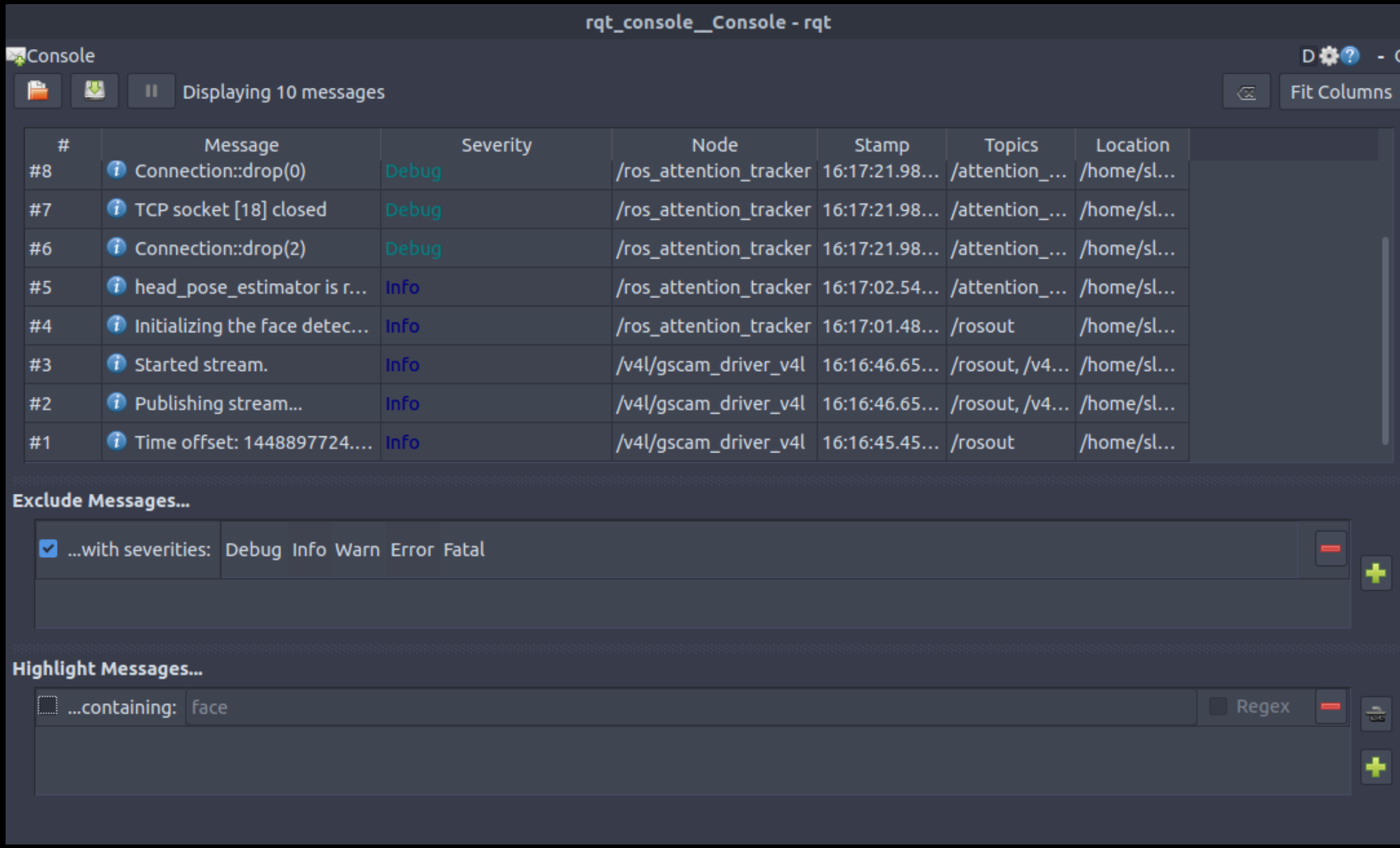
- Package, Node, Nodelet, Launch files
- Master
- Messages
- Topics
- Services
- Actions
- Transformations/frames

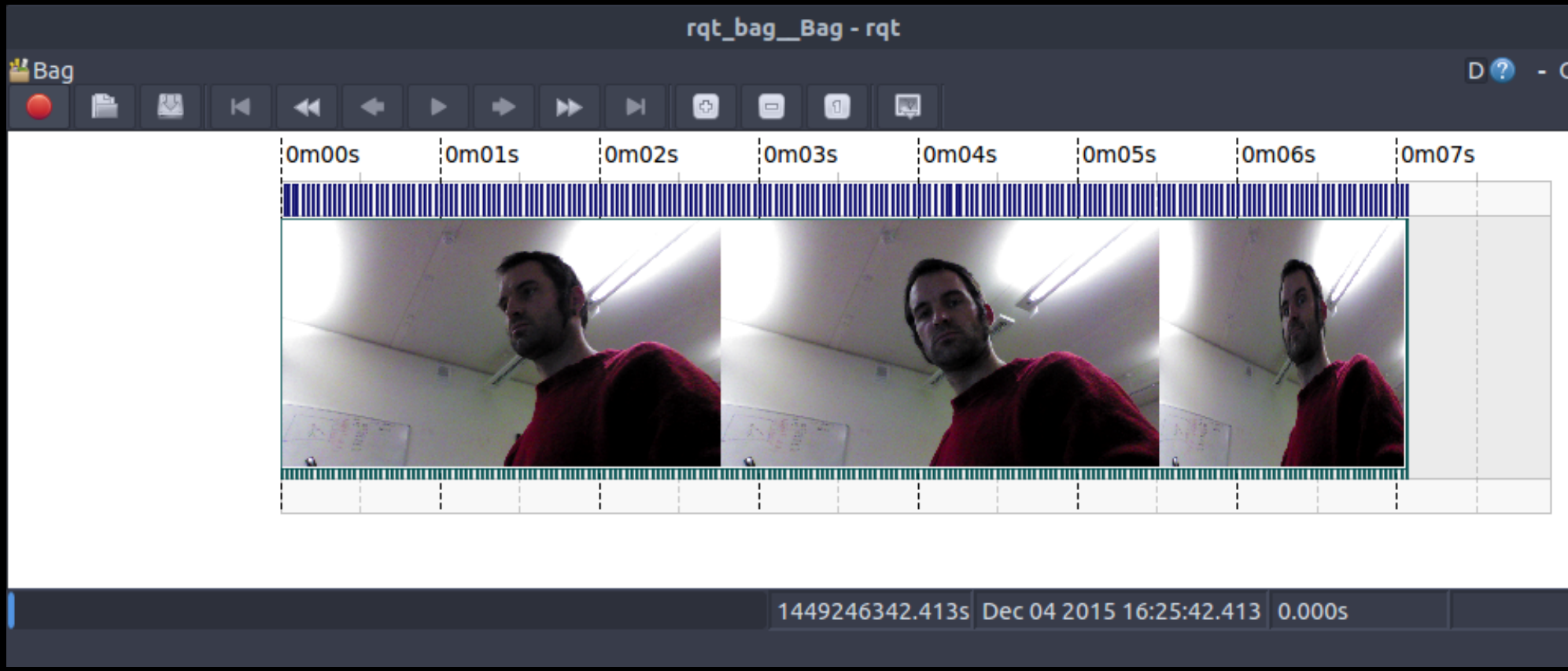
TOOLING

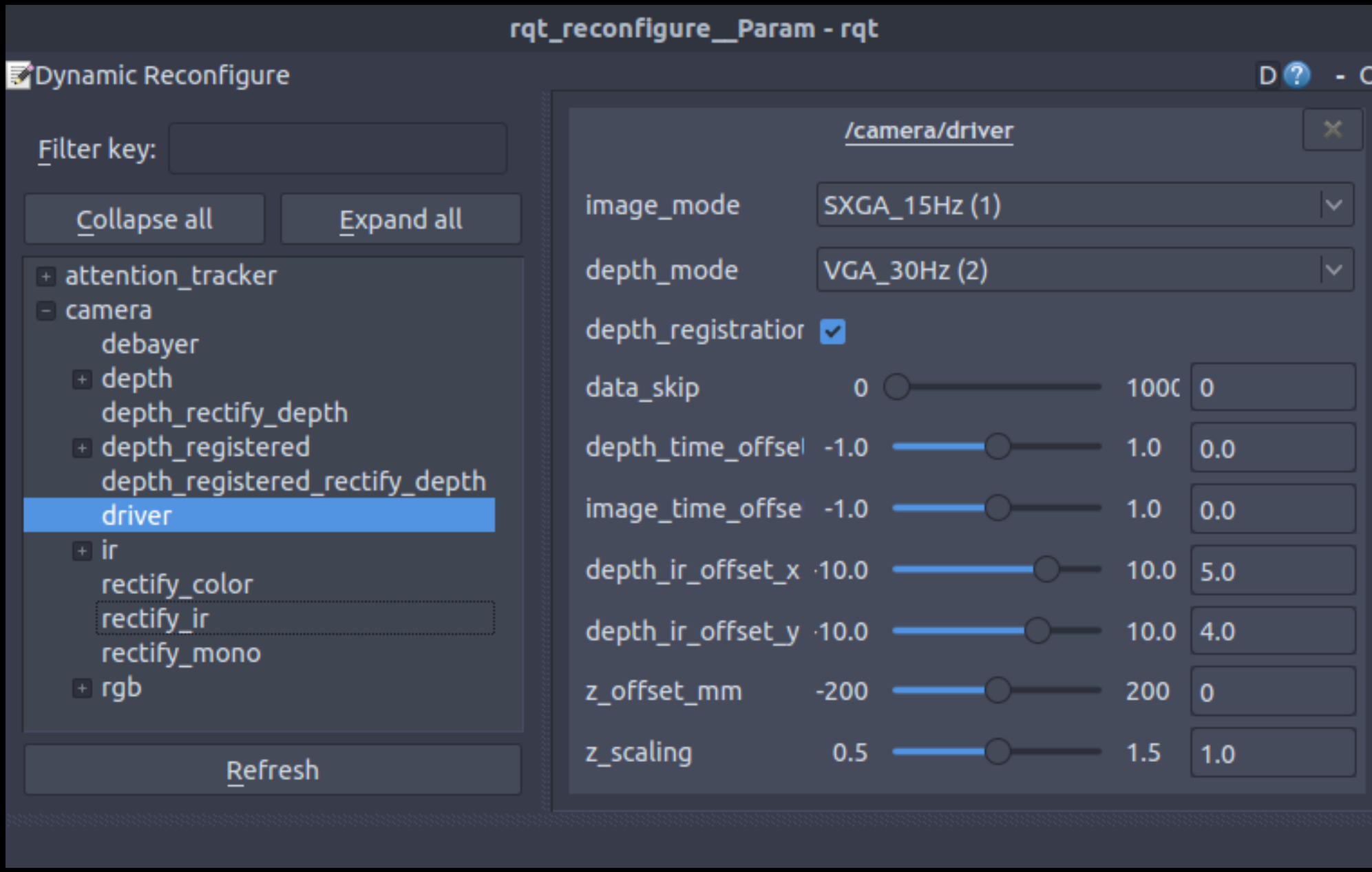
TOOLS

`rviz`	versatile 2D/3D visualization
`rosconsole`	Centralized logging
`rosbag`	Record and replay messages
`rqt_reconfigure`	Live configuration of nodes
`rqt_diagnostics`	Standardized diagnostics
`rosgraph`	plots the node network
+ tons of introspection tools	Print out/publish/call messages, services, nodes


Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
Grid
PointCloud2
Camera
Status: Ok
Visibility
Image Topic
/camera/rgb/image_rect_color
Transport Hint
raw
Queue Size
2
Image Rendering
background and overlay
Overlay Alpha
0.5
Zoom Factor
1
TF
Add
Remove
Rename
Camera
Time
ROS Time:
1448898454.50
ROS Elapsed:
693.34
Wall Time:
1448898454.53
Wall Elapsed:
693.28
Experimental


rqt_console__Console - rqt
Console
Displaying 10 messages
Fit Columns
Message Severity Node Stamp Topics Location
#8 Connection::drop(0) Debug /ros_attention_tracker 16:17:21.98... /attention_... /home/sl...
#7 TCP socket [18] closed Debug /ros_attention_tracker 16:17:21.98... /attention_... /home/sl...
#6 Connection::drop(2) Debug /ros_attention_tracker 16:17:21.98... /attention_... /home/sl...
#5 head_pose_estimator is r... Info /ros_attention_tracker 16:17:02.54... /attention_... /home/sl...
#4 Initializing the face detec... Info /ros_attention_tracker 16:17:01.48... /rosout /home/sl...
#3 Started stream. Info /v4l/gscam_driver_v4l 16:16:46.65... /rosout, /v4... /home/sl...
#2 Publishing stream... Info /v4l/gscam_driver_v4l 16:16:46.65... /rosout, /v4... /home/sl...
#1 Time offset: 1448897724.... Info /v4l/gscam_driver_v4l 16:16:45.45... /rosout /home/sl...
Exclude Messages...
...with severities: Debug Info Warn Error Fatal
Highlight Messages...
...containing: face
Regex


rqt_bag__Bag - rqt
Bag
0m00s
0m01s
0m02s
0m03s
0m04s
0m05s
0m06s
0m07s
1449246342.413s
Dec 04 2015 16:25:42.413
0.000s

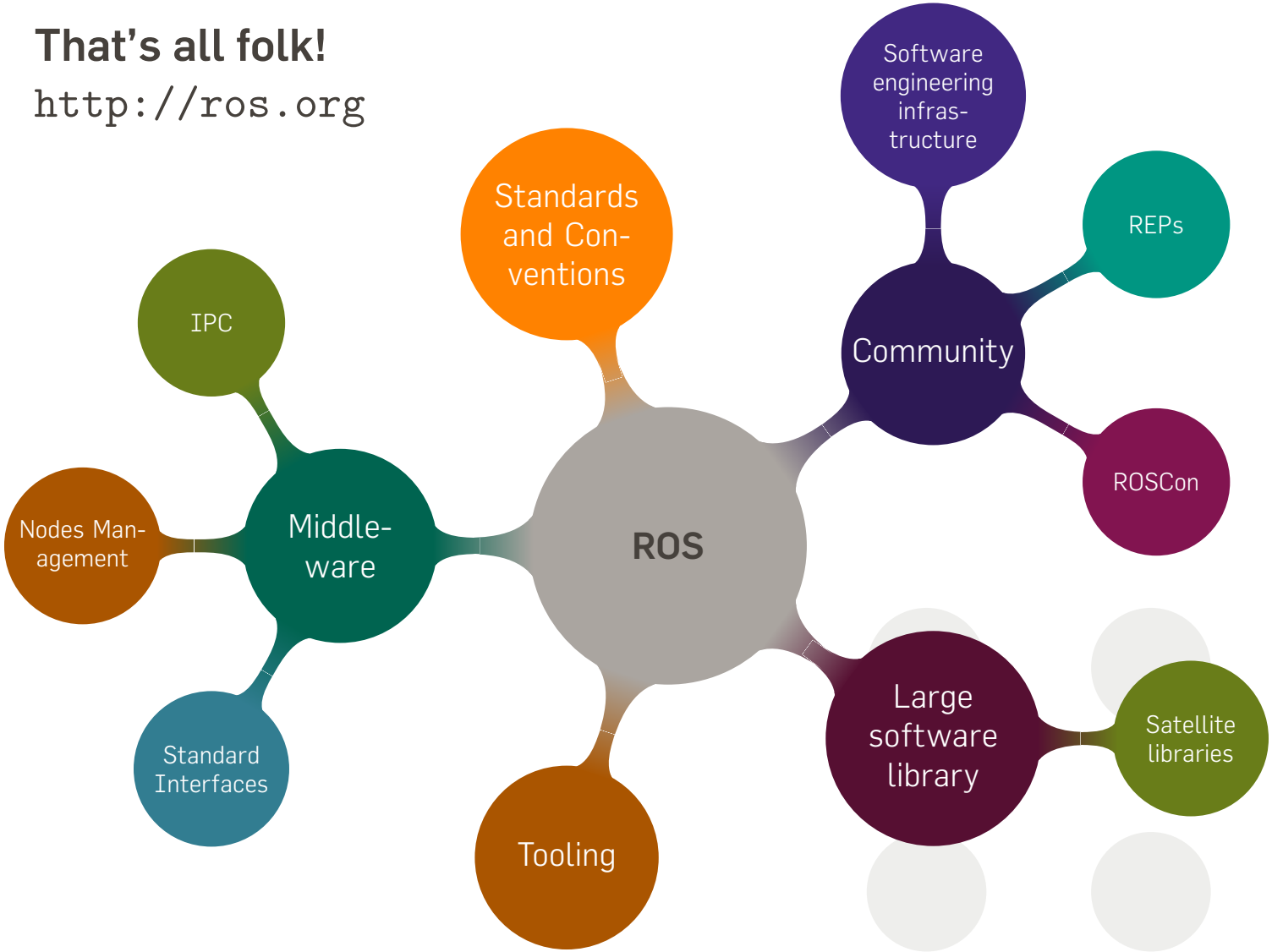

rqt_reconfigure__Param - rqt
Dynamic Reconfigure
Filter key:
Collapse all
Expand all
attention_tracker
camera
debayer
depth
depth_rectify_depth
depth_registered
depth_registered_rectify_depth
driver
ir
rectify_color
rectify_ir
rectify_mono
rgb
Refresh
/camera/driver
image_mode
SXGA_15Hz (1)
depth_mode
VGA_30Hz (2)
depth_registration
data_skip
0
0
depth_time_offset
-1.0
1.0
0.0
image_time_offset
-1.0
1.0
0.0
depth_ir_offset_x
-10.0
10.0
5.0
depth_ir_offset_y
-10.0
10.0
4.0
z_offset_mm
-200
200
0
z_scaling
0.5
1.5
1.0

ROS VS YARP TOOLS

YARP	ROS
`yarpscope/yarpview`	`rviz`
`yarpdatadumper/yarpdataplayer`	`rosbag`
`yarpmanager`	`roslaunch`

That's all folk!

`http://ros.org`



Slides available online:

`www.github.com/severin-lemaignan/ros-presentation`