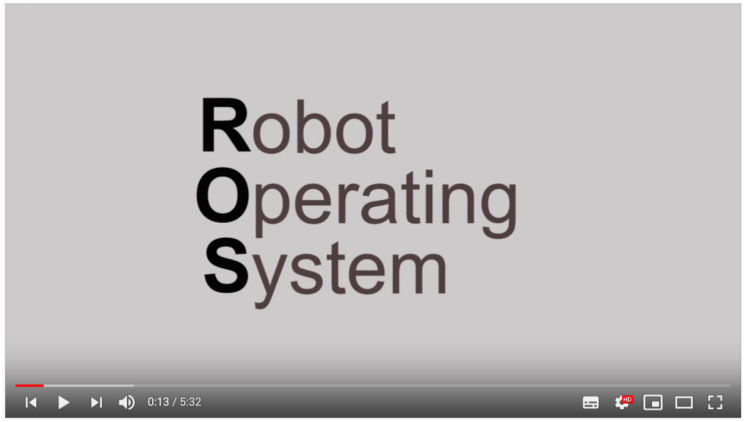
Ein kurze Einführung in ROS

- Was ist ROS
- Interprozesskommunikation
- Werkzeuge:
  Visualisierung, Simulation, Transformationen, ...
- Pakete: Navigation, Mapping, ...
- Referenzen

ROS Videos

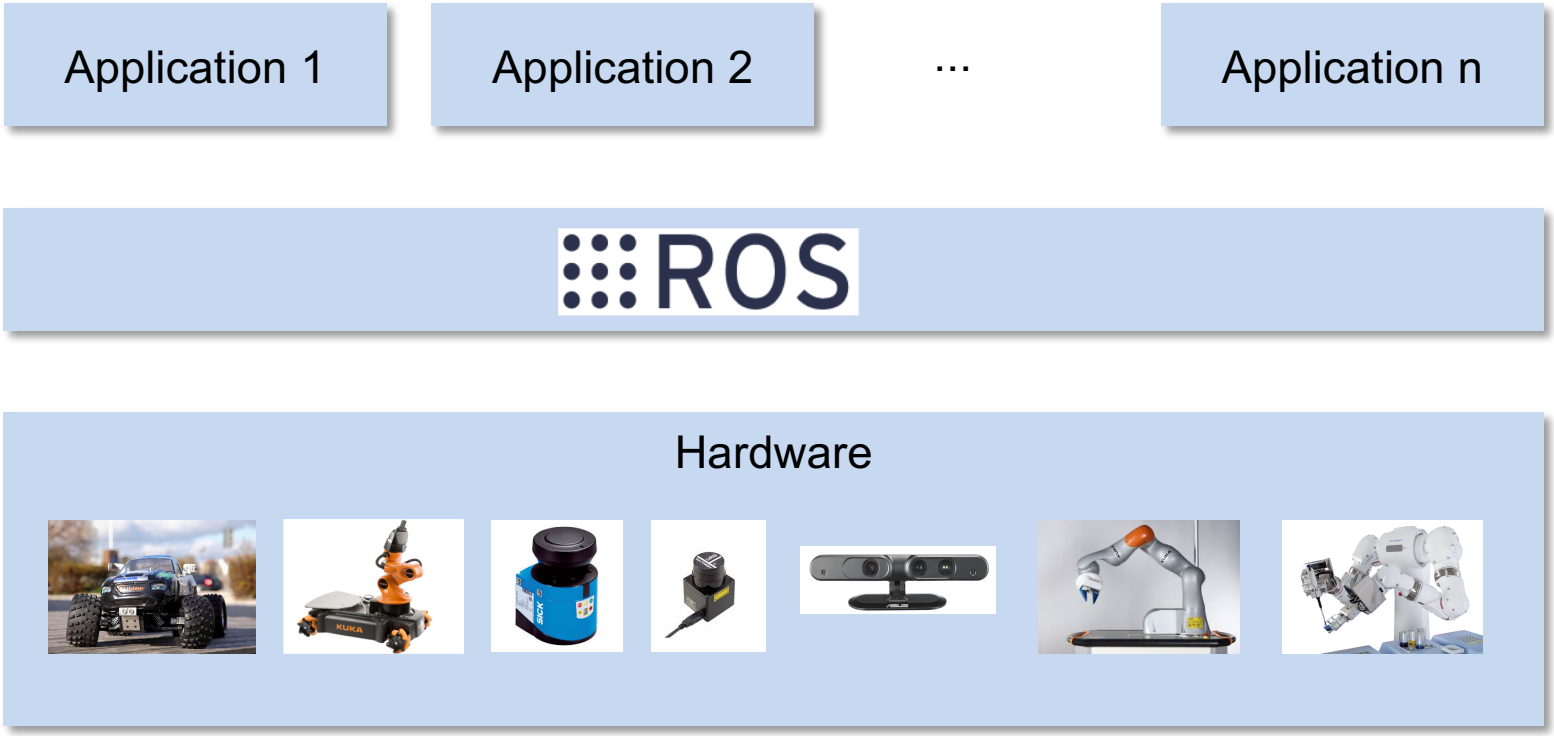
https://www.youtube.com/watch?v=mDwZ21Zia8s



What is ROS (short description)
https://www.youtube.com/watch?v=UL1_Ue4rUWs

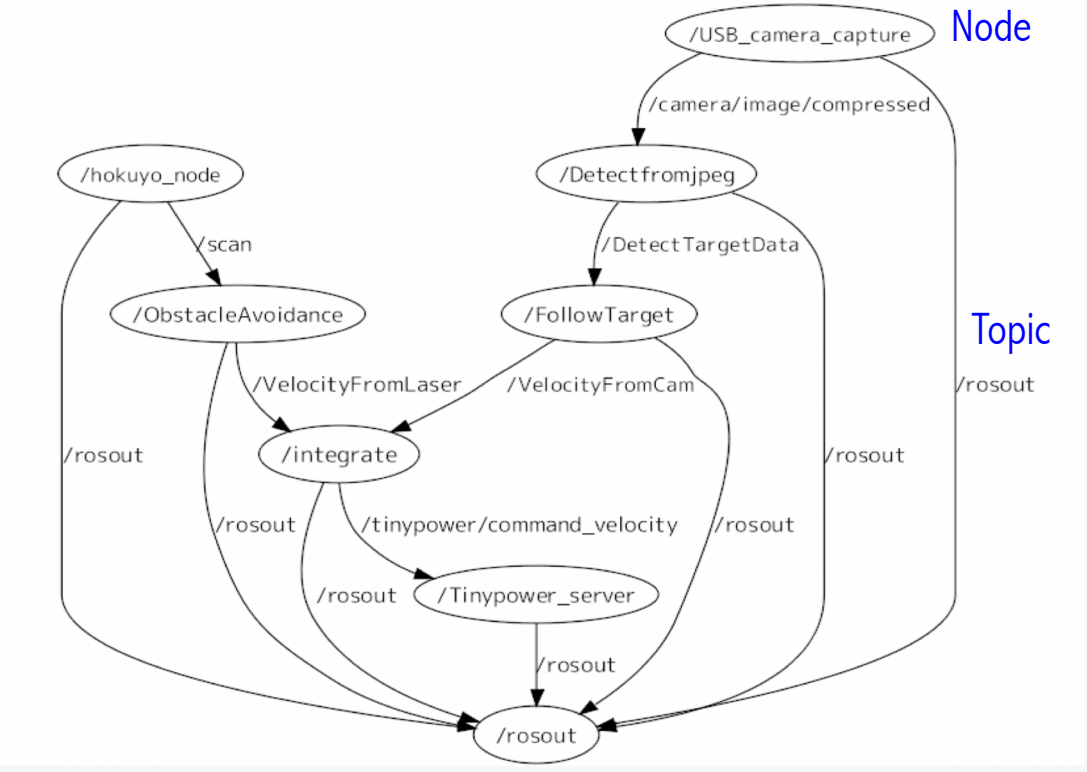
Was ist ROS (1)

- ROS = Robot Operating System (eigentlich Meta-Betriebssystem)
- Hardwareabstraktion und Gerätetreiber für zahlreiche Roboterplattformen, Sensoren und Aktoren



Was ist ROS (2)

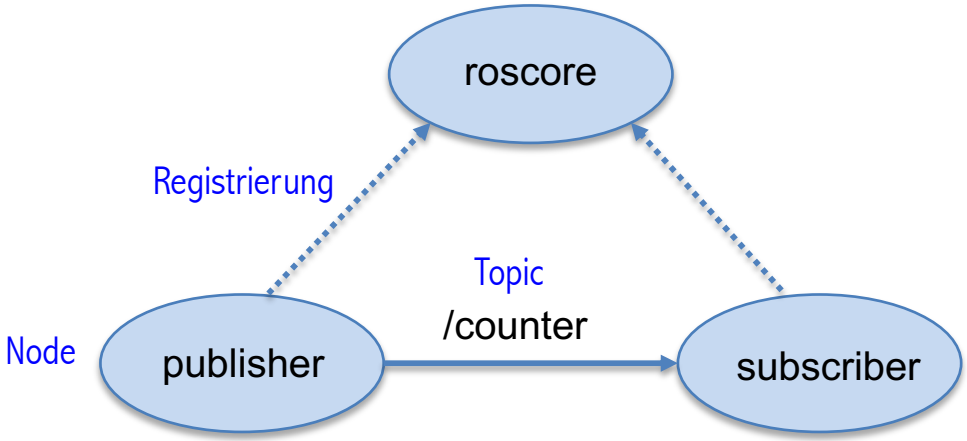
- Eine ROS-Anwendung besteht aus einer Menge von Nodes (Komponenten), die miteinander über Topics kommunizieren (ROS Graph)
- ROS-Anwendung kann auch verteilt laufen
- ROS bietet dazu Kommunikationsplattform: Publish/Subscribe, Service, ...



Was ist ROS (3)

- Werkzeug-basiert
  - Erstellung von Paketen und Navigation durch Paketstrukturen
  - Launch-Konzept: Start mehrerer Nodes
  - Visualisierung der Interprozesskommunikation
  - Logging-Mechanismus
  - Plotten und Visualisierung von Datenströmen
- Robotersimulatoren
- Zahlreiche Bibliotheken und Pakete
- Plattform: Linux (Ubuntu, Debian) und Windows 10
- Mehrsprachig: Python, C/C++, Java, ...
- Open Source
- Große Community

Publish/Subscribe Konzept




```
user@hostname$ roscore
user@hostname$ rosrun basics publisher.py
user@hostname$ rosrun basics subscriber.py
355
356
357
358
359
360
```

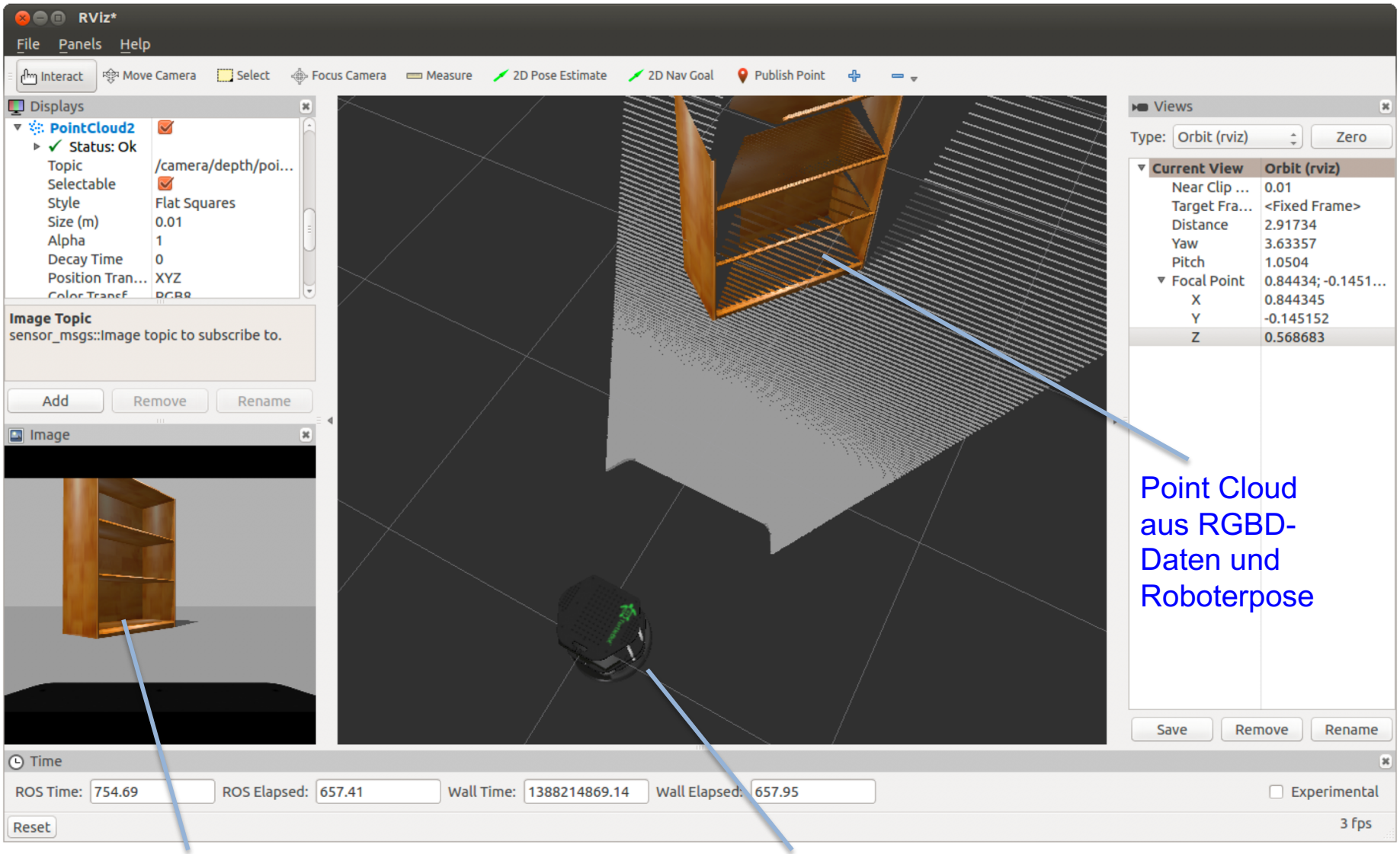
```
rospy.init_node('publisher')
pub = rospy.Publisher('counter', Int32)

rate = rospy.Rate(2)
count = 0

while not rospy.is_shutdown():
        pub.publish(count)
        count += 1
        rate.sleep()
```

publisher.py

```
def callback(msg):
        print msg.data

rospy.init_node('subscriber')
sub = rospy.Subscriber('counter', Int32, callback)

rospy.spin()
```

subscriber.py

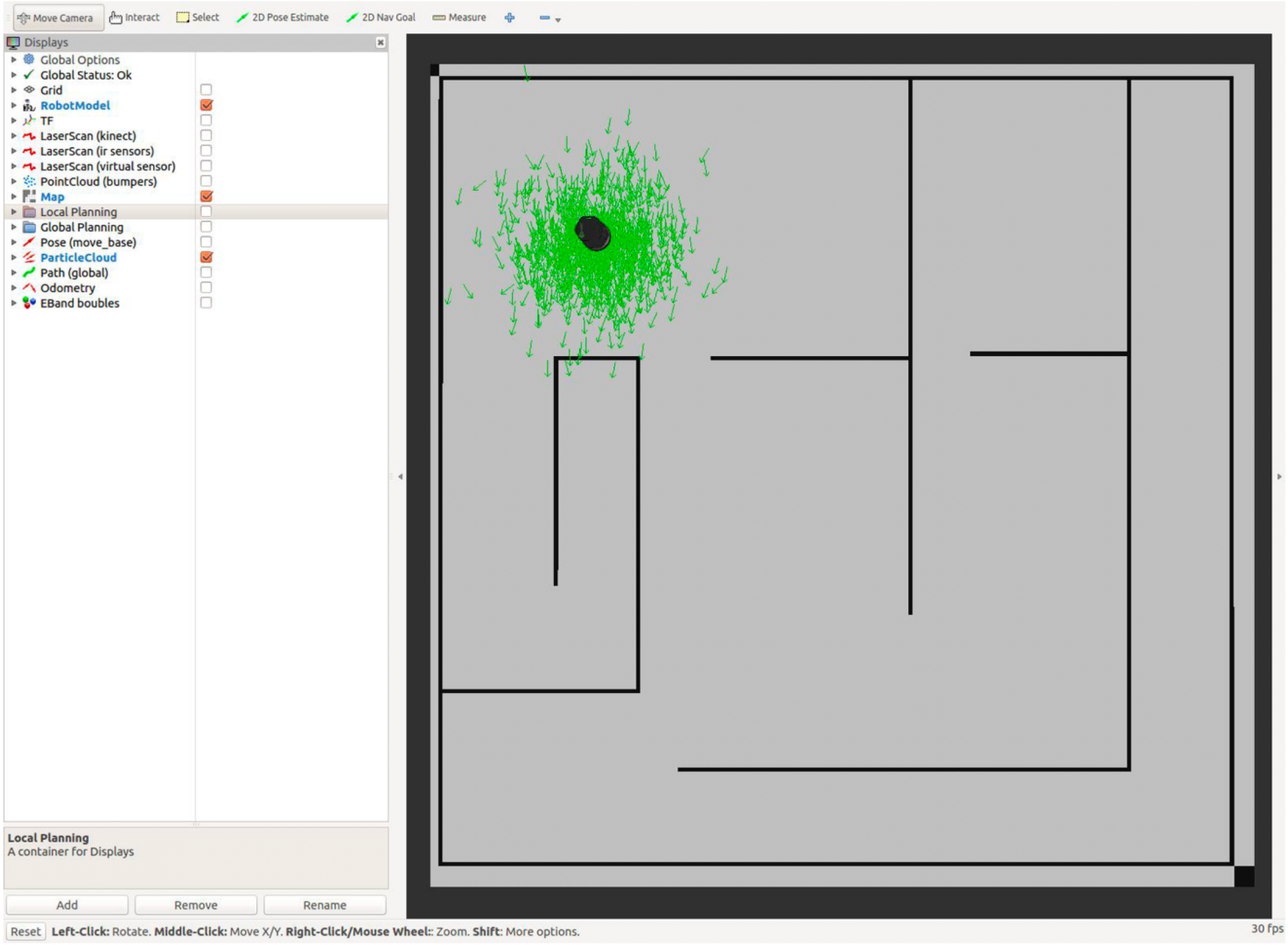
Visualisierung mit rviz (1)



Quickley u.a., Programming Robots with ROS, 2015

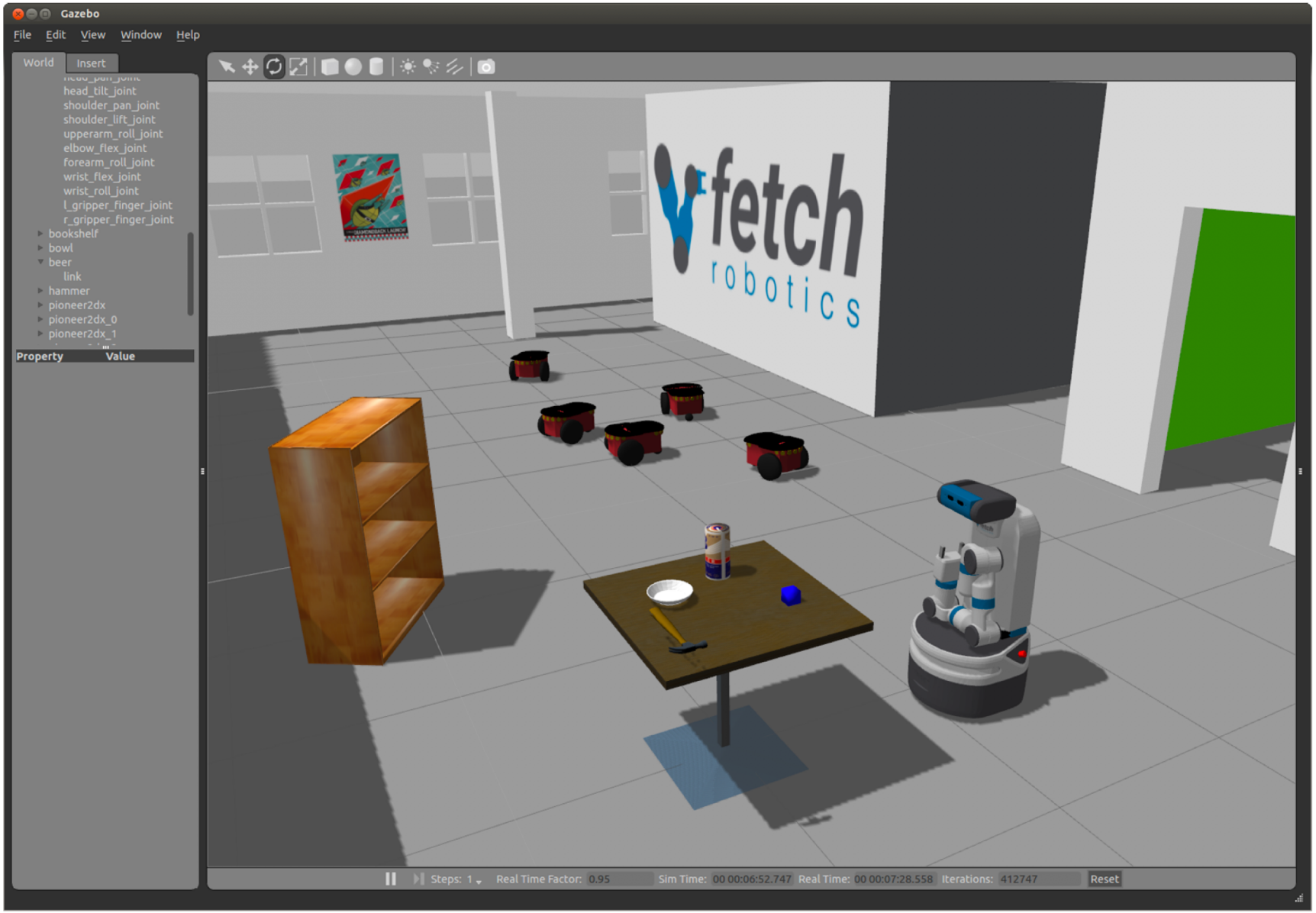
Visualisierung mit rviz (2)

Lokalisierung von Turtlebot mir einem Partikelfilter und einer bereitgestellten CAD-Karte.



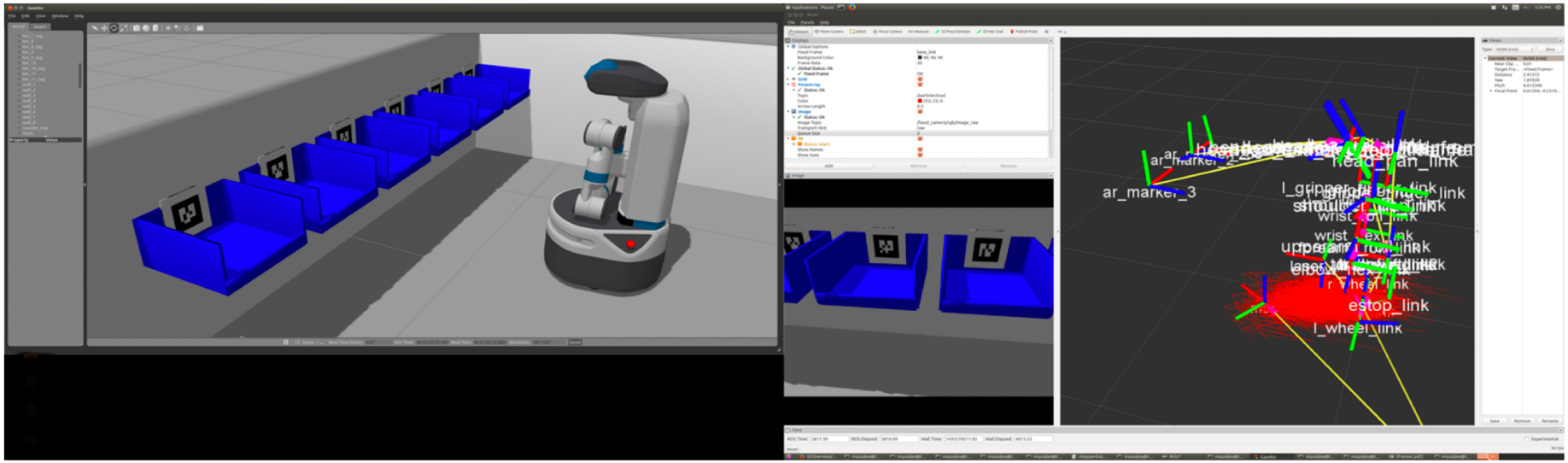
Quickley u.a., Programming Robots with ROS, 2015

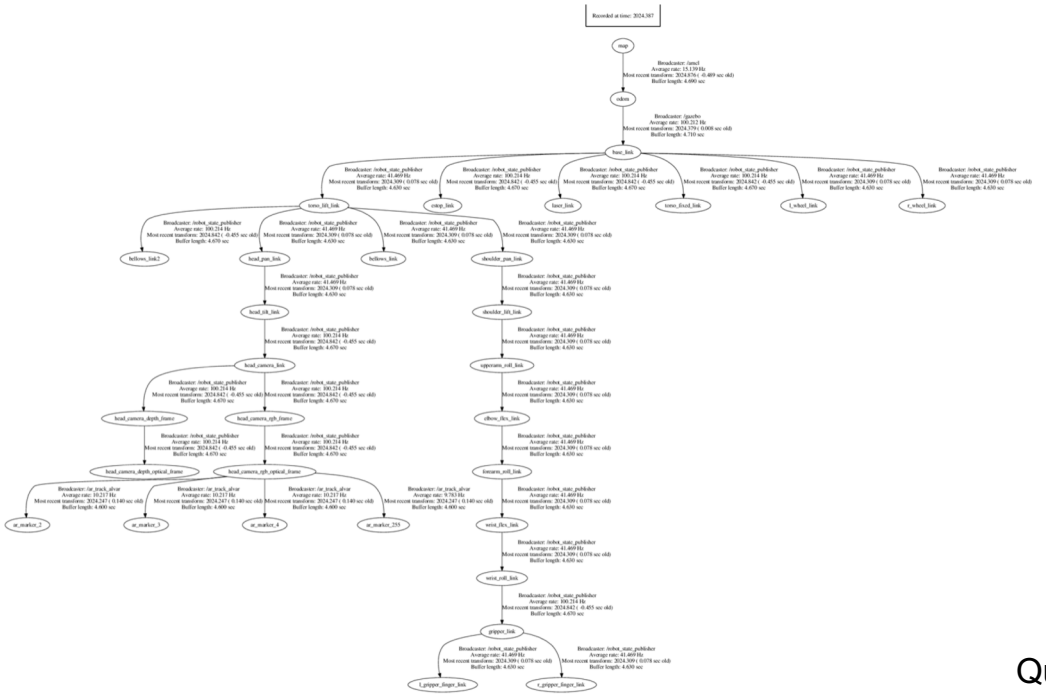
Simulation mit gazebo



Quickley u.a., Programming Robots with ROS, 2015

ROS tf: Verwaltung von KS-Transformationen





- Transfomationsbaum
- Knoten: KS
- Kante:
  Transformation mit Zeitstempel, Rate und Herausgeber (broadcaster)

Quickley u.a., Programming Robots with ROS, 2015

ROS tf: Beispiel

tf Listener

```
import rospy
import tf

listener = tf.TransformListener()

(trans,rot) = listener.lookupTransform('world', 'robot_base', rospy.Time(0)))

angles = tf.transformations.euler_from_quaternion(rot, 'sxyz')
```

- Hole von tf aktuellste Transformation von world nach robot_base
- Translation trans = (x,y,z)
- Rotation rot = (x,y,z,w) als Quaternion
- angles: Eulerwinkel im xyz-Drehsystem

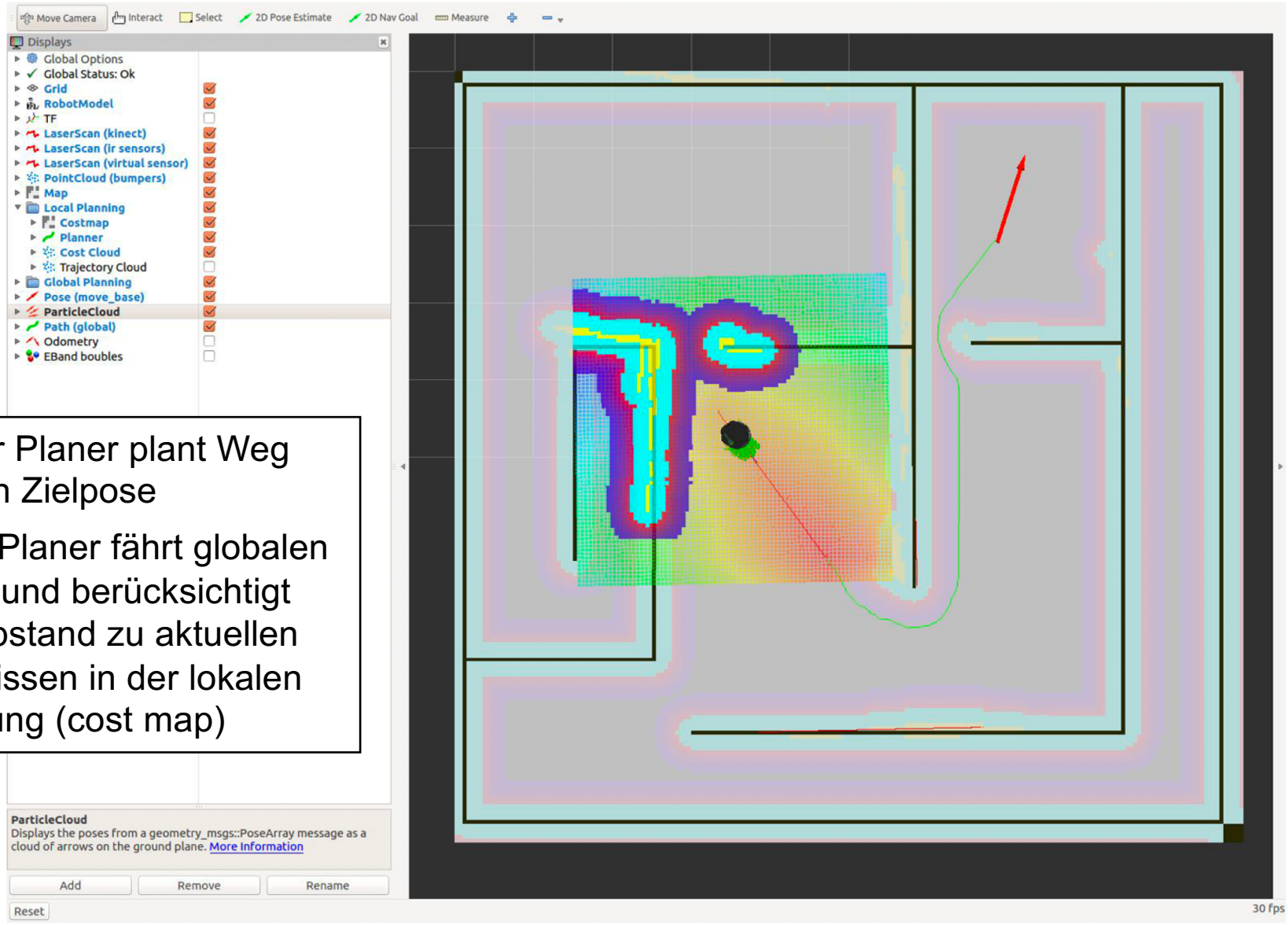
tf Broadcaster

```
import rospy
import tf

x = ...
y = ...
theta = ...

trans = (x, y, 0)
rot = tf.transformations.quaternion_from_euler(0, 0, theta, 'sxyz'),

broadcaster = tf.TransformBroadcaster()

broadcaster.sendTransform(trans, rot, rospy.Time.now(), 'robot_base', 'world')
```

- Veröffentliche auf tf Transformation von world nach robot_base

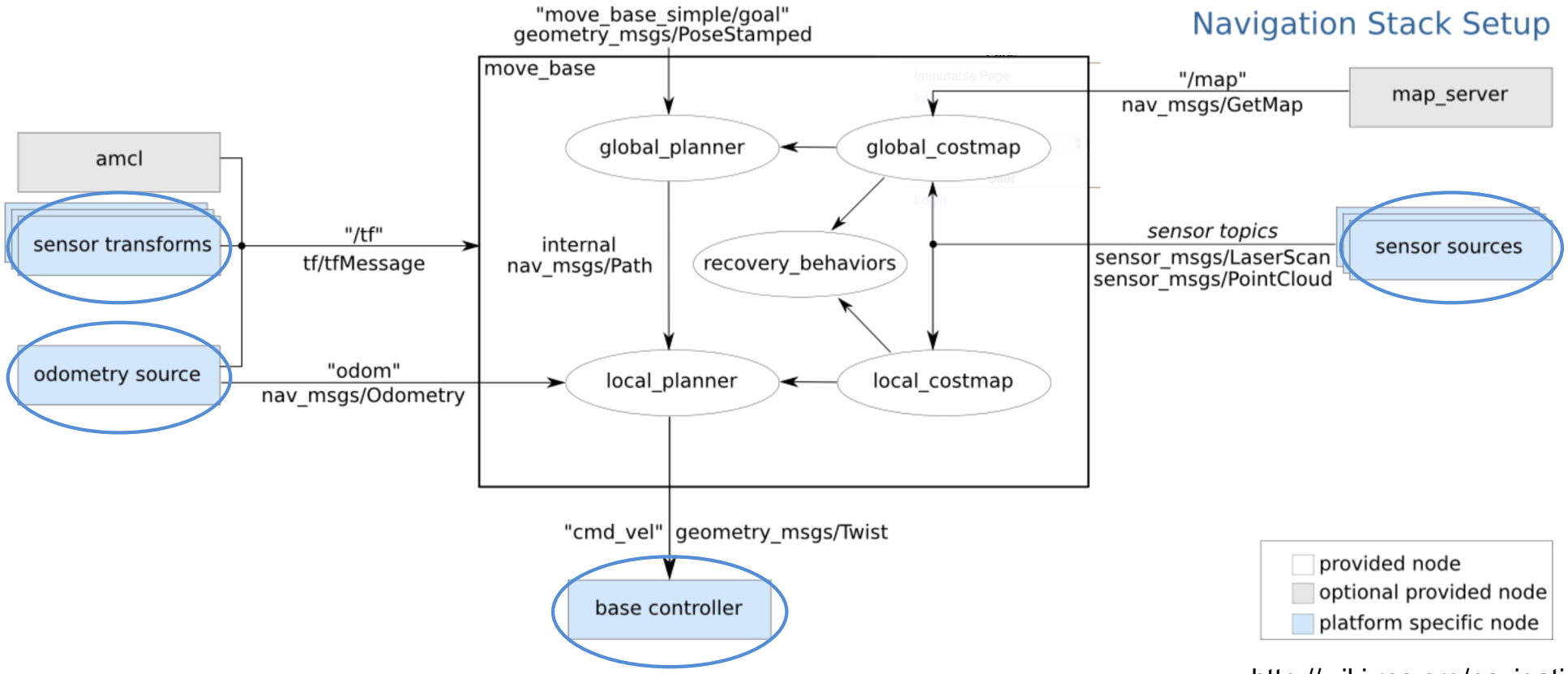
ROS Navigation Stack (1)

- Globaler Planer plant Weg zur roten Zielpose
- Lokaler Planer fährt globalen Plan ab und berücksichtigt dabei Abstand zu aktuellen Hindernissen in der lokalen Umgebung (cost map)



Quickley u.a., Programming Robots with ROS, 2015

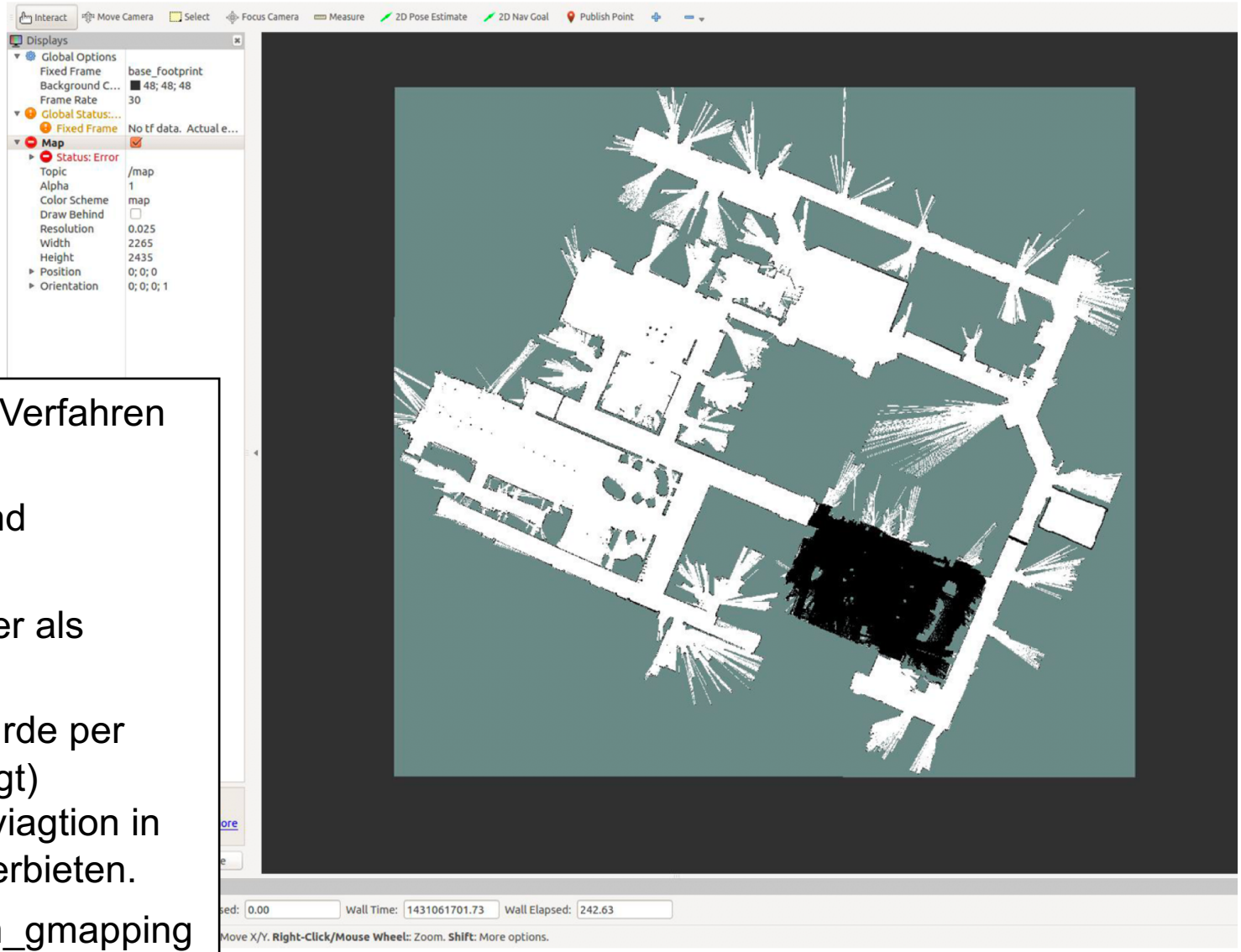
ROS Navigation Stack (2)



http://wiki.ros.org/navigation

- Wenige plattformspezifische Teile sind bereit zu stellen
- Roboter, der Geschwindigkeitsbefehle abonniert (cmd_vel)
- Roboter, der Odometriedaten veröffentlicht (odom + tf)
- Sensor-Pose in tf veröffentlichen
- Laser oder RGBD-Sensor, der Sensordaten veröffentlicht (sensor_topic)

Kartierung mit ROS gmapping

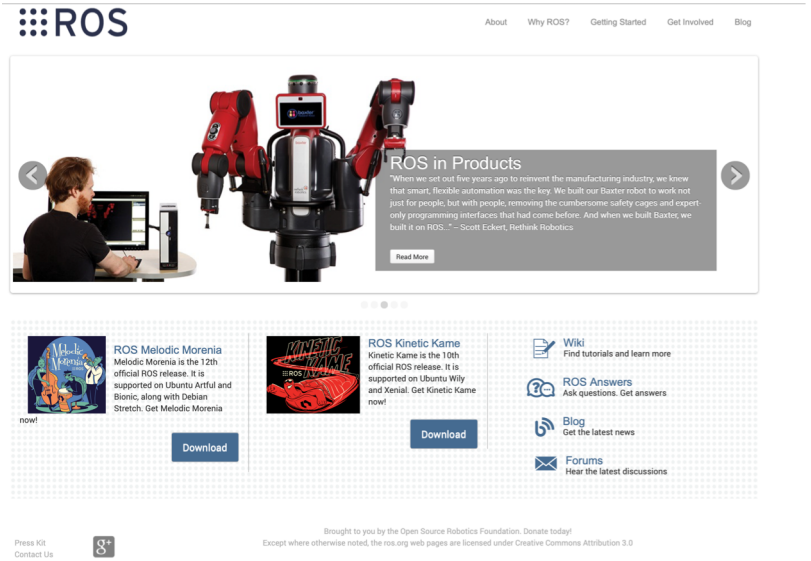


- Gitterbasiertes SLAM-Verfahren mit einem Partikelfilter
- Benötigt Odometrie und Laser-Scans
- Erstellt Belegtheitsgitter als png-Datei.
- Teil der Umgebung wurde per Hand schwarz (= belegt) übermalt, um eine Naviagtion in diesem Beereich zu verbieten.
- http://wiki.ros.org/slam_gmapping

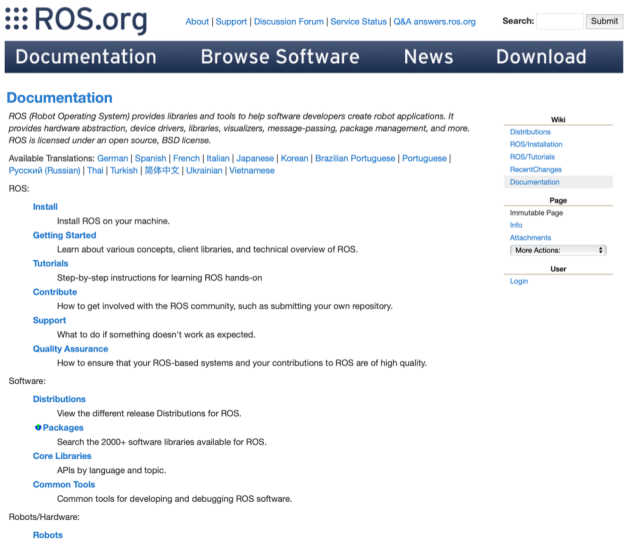
Quickley u.a., Programming Robots with ROS, 2015

Referenzen (1)

- ROS: https://www.ros.org

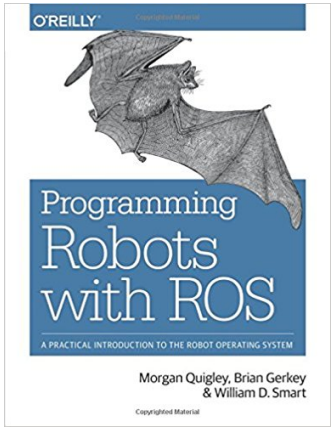


- ROS wiki: http://wiki.ros.org

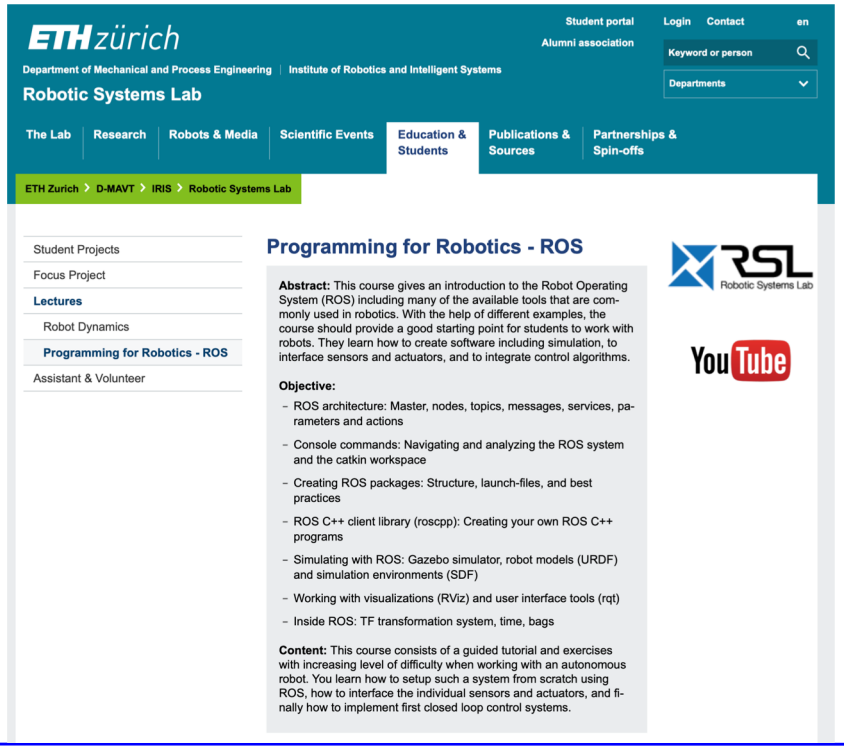


Referenzen (2)

- Morgan Quigley, Brian Gerkey, William Smart, *Programming Robots with ROS,* O'Reilly, 2015
- als E-Book im Rahmen der Vorlesung ausleihbar
- Sehr gute Einführung in ROS
- Beispiele mit Python



- Sehr guter ROS-Kurs in C/C++ (mit Videos, Folien und Übungen) https://rsl.ethz.ch/education-students/lectures/ros.html



Referenzen (3)

- „ROS-Industrial is an open-source project that extends the advanced capabilities of ROS software to manufacturing.“
- https://rosindustrial.org/