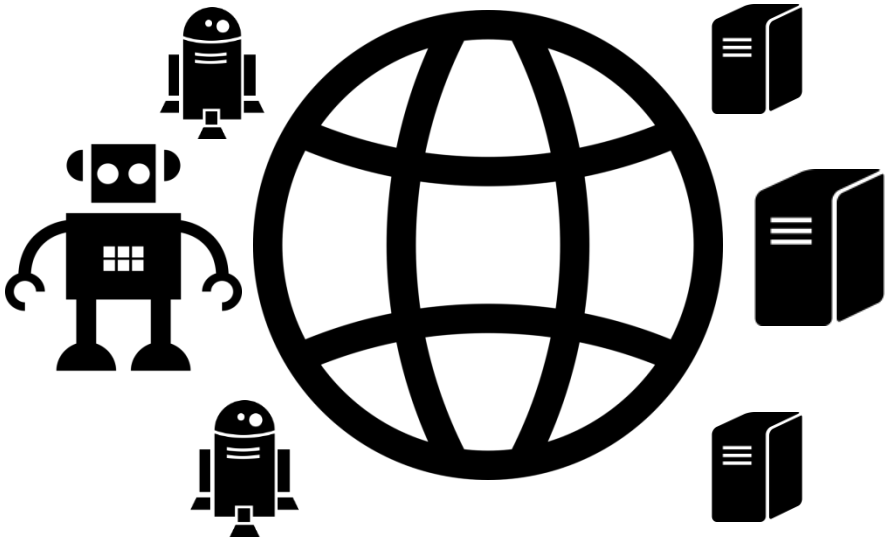
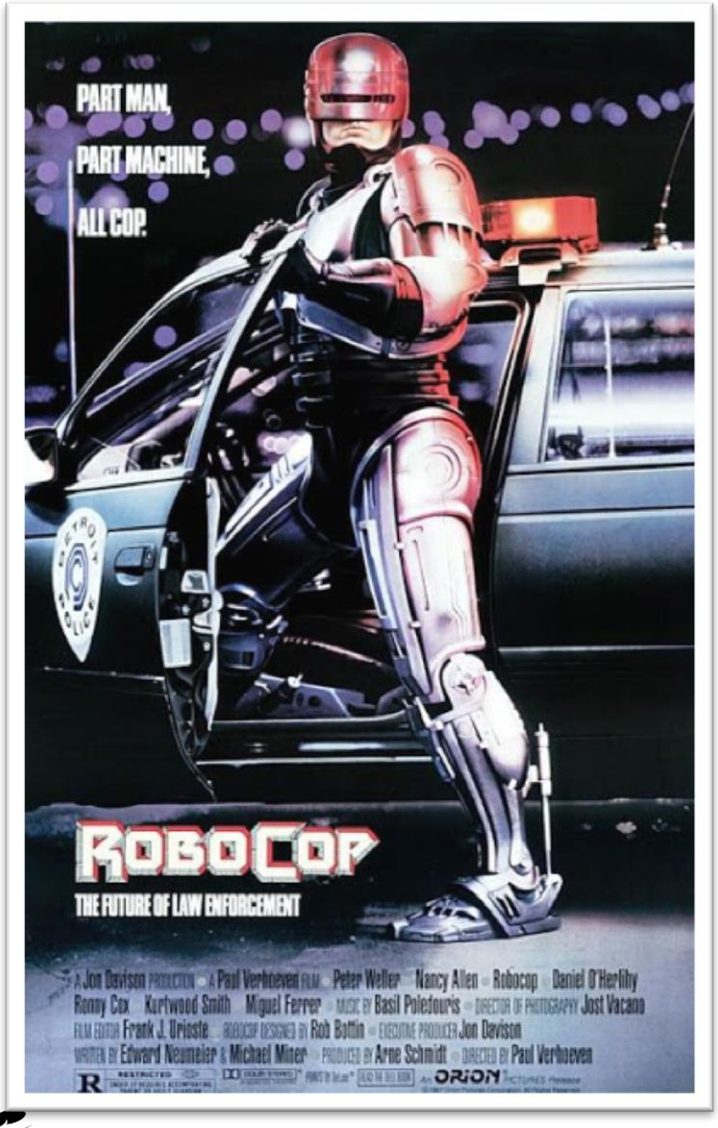
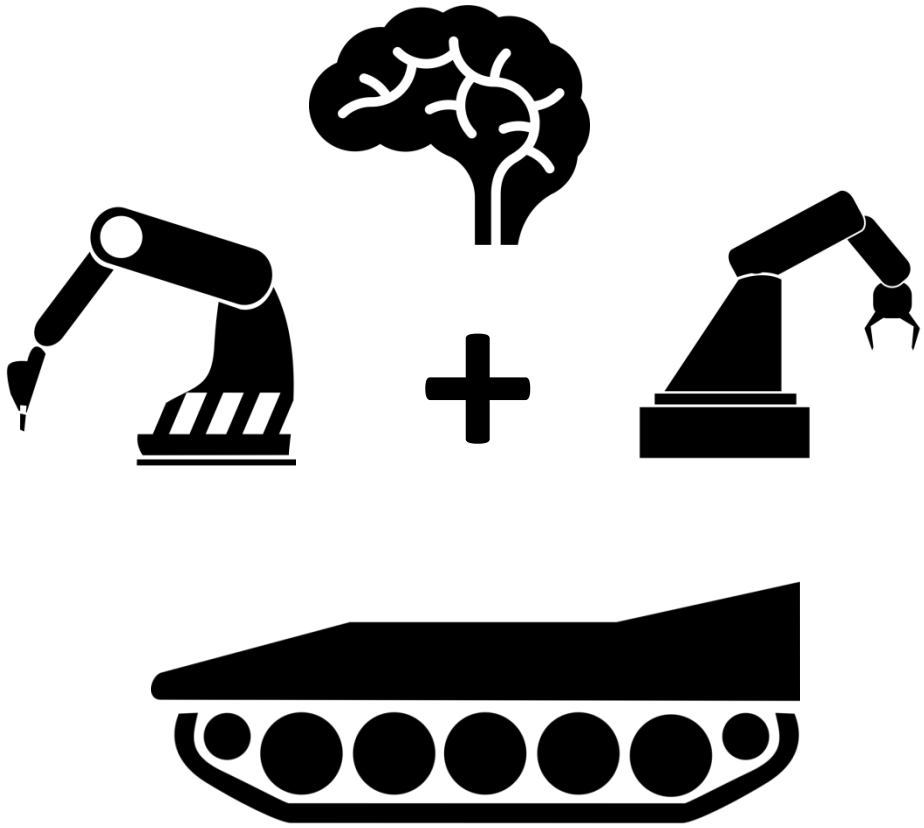
Robotic Middlewares

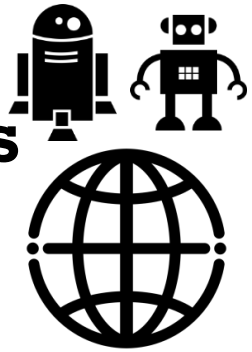


What is Robotic today ?

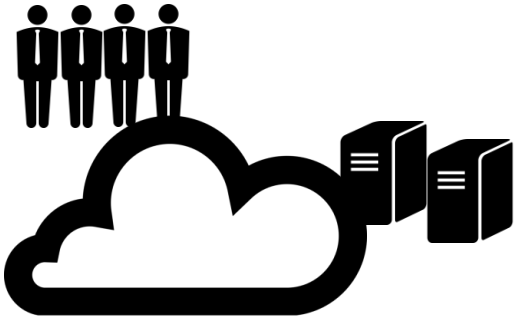




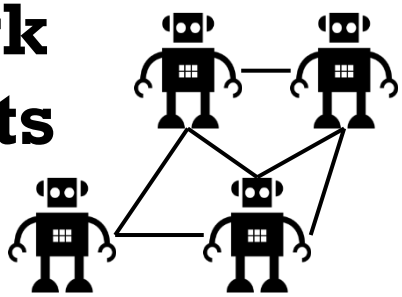
Internet for robots



Crowd sourcing Cloud Computing

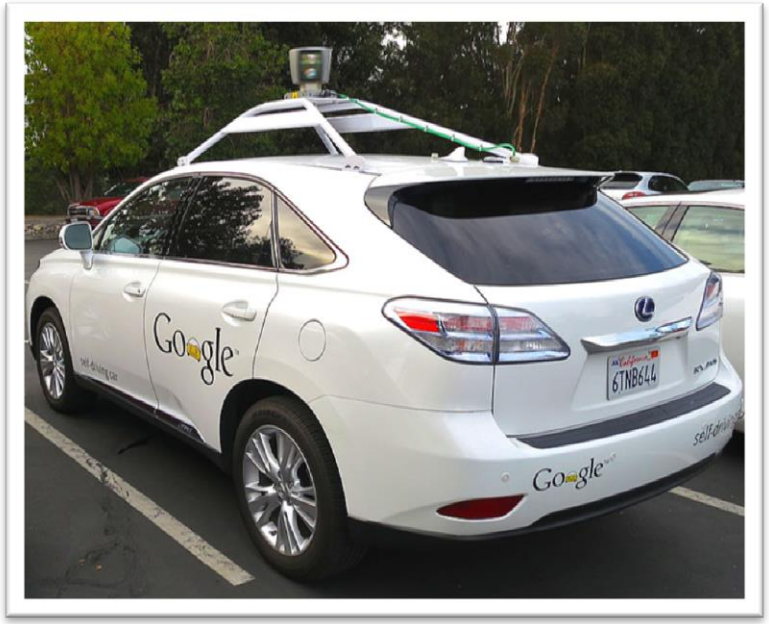


Network of robots



Distributed Intelligence

Industrial VS Service



Industrial VS Service

- Known environment
- High speed design
- Intelligence close to the hardware
- Limited robot cooperation
- Limited (forbidden ?) human interactions

- Unknown environment
- Need to deal with uncertain
- Higher level of intelligence
- Strong human interactions
- Possibilité of robot to robot, human to robot cooperation

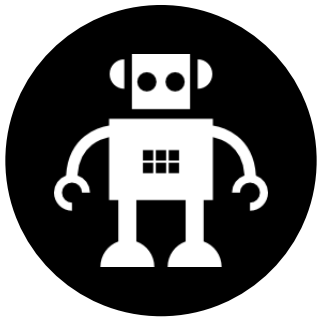
CPE LYON
ÉCOLE SUPÉRIEURE DE CHIMIE PHYSIQUE ÉLECTRONIQUE DE LYON

Current Robotic Needs

- ❑ Specific implementation
- ❑ Heterogeneous communication
- ❑ Centralized communication
- ❑ Re-Inventing the Wheel Syndrom

Develop Robotic Framework

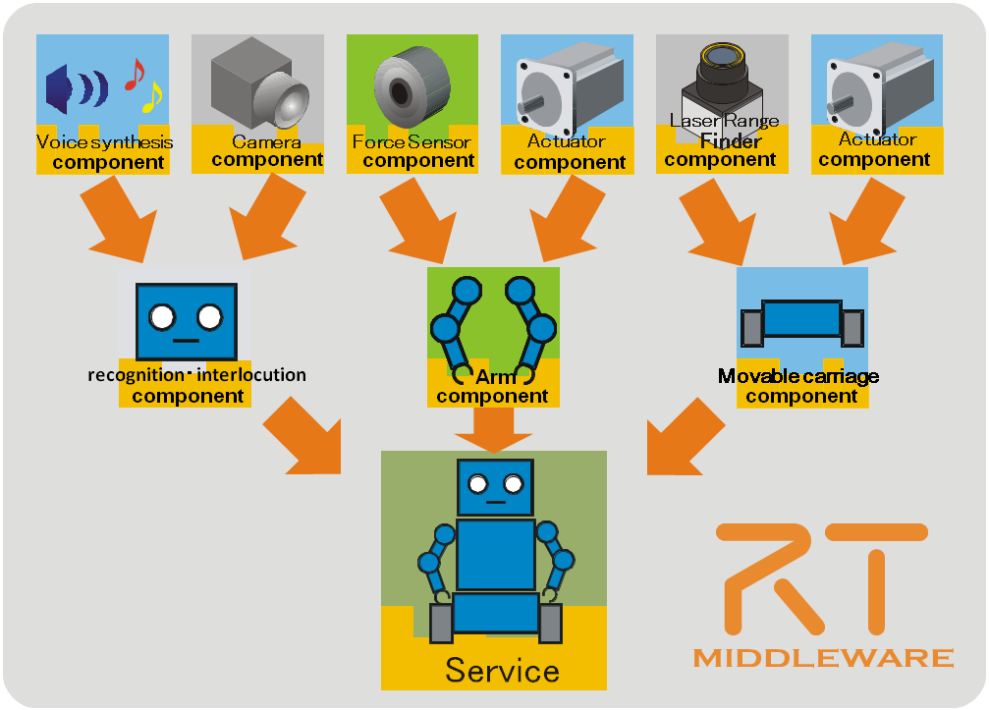
- ❑ Naoqi
- ❑ Urbi
- ❑ RT-Middleware
- ❑ ROS



RT Middleware: Robot Technology Middleware

What is RT Middleware ?

- A Robotic Middleware
- A common plateform to assist development of robots
- A Modular Design
- OpenRTM-aist implementation based on Corba
- Under LGPL License



Who provides RT-MiddleWare ?

RT MIDDLEWARE

Develop an implementation of RT-Middleware called OpenRTM-aist

Defines / discusses RT-component specification/ standard

CPE LYON
ÉCOLE SUPÉRIEURE DE CHIMIE PHYSIQUE ÉLECTRONIQUE DE LYON

Who uses RT-MiddleWare ?

VSTONE

Segway Blackship

Mobilerobots PeopleBot

Choromet 2

HRP-2

Kanto Auto Works, Patrafour

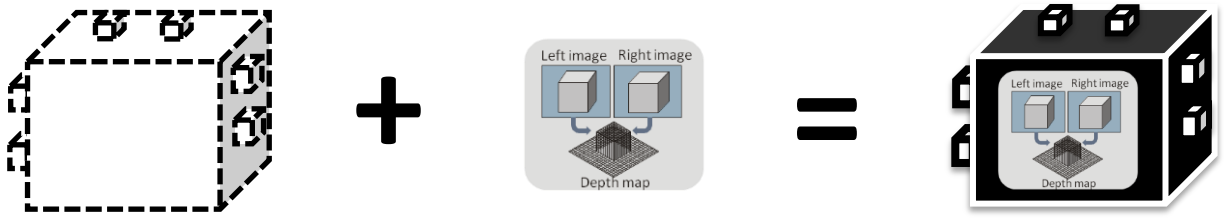
Segway RMP

G-ROBORS GR001

Who uses RT-MiddleWare ?

Kawada Industry Humanoid Robot - HIRO

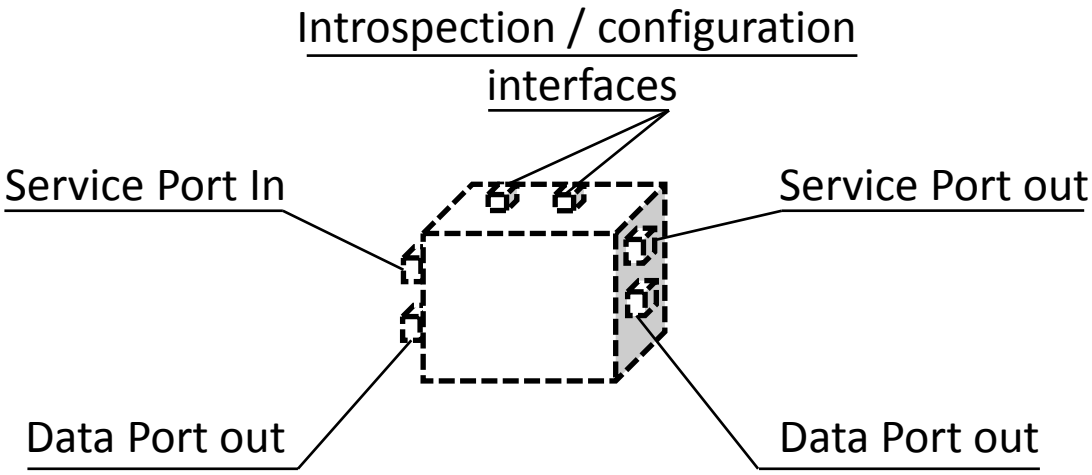
How does it work ?



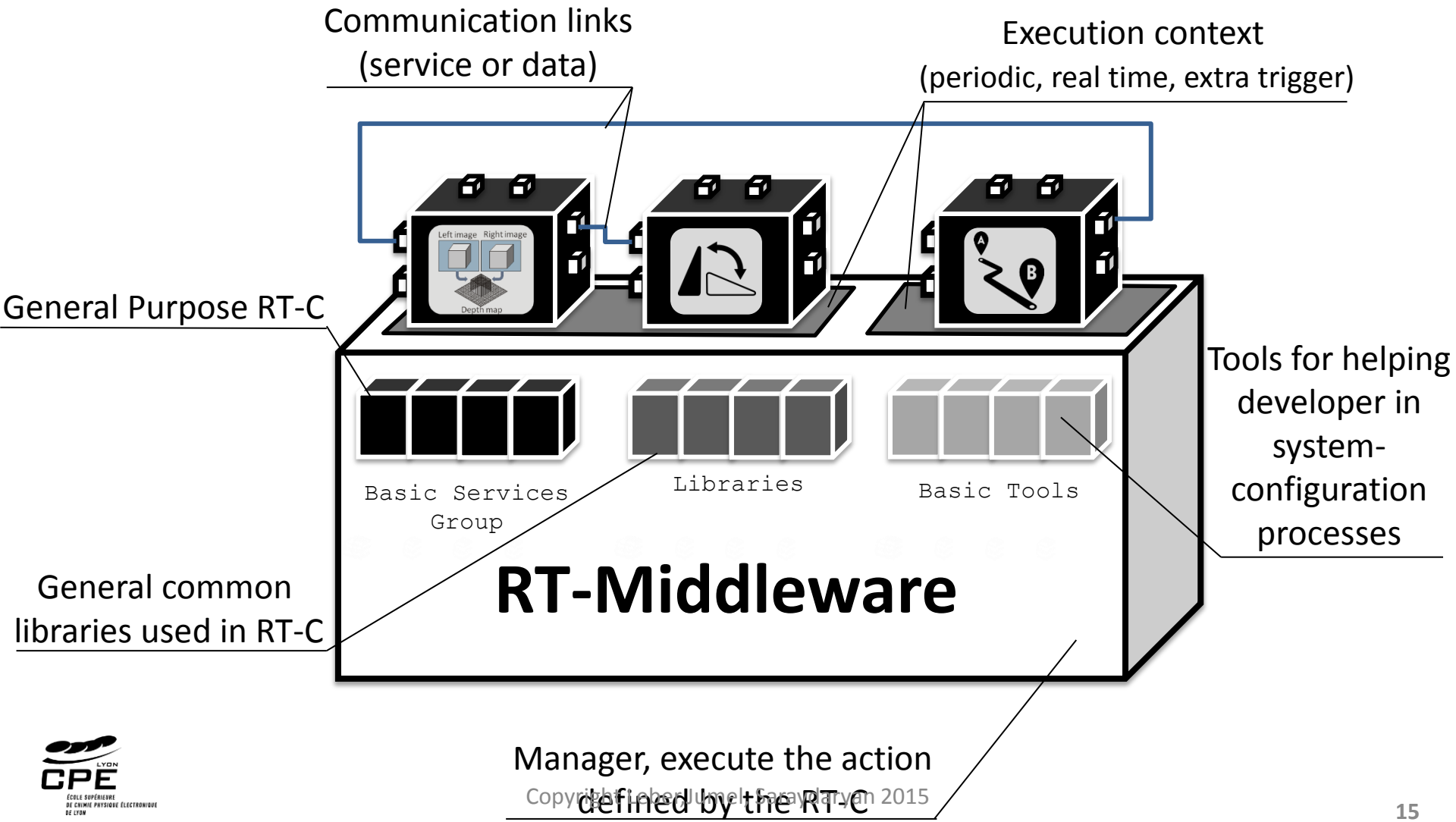
RT-Component Framework

Core Logic

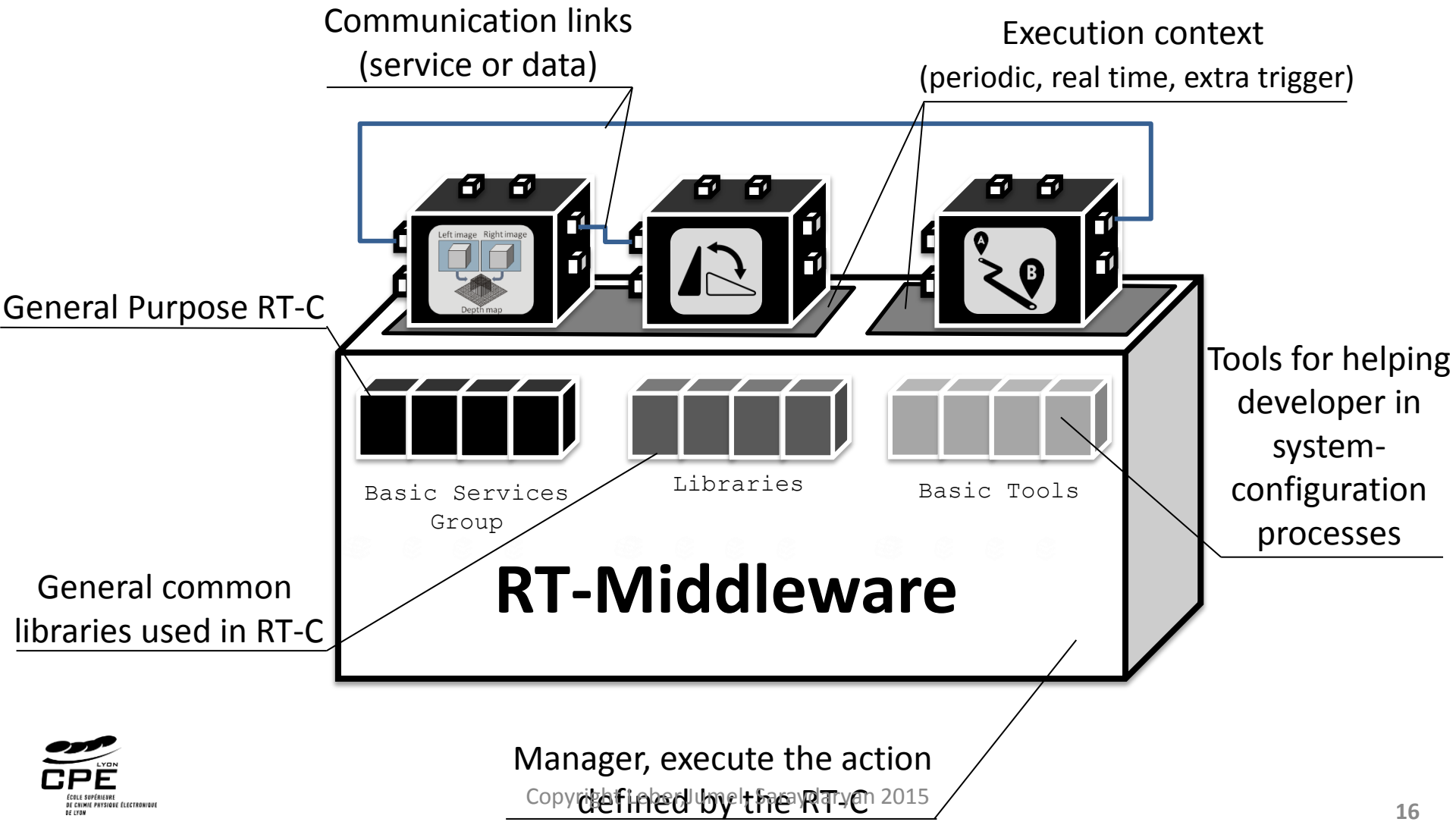
RT-Component



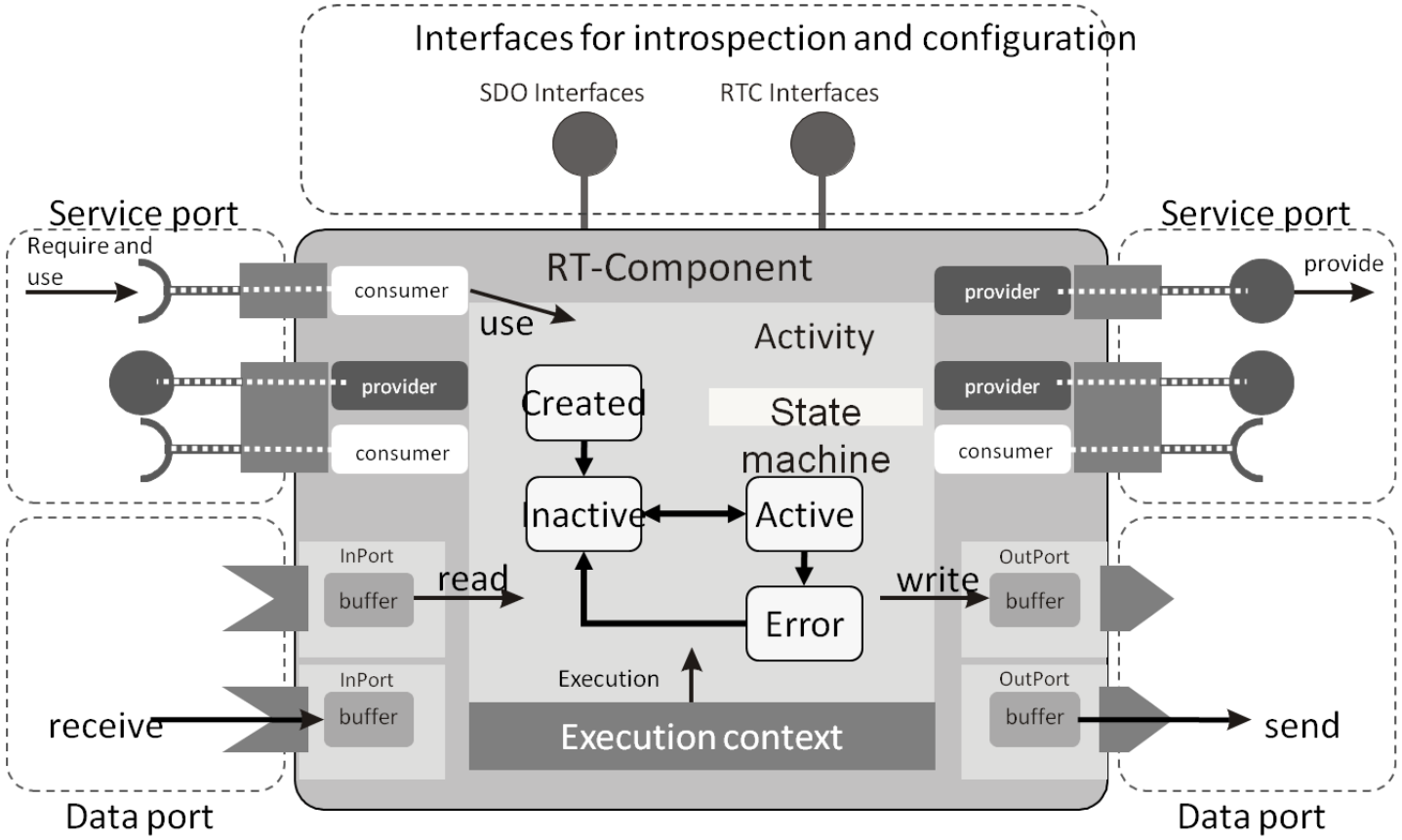
How does it work ?



How does it work ?



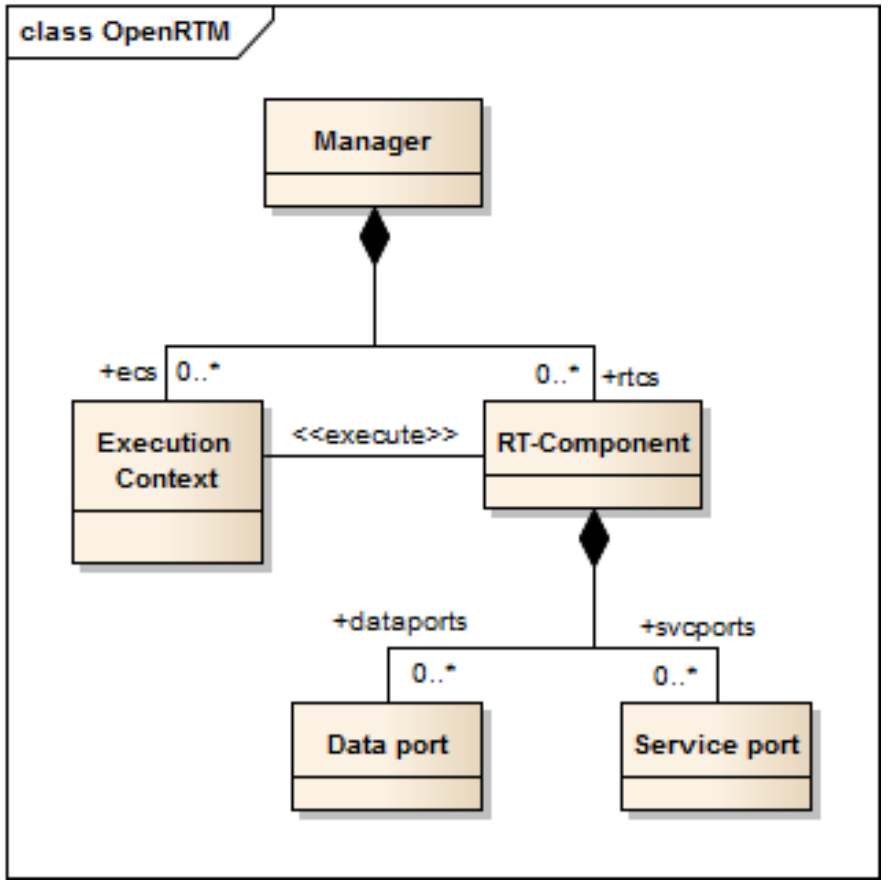
How does it work ?

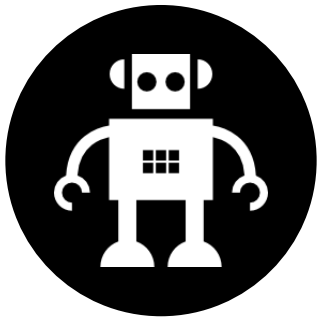


http://www.openrtm.org/openrtm/sites/default/files/440/RTCArchitecture_en.png

How To develop ?

1. Define the specification of a component.
2. Generate the templates of the component by using a cord generator.
3. Integrate libraries etc. you have in a specific location in the template.
4. Compile, test and debug.
5. Integrate into a system.

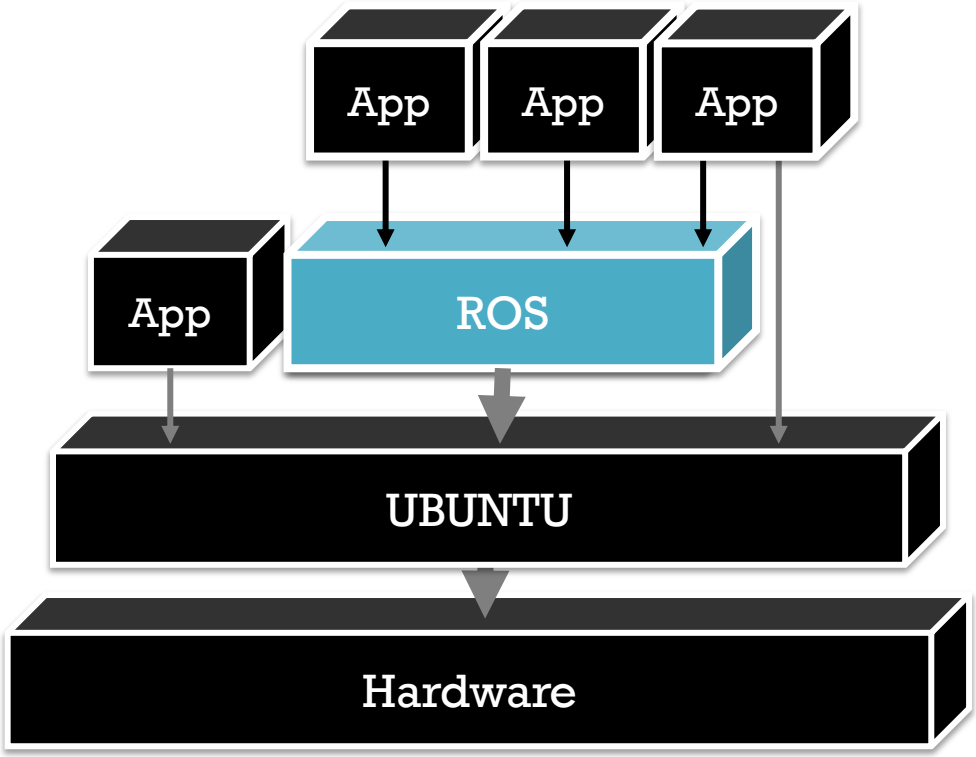




ROS:
Robot Operating System

What is ROS ?

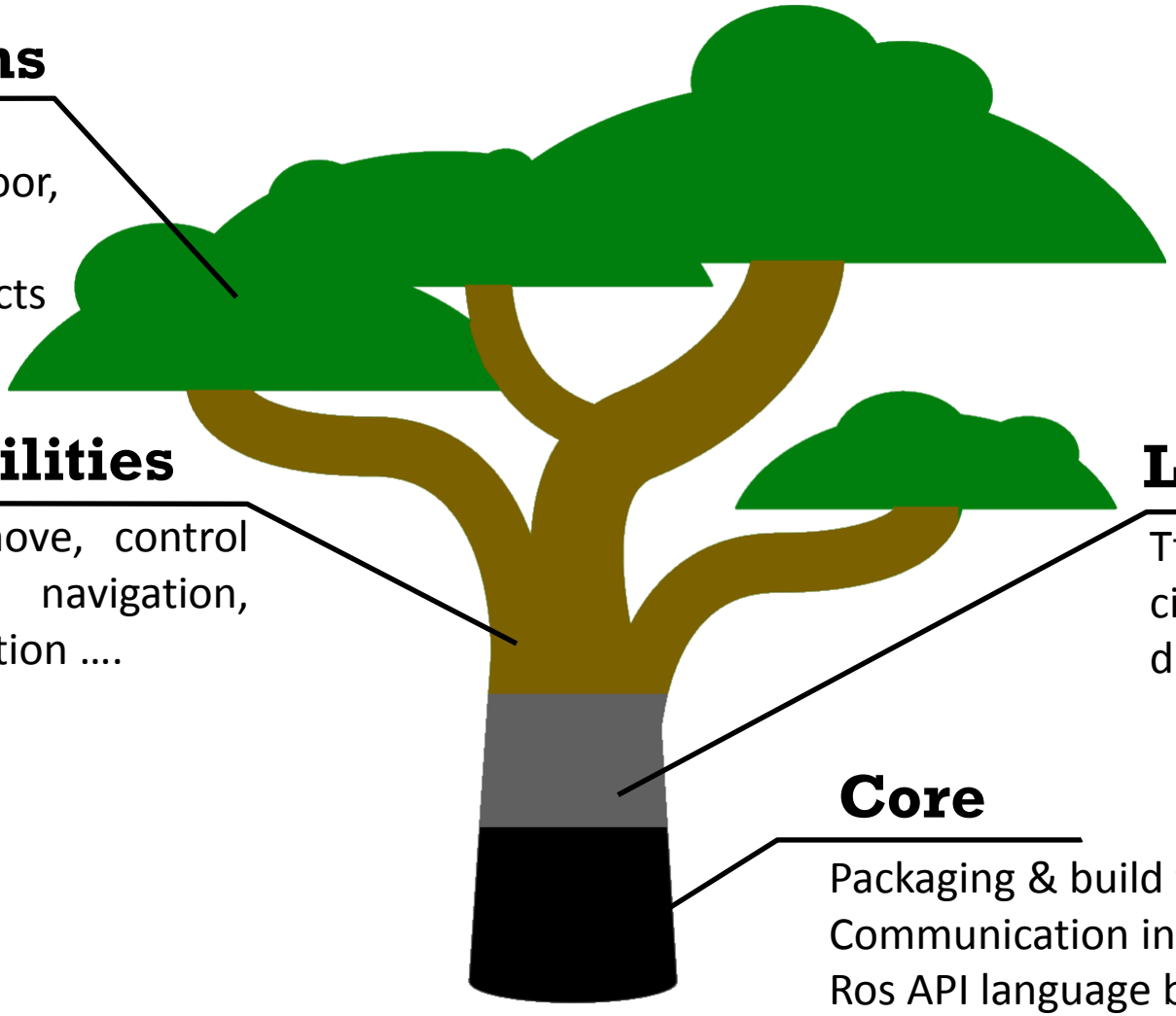
- A Robotic Middleware
- A set of libraries
- A distributed Modular Design
- Free and OpenSource



What does ROS Provide?

Applications

Fetching beer,
scraping the seafloor,
play football,
manufacture objects



Capabilities

Grasping, move, control execution, navigation, object detection ….

Libraries

Tf, opencv, pcl, kdl, cisst, simulation, drivers …

Core

Packaging & build tools,
Communication infra
Ros API language binding
Introspection tools …

Who provides ROS ?

An Active Community

A great history

May 1, 2007 — September 1, 2007

Switchyard at Stanford

Before the ideas fully coalesced to become ROS, several robotics software frameworks were prototyped in research projects at Stanford, including the STanford Artificial Intelligence Robot (STAIR) and the Personal Robotics (PR) program.

NOVEMBER 1, 2007
ROS at Willow Garage

Who uses ROS ?

Fraunhofer IPA
Care-O-bot

Videre Erratic

TurtleBot

Aldebaran Nao

Lego NXT

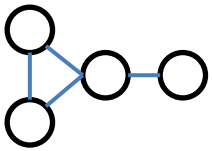
Shadow
Hand

Willow Garage PR2

iRobot
Roomba

Robotnik
Guardian

How does it work ?



Set of Nodes

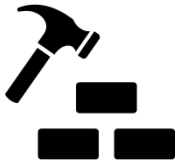
- Application
- Services
- Modules

Name Service

- Node registration
- Node Directory
- Communication orchestrator

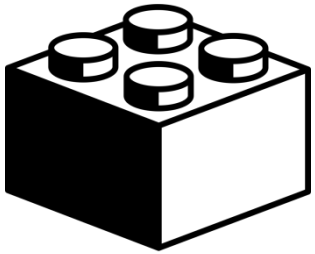
Communication

- Asynchrone (topics)
- Synchrone (services)



Build Tools

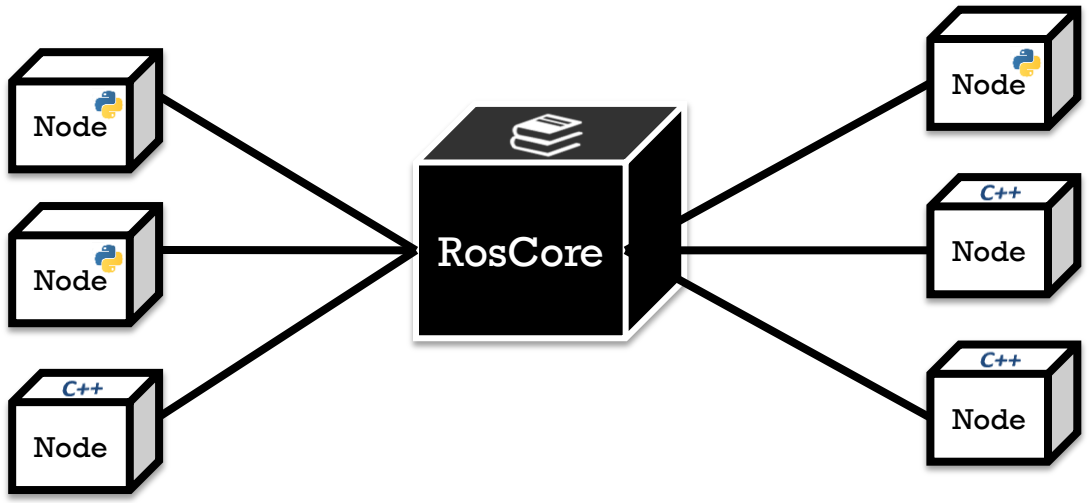
- Complilation,
- Packaging
- Dependency management



ROS: The Basics

ROS: Node

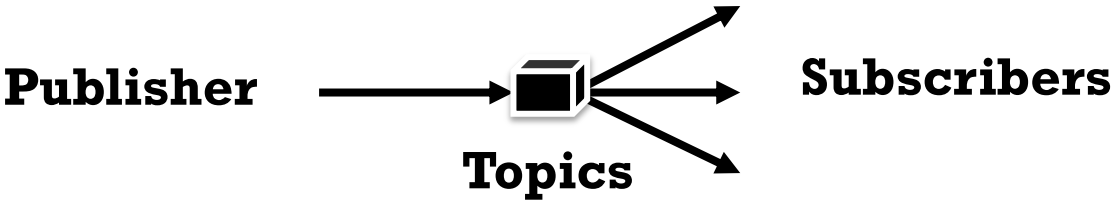
- Process that uses ROS framework
- Can be Remote
- Are known via Roscore
- Can be written into different languages : Python, C++



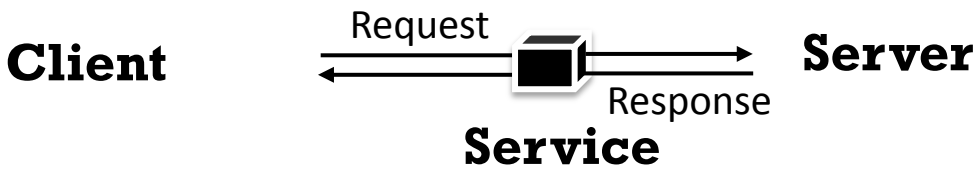
- Ros Core acts primarily as a name serveur
- Provide communication architecture
- Hold global parameters

ROS: Communication

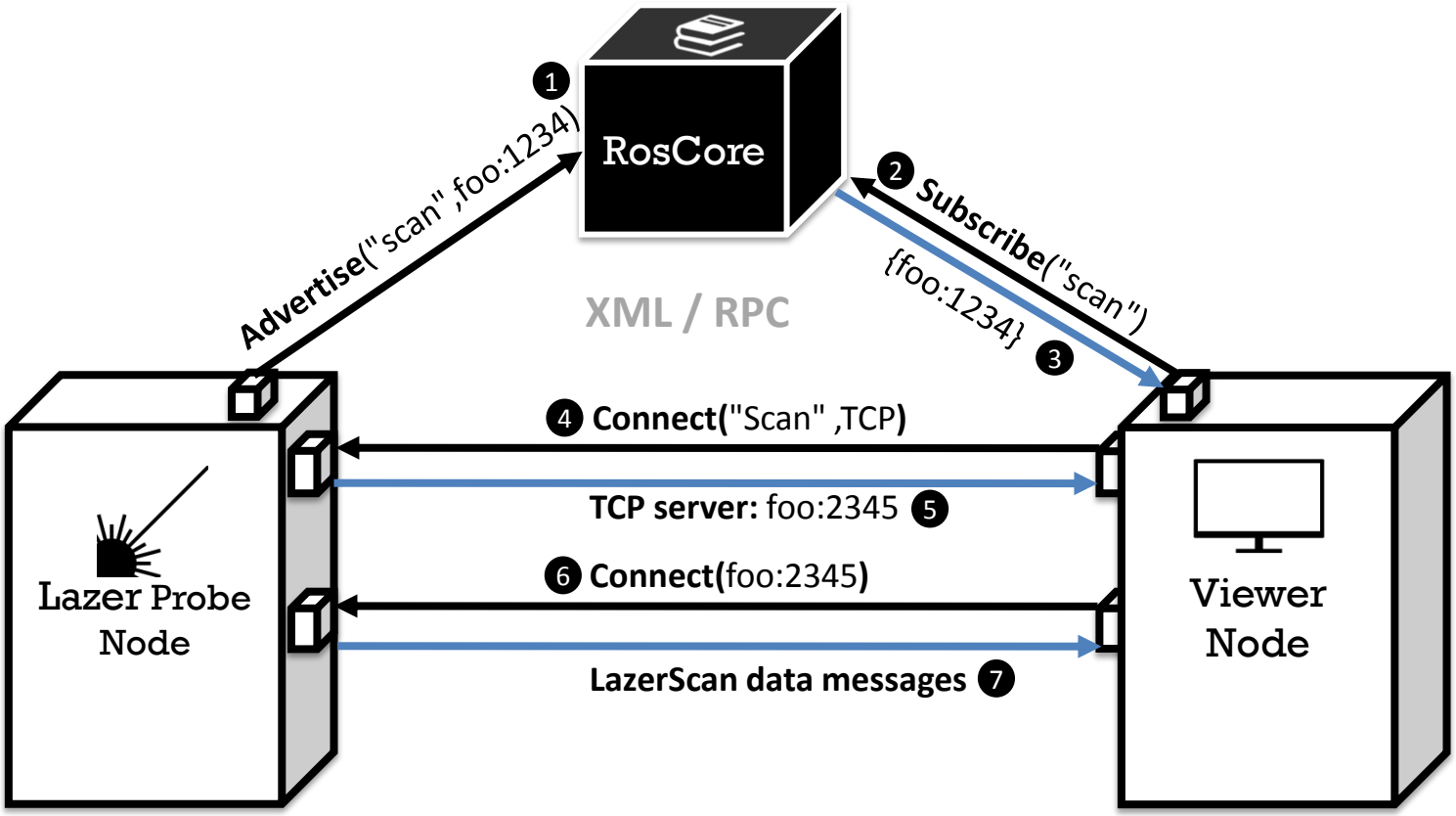
- Topics
  - Asynchronous
  - Many to Many
  - Publish/subscribe anonymous message passing



- Services
  - Synchronous
  - RPC request /reply interaction



ROS: Communication



ROS: Command line

`roscore`	Start the ros Master
`rosrun`	Start a ros node
`roslaunch`	Start a set of ros node
`rostopic`	Interact with ros topic
`rosservice`	Interact with ros service
`rosparam`	Getting and setting ros parameters
`rosbag`	Register/replay ros topic
`rosnode`	Display information about ros node
`rosmsg`	Display message data structure

…

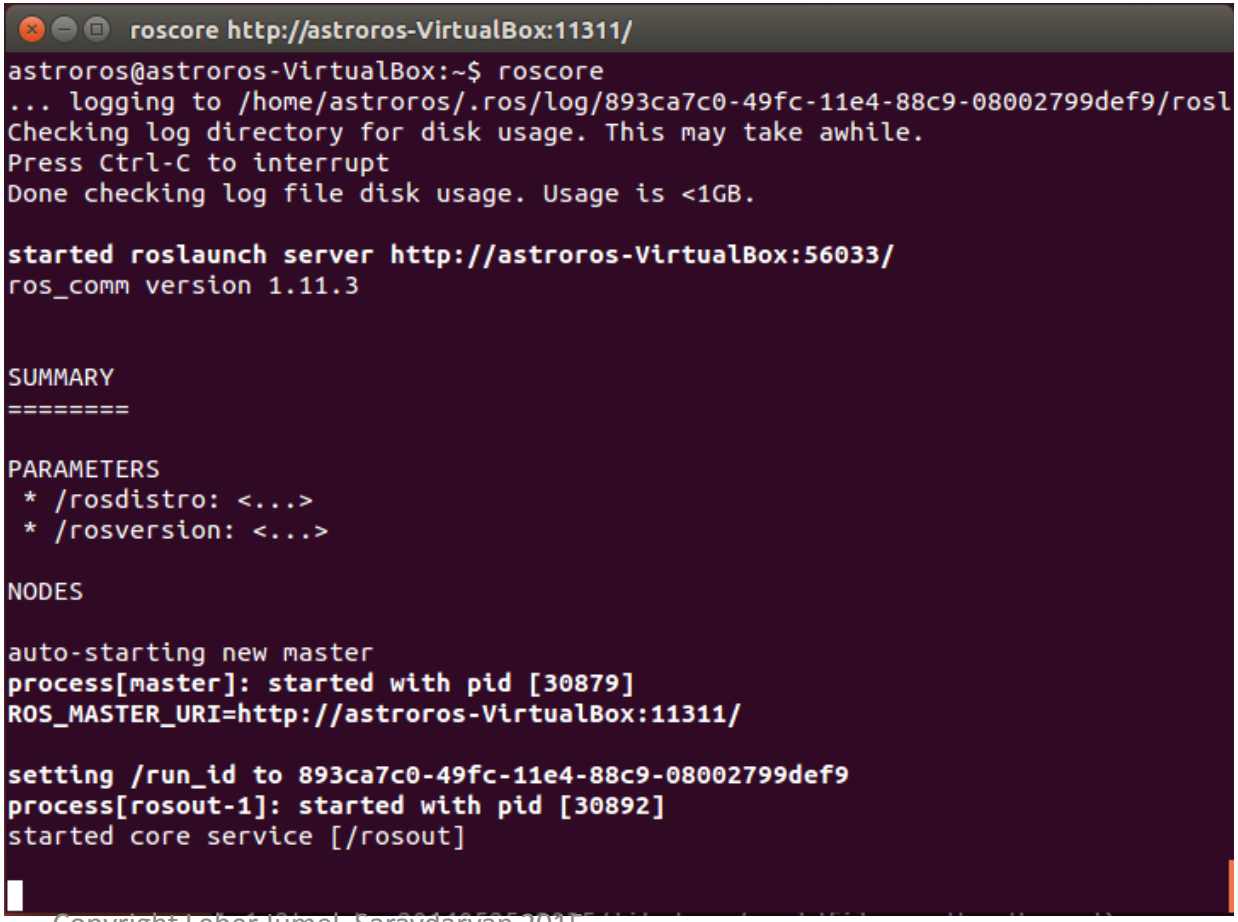
CPE LYON
ÉCOLE SUPÉRIEURE DE CHIMIE PHYSIQUE ÉLECTRONIQUE DE LYON

ROS: Command line

roscore

- Dfn:
  Start the ros Master
- Usage

```
roscore
roscore –p 4523
```



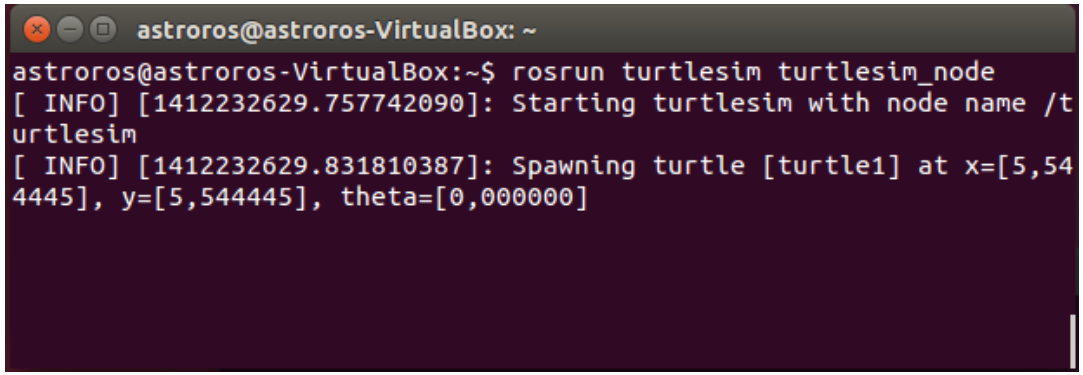
ROS: Command line

rosrun

- Dfn:
  Start the a ros node

- Usage

```
rosrun <package> <executable>
rosrun vision facedetection
```

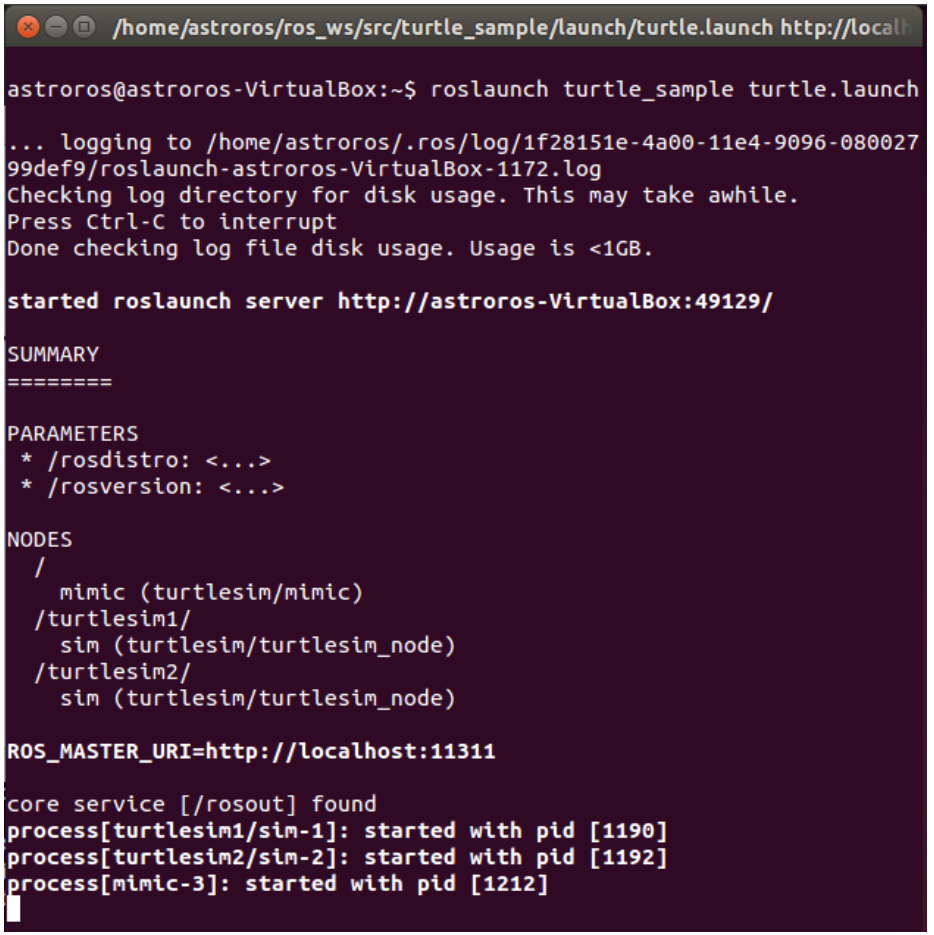


ROS: Command line

roslaunch

- Dfn:
  Start the multiple ros node

- Usage
  `roslaunch <package> <file.launch>`
  `rosrun vision facedetection`



ROS: Command line

roslaunch

- Dfn:
  Start the multiple ros node

- Usage

```
roslaunch <package> <file.launch>
rosrun turtle_sample turtle.launch
```

turtle.launch

```
<launch>

    <node pkg="turtlesim" name="operator" type="turtle_teleop_key">
         <remap from="/turtle1/cmd_vel"
                to="/turtlesim1/turtle1/cmd_vel"/>
    </node>

    <group ns="turtlesim1">
        <node pkg="turtlesim" name="sim" type="turtlesim_node"/>
    </group>

    <group ns="turtlesim2">
        <node pkg="turtlesim" name="sim" type="turtlesim_node"/>
    </group>

    <node pkg="turtlesim" name="mimic" type="mimic">
       <remap from="input" to="turtlesim1/turtle1"/>
       <remap from="output" to="turtlesim2/turtle1"/>
    </node>

</launch>
```

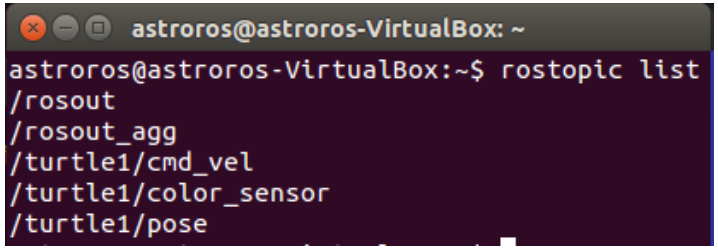
ROS: Command line

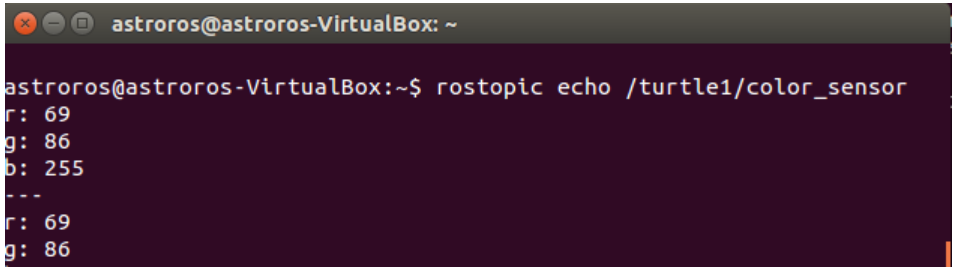
rostopic

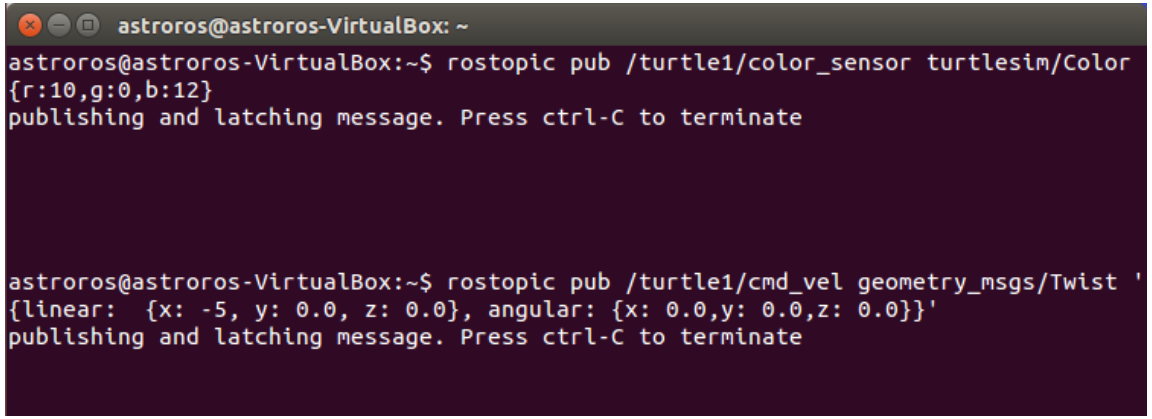
- Dfn:
  Interact with published topics

- Usage

```
roslaunch list
rostopic echo
rostopic pub
rostopic type
```

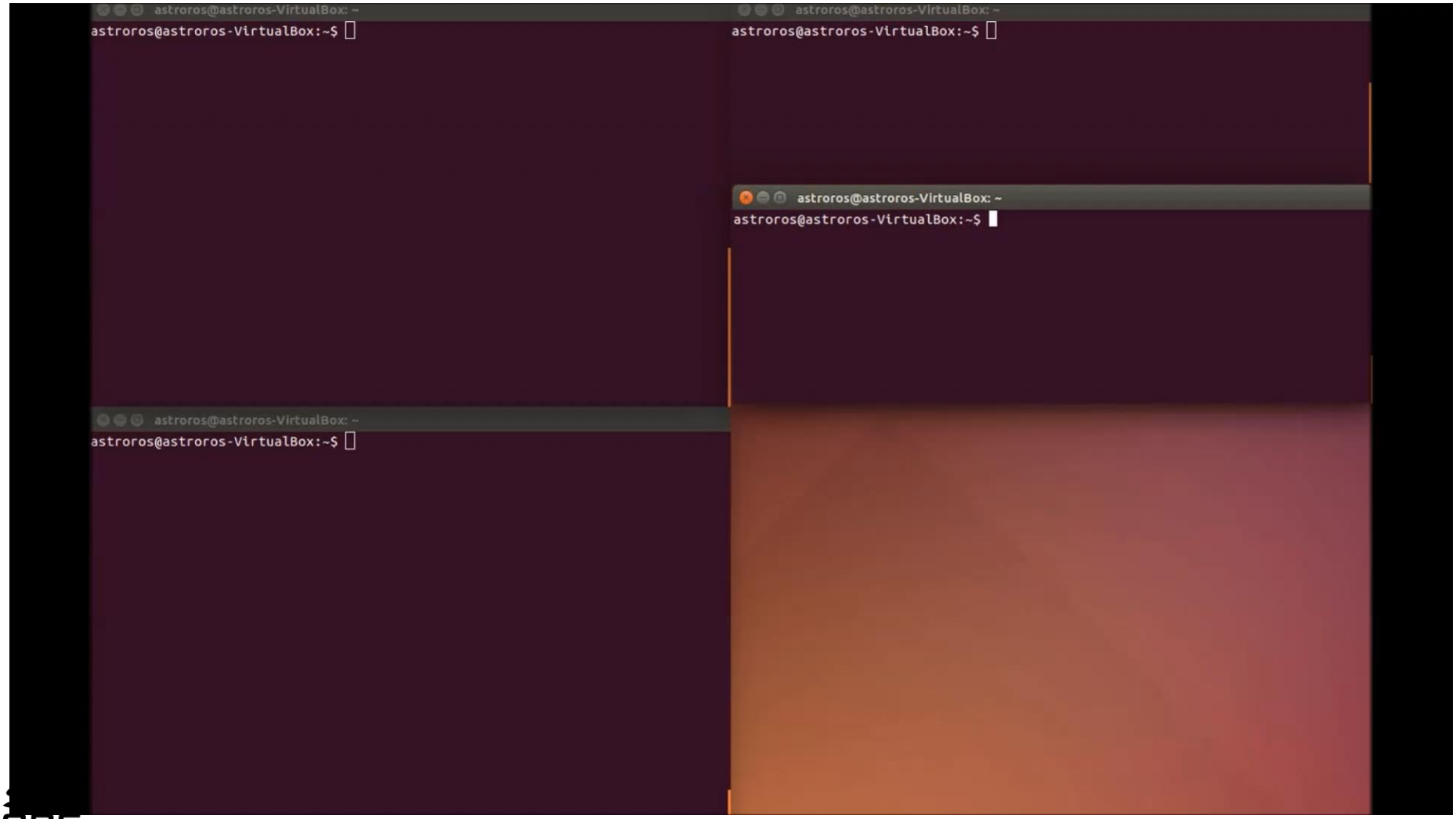






CPE LYON
ÉCOLE SUPÉRIEURE DE CHIMIE PHYSIQUE ÉLECTRONIQUE DE LYON

ROS: Command line



ROS: Sample Python publisher

```
#!/usr/bin/env python
# license removed for brevity
import rospy
from geometry_msgs.msg import Twist

def talker():
   pub = rospy.Publisher('/turtle1/cmd_vel', Twist)
   rospy.init_node('Turtle_Manager', anonymous=True)
   r = rospy.Rate(10) # 10hz
   while not rospy.is_shutdown():
                  twist = Twist()
      twist.linear.x = 0.1;
      rospy.loginfo(twist)
      pub.publish(twist)
      r.sleep()

if __name__ == '__main__':
   try:
      talker()
   except rospy.ROSInterruptException: pass
```

ROS: Sample C++ publisher

```
#include "ros/ros.h"
#include <geometry_msgs/Twist.h>
#include <sstream>

int main(int argc, char **argv)
{
 ros::init(argc, argv, "Turtle_Manager");
 ros::NodeHandle n;
 ros::Publisher turtle_pub = n.advertise<geometry_msgs::Twist>("/turtle1/cmd_vel", 1000);
 ros::Rate loop_rate(10);

 int count = 0;
 while (ros::ok())
 {
   geometry_msgs::Twist base_cmd;
   base_cmd.linear.x = 0.1;
  msg.data = ss.str();
  ROS_INFO("%s", base_cmd.str());

  turtle_pub.publish(base_cmd);

  ros::spinOnce();
  loop_rate.sleep();
  ++count;
 }
 return 0;
}
```

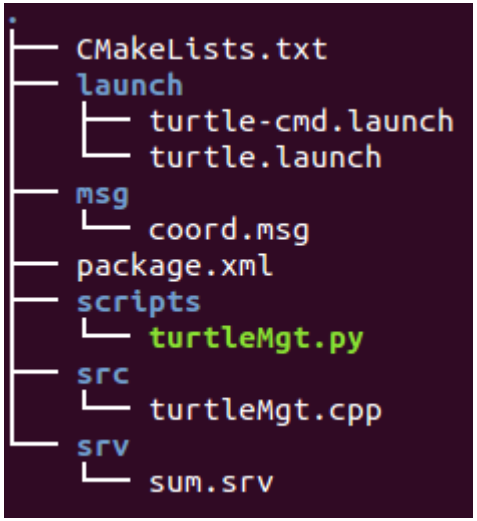
ROS: Distribution

- Package

  Contains nodes and libraries that logically constitues a useful module

- Composition

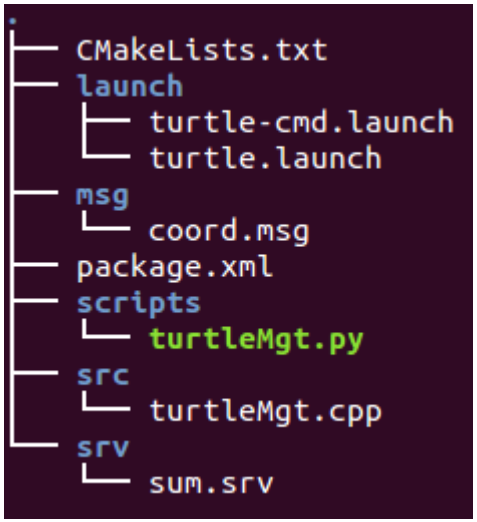
/src	cpp source files
/scripts	python source files
/msg	message definition files
/srv	service definition files
/launch	launch files
package.xml	package manifests (metadata about package)
CMakeLists.txt	build and installation instruction



ROS: Distribution

CMakeLists.txt (1/2)

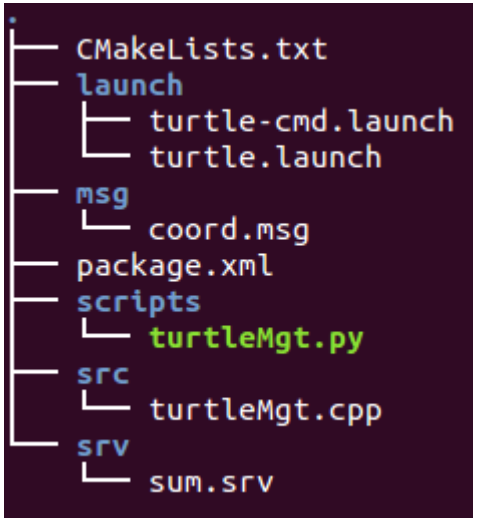
```
cmake_minimum_required(VERSION 2.8.3)
project(turtle_sample)

find_package(catkin REQUIRED COMPONENTS
  roscpp
  rospy
  std_msgs
  message_generation
)
## Generate messages in the 'msg' folder
 add_message_files(
   FILES
   coord.msg
)
## Generate services in the 'srv' folder
 add_service_files(
   FILES
   sum.srv
 )
## Generate added messages and   services with any dependencies listed here
 generate_messages(
   DEPENDENCIES
   std_msgs
 )
```



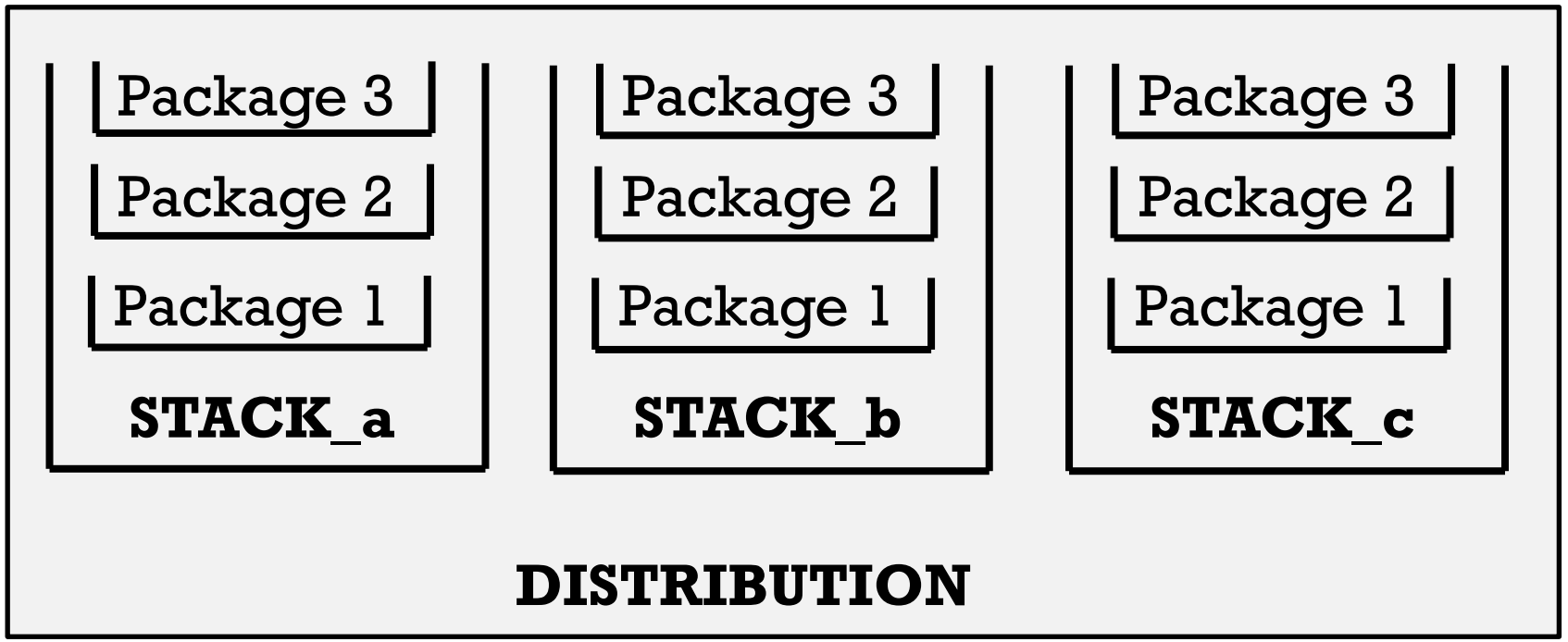
ROS: Distribution

CMakeLists.txt (2/2)

```
catkin_package(
  CATKIN_DEPENDS roscpp
    rospy
    std_msgs
     message_runtime
)
include_directories(
  ${catkin_INCLUDE_DIRS}
)
## Declare a cpp executable
 add_executable(turtleMgt src/turtleMgt.cpp)
## Add cmake target dependencies of the executable/library
 add_dependencies(turtleMgt
turtle_sample_generate_messages_cpp)
## Specify libraries to link a library or executable target
against
 target_link_libraries(turtleMgt
   ${catkin_LIBRARIES}
 )
```



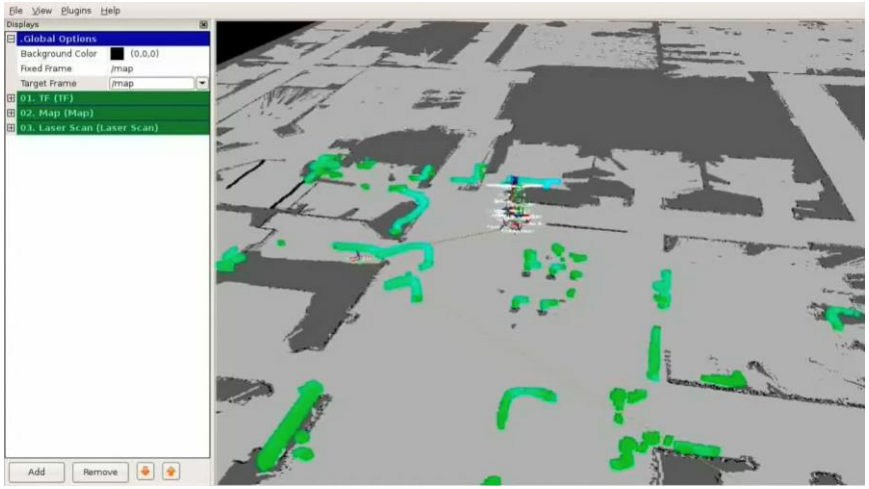
ROS: Distribution

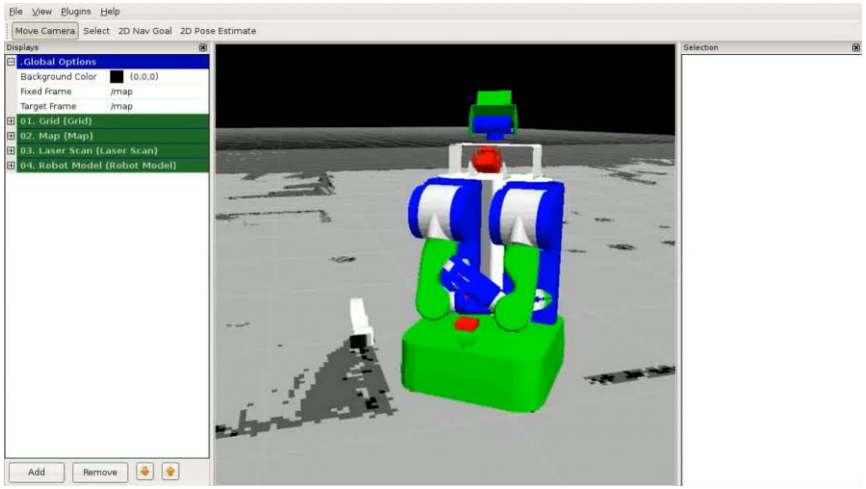


ROS: Toolbox

RVIZ

```
$ rosrun rviz rviz
```



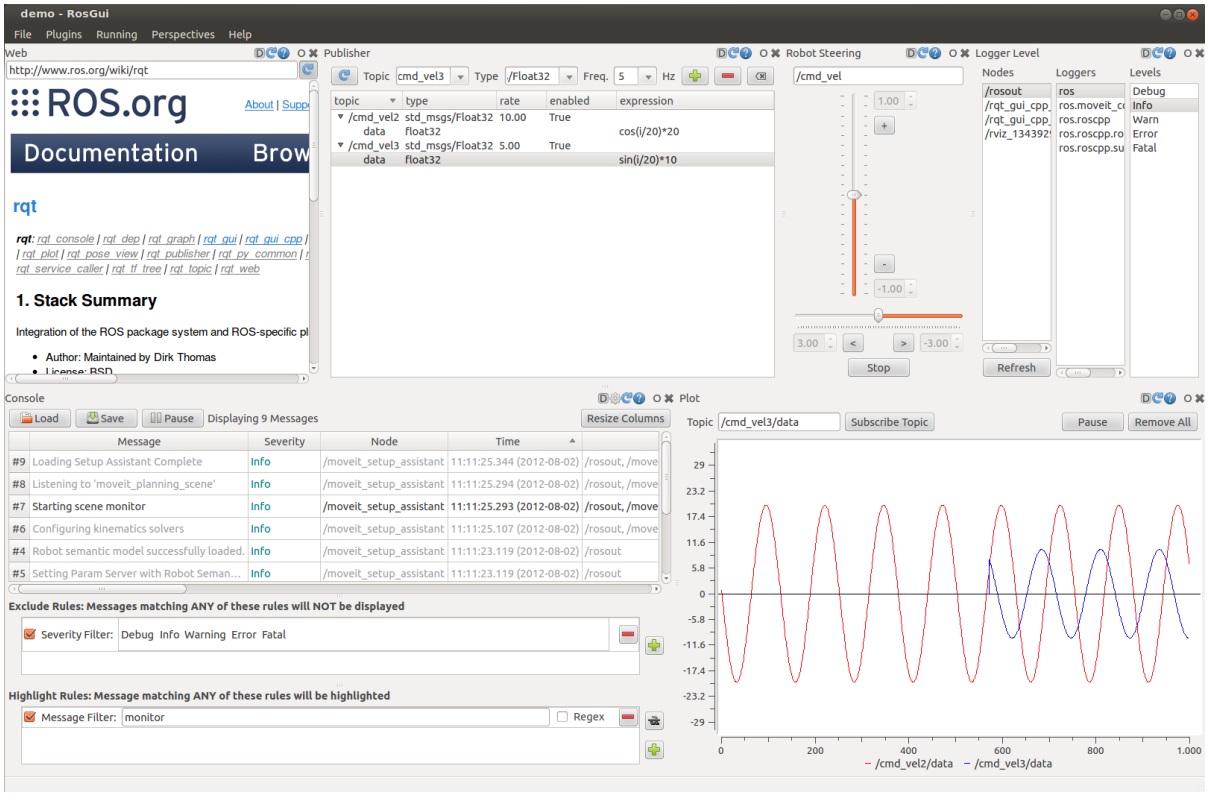


ROS: Toolbox

RQT Software framework implementing various GUI tools in the form of plug-ing

Usage

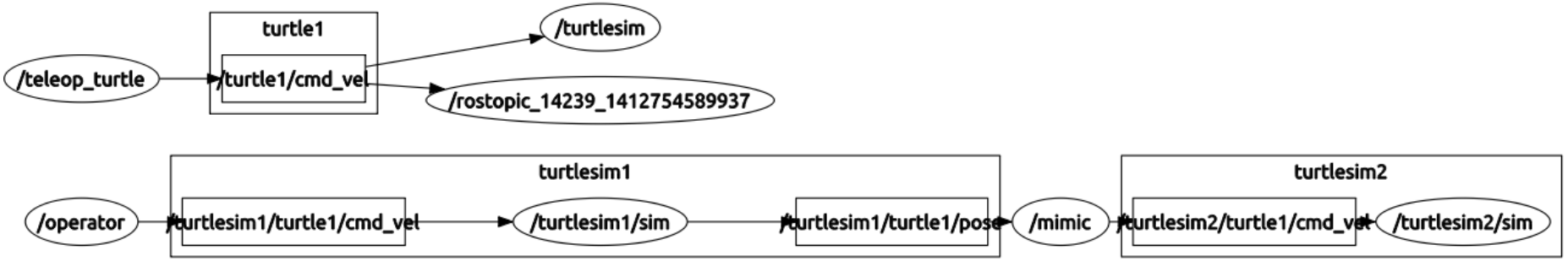
```
rqt_graph
rqt_image_view
rqt_logger_level
rosrun rqt_gui rqt_gui
```



ROS: Toolbox

- Usage

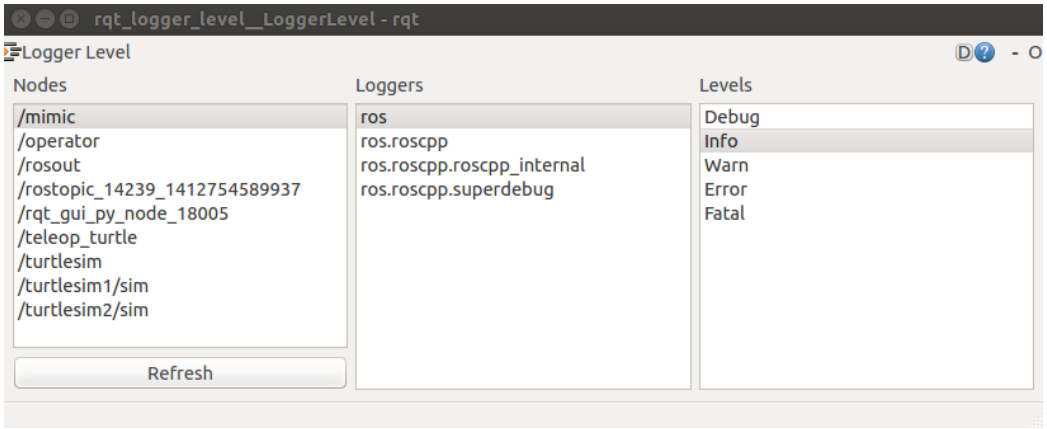
`rqt_graph`	`display a graph of ros Node and topics`
`rqt_image_view`	`display available image topics`
`rqt_logger_level`	`set Ros log level per ros node`
`rosrun rqt_gui rqt_gui`	`gui container for ros helper tools`

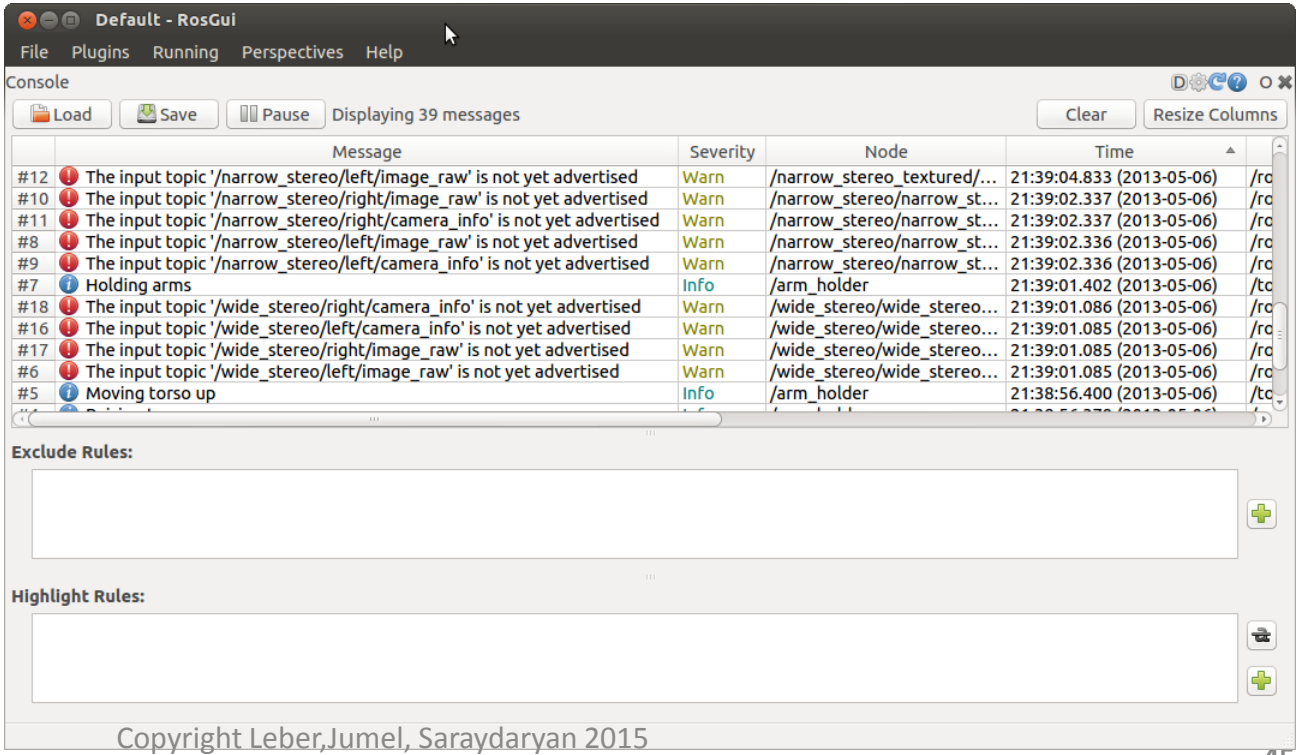


ROS: Toolbox

- Usage for debug

```
rqt_graph
rqt_logger_level
rqt_console
```





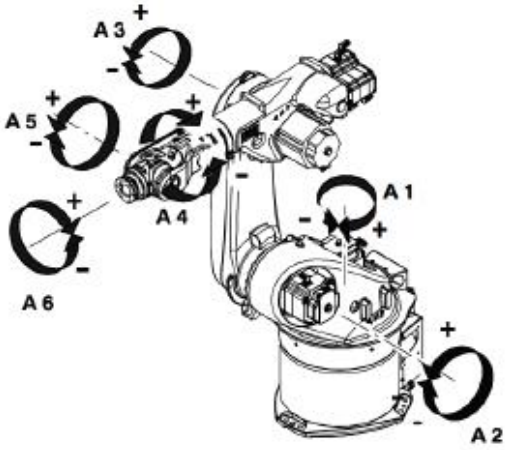
ROS: Libraries

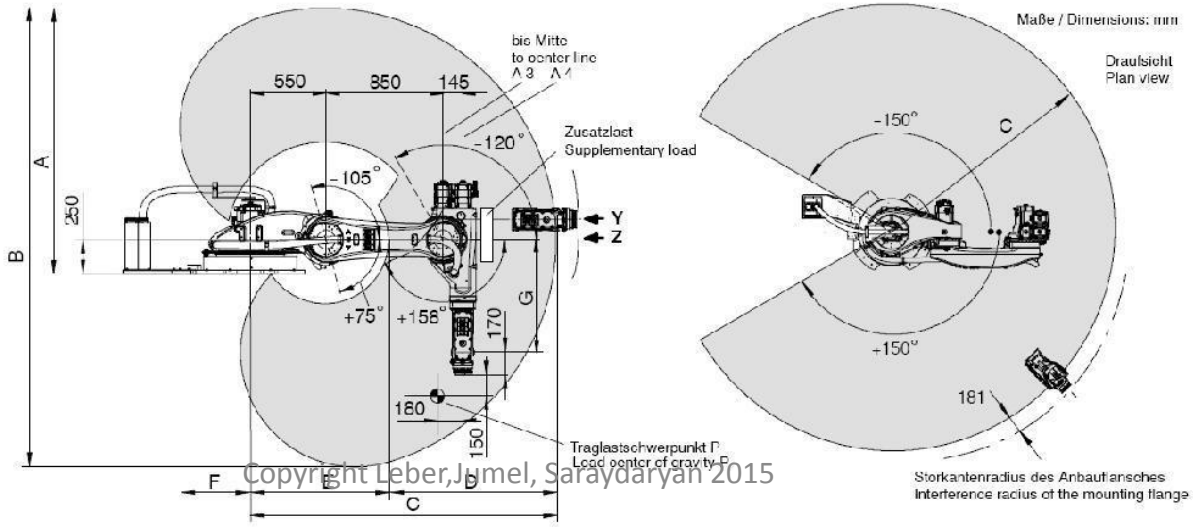
About Arm and Gripper Manipulation with ROS; **TF** and **Move it**

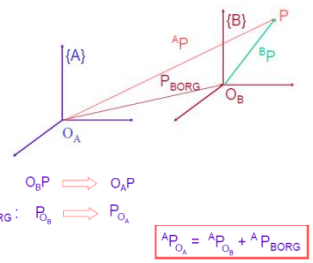
KR 30-3

Le KR 30-3, un artiste avec une enveloppe d'évolution sphérique, est l'idéal quand il s'agit d'espace et d'investissements limités.

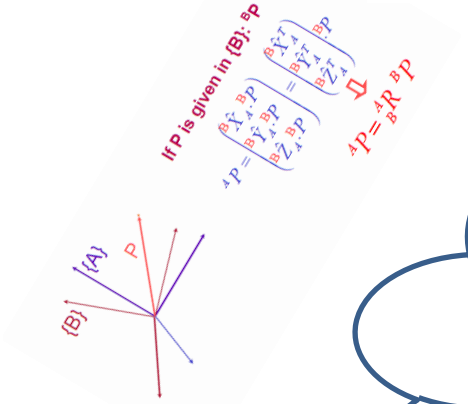
Charges	
Charge admissible	30 kg
Charge supplémentaire	35 kg
Enveloppe d'évolution	
Portée max.	2033 mm
Autres caractéristiques et versions	
Nombre des axes	6
Répétabilité	<±0,06 mm
Poids	665 kg
Positions de montage	Sol, plafond
Commande	KR C4

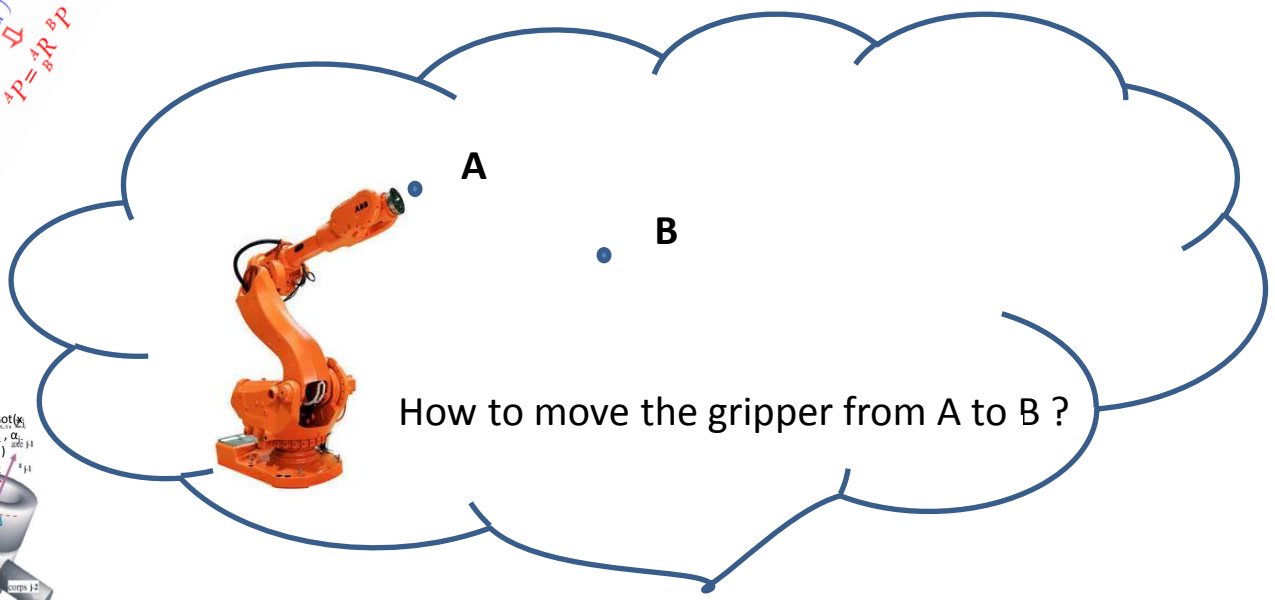


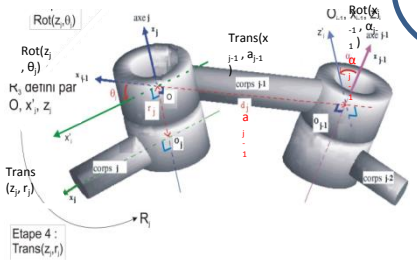




i	α_{i-1}	a_{i-1}	θ_i	r_i
1	0	0	θ1	0
2	-π/2	0	-π/2	r2
3	-π/2	L_2	θ3	0
4	π/2	0	θ4	0
5	0	L_4	0	L_5





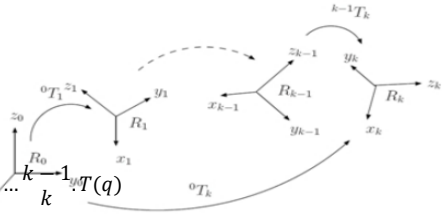


$$T_{i-1,\,i} = \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_{i-1} & -\sin\alpha_{i-1} & 0 \\ 0 & \sin\alpha_{i-1} & \cos\alpha_{i-1} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{R(\underline{x}_{i-1},\,\alpha_{i-1})} \underbrace{\begin{pmatrix} 1 & 0 & 0 & a_{i-1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{\text{translation de } a_{i-1}\underline{x}_{i-1}} \underbrace{\begin{pmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{R(\underline{z}_i,\,\theta_i)} \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & r_i \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{\text{translation de } r_i\underline{z}_i}$$

soit :

$$T_{i-1,\,i} = \begin{pmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \cos\alpha_{i-1}\sin\theta_i & \cos\alpha_{i-1}\cos\theta_i & -\sin\alpha_{i-1} & -r_i\sin\alpha_{i-1} \\ \sin\alpha_{i-1}\sin\theta_i & \sin\alpha_{i-1}\cos\theta_i & \cos\alpha_{i-1} & r_i\cos\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (2.1)$$

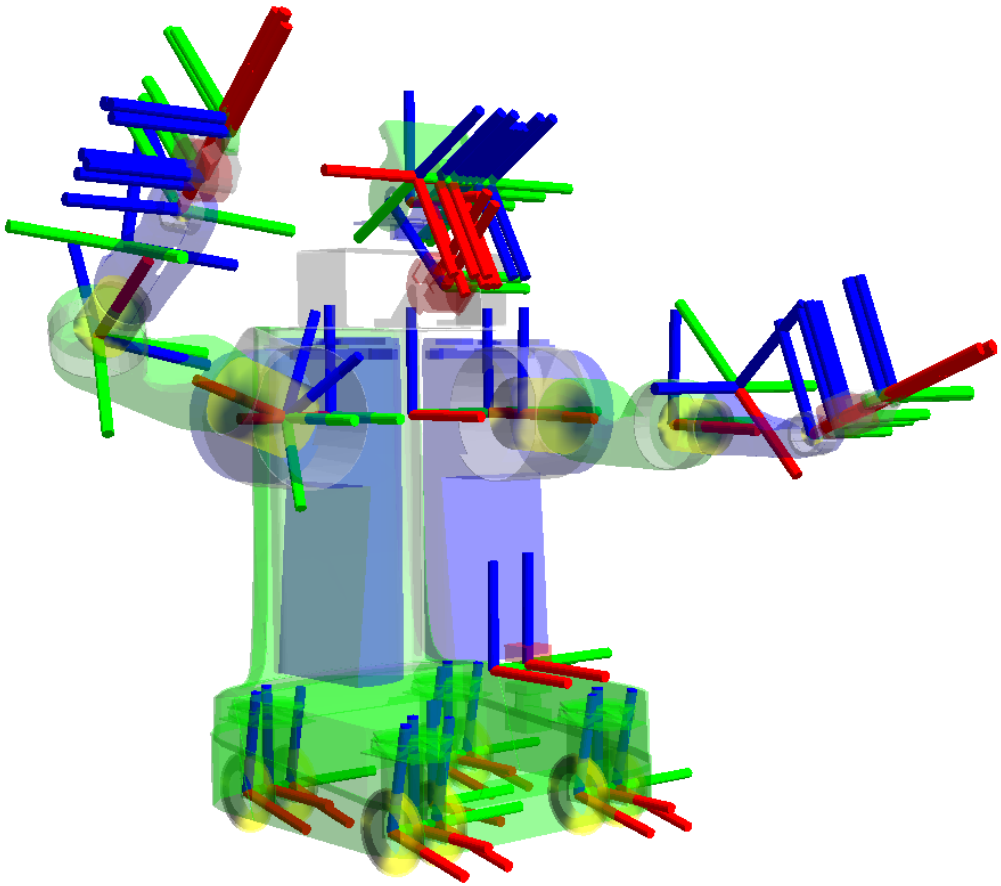
$${}^0_kT(q) = {}^0_1T(q).{}^1_2T(q).{}^2_3T(q)\ldots{}^{k-1}_kT(q)$$



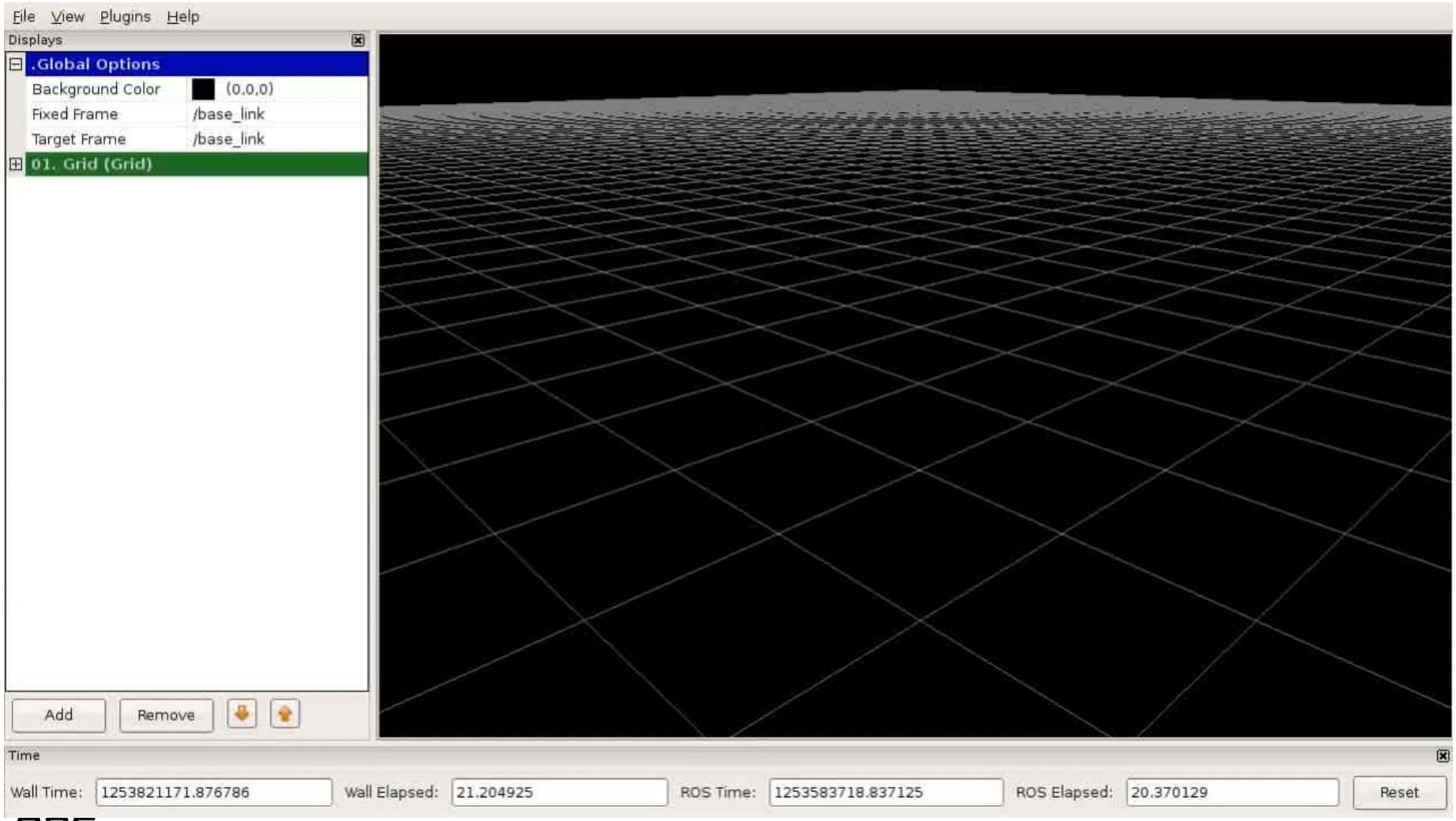
TF

- Definition:
  Move a robot arm to a given position

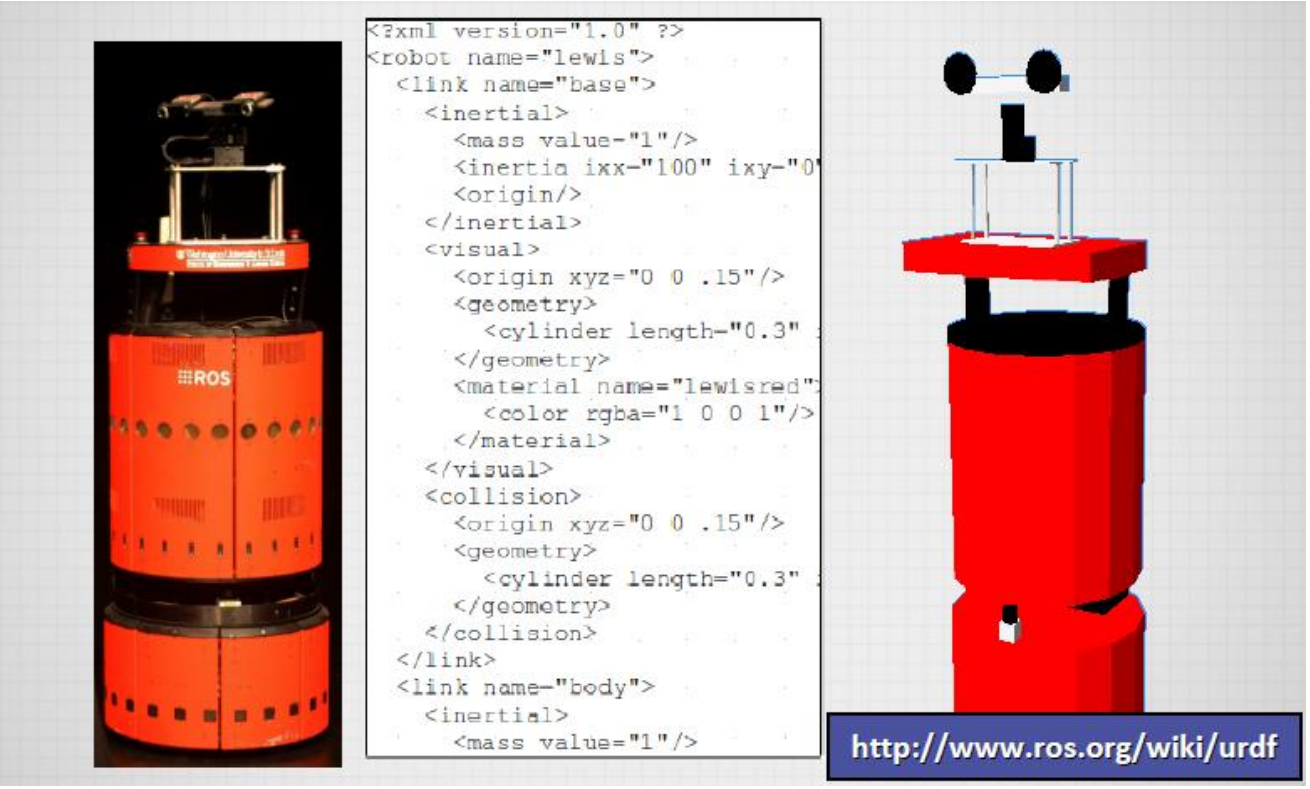
- Properties
  register object coordiantes,
  transform data from one
  landmark to another
  distributed architecture
  timestamp coordinate frame



TF

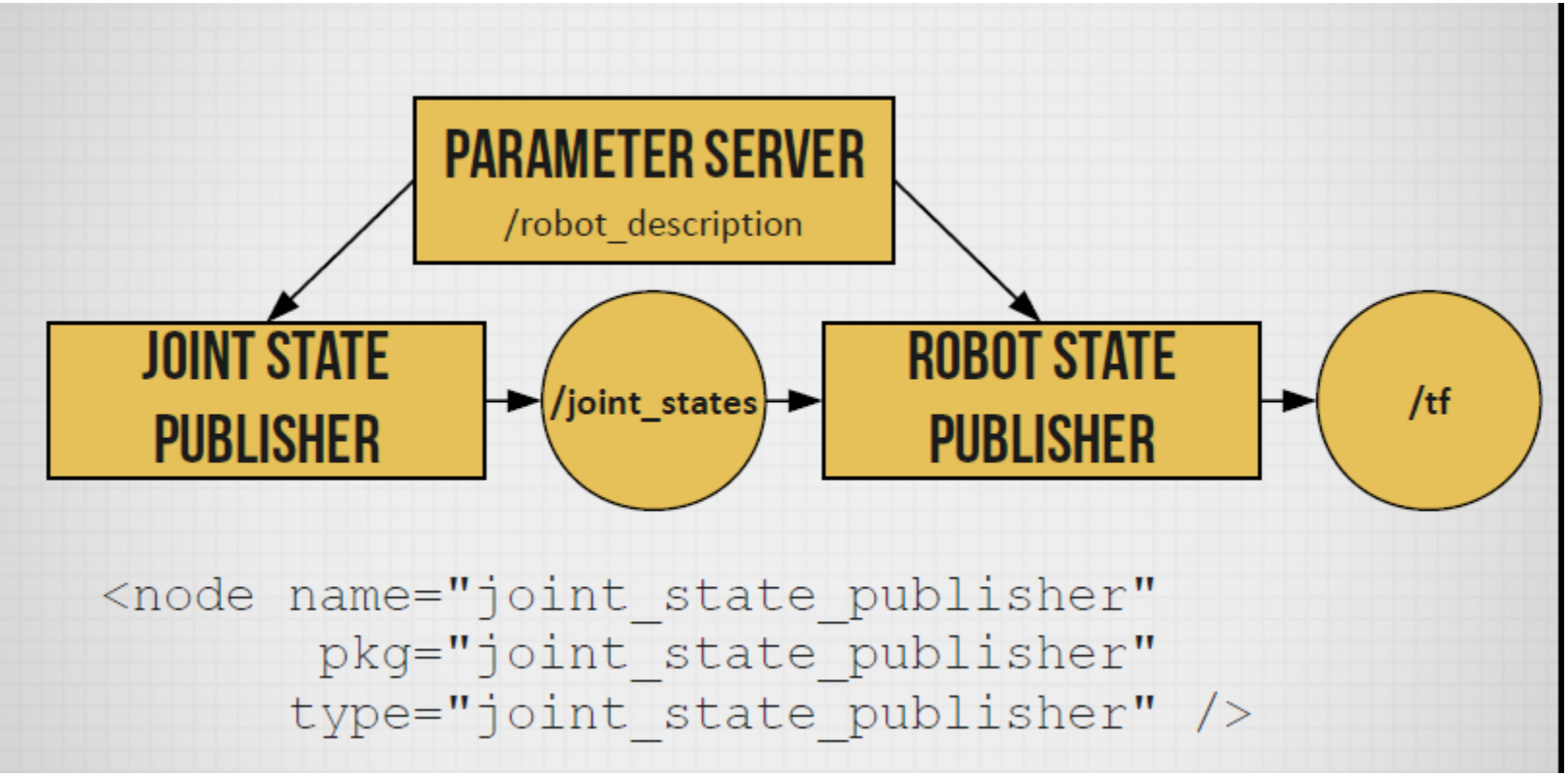


URDF; Unified Robot Description Format



from David Lu ; ROSCON 2012 presentation

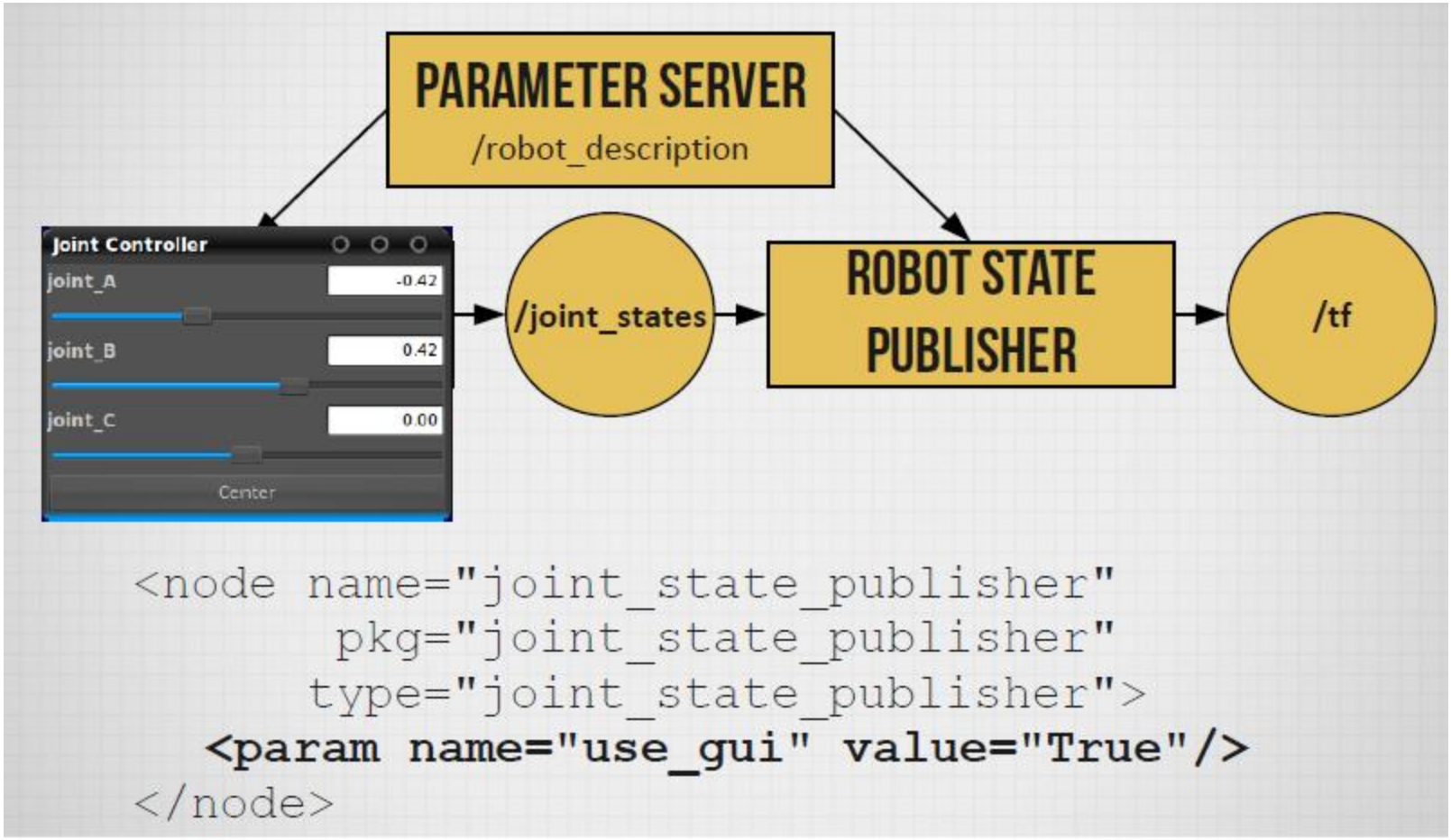
State Publisher




```
<node name="joint_state_publisher"
      pkg="joint_state_publisher"
      type="joint_state_publisher" />
```

from David Lu ; ROSCON 2012 presentation

Generated Gui Controller

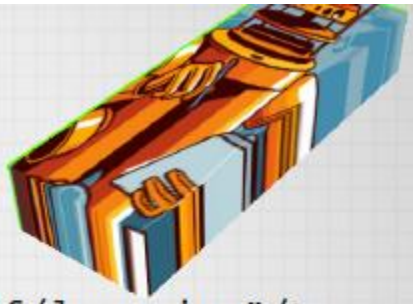


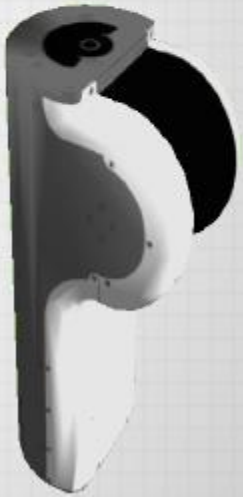

```
<node name="joint_state_publisher"
      pkg="joint_state_publisher"
     type="joint_state_publisher">
  <param name="use_gui" value="True"/>
</node>
```

from David Lu ; ROSCON 2012 presentation

Model Representation

```
<link name="box">
 <visual>
    <geometry>
        <box size="1.5 .5 .25"/>
    </geometry>
    <material name="pattern">
        <texture filename="package://roscon_urdf/logo.jpg"/>
    </material>
 </visual>
</link>
<link name="mesh">
 <visual>
    <geometry>
      <mesh filename=
        "package://roscon_urdf/shoulder_pan.dae" />
    </geometry>
 </visual>
</link>
```



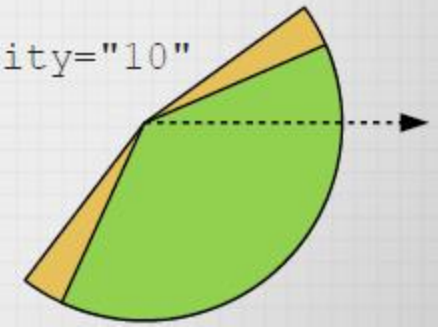


from David Lu ; ROSCON 2012 presentation

Model Simulation

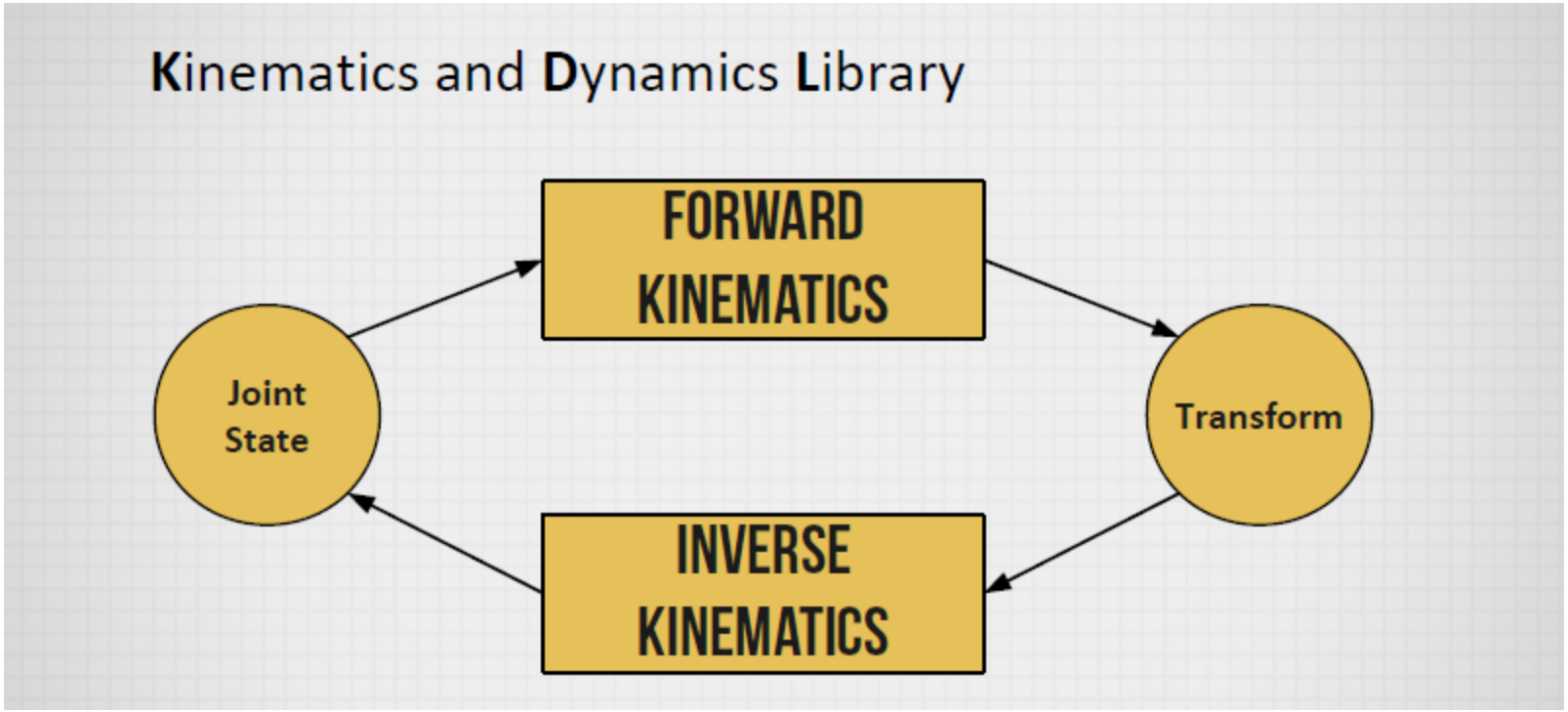
```
<link name="r_shoulder_pan_link">
  <inertial>
    <mass value="25.799322"/>
    <origin xyz="-.001 .024 -.098"/>
    <inertia ixx=".866" ixy="-.061" ixz="-.121"
             iyy=".874" iyz="-.059" izz=" .274"/>
  </inertial>
  ...
</link>
<joint name="r_shoulder_pan_joint" type="revolute">
  ...
  <limit effort="30" velocity="2.088"
         lower="-2.28" upper=".71" />
  <safety_controller k_position="100" k_velocity="10"
         soft_lower_limit="-2.14
         soft_upper_limit="0.56"/>
  <dynamics damping="10.0"/>
  <calibration rising="-.785"/>
</joint>
```

$$\mathbf{I} = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}$$



from David Lu ; ROSCON 2012 presentation

How move gripper from A to B ?



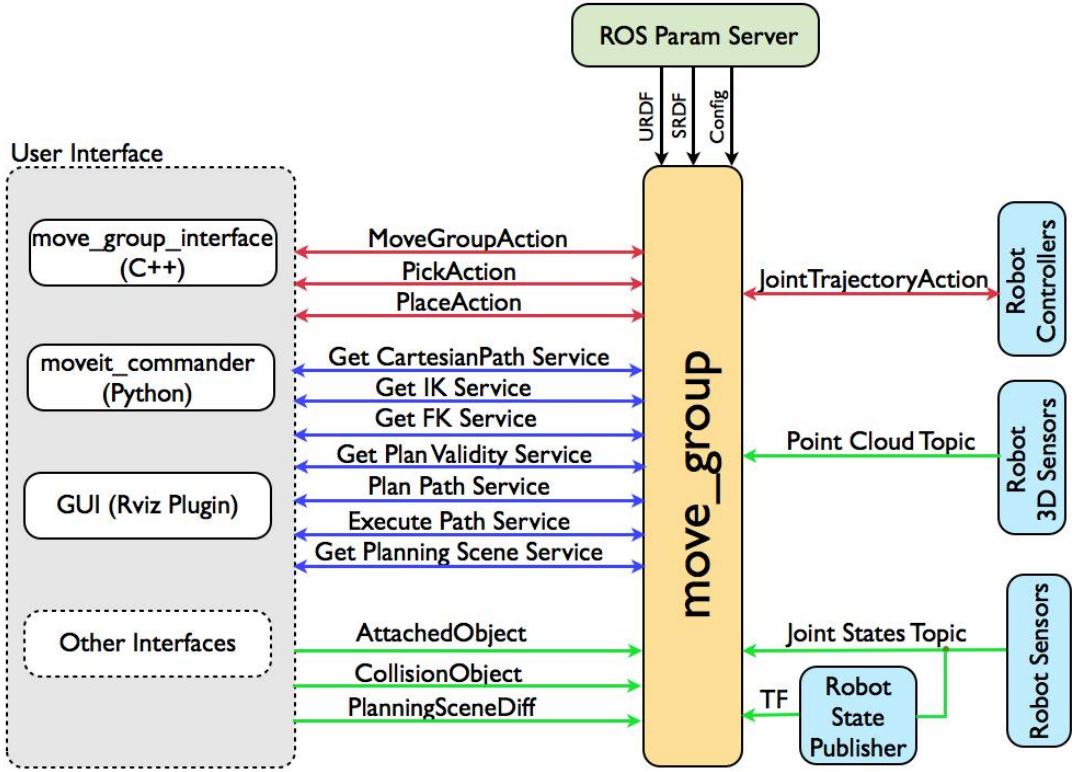
from David Lu ; ROSCON 2012 presentation

Move it !

- Definition:
  Move a robot arm to a given position

- Properties

  Inverse kinematics
  motion planning,
  manipulation,
  3D perception,
  kinematics,
  control and navigation
  Collision avoidance

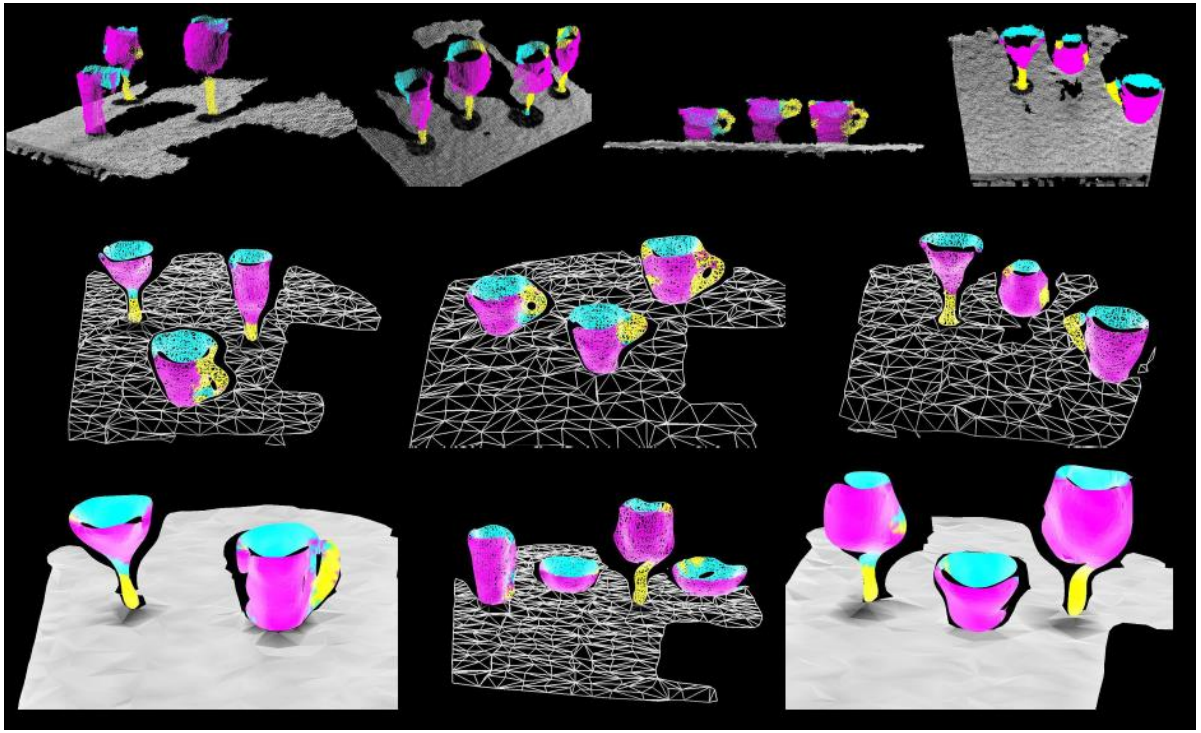


Move it !

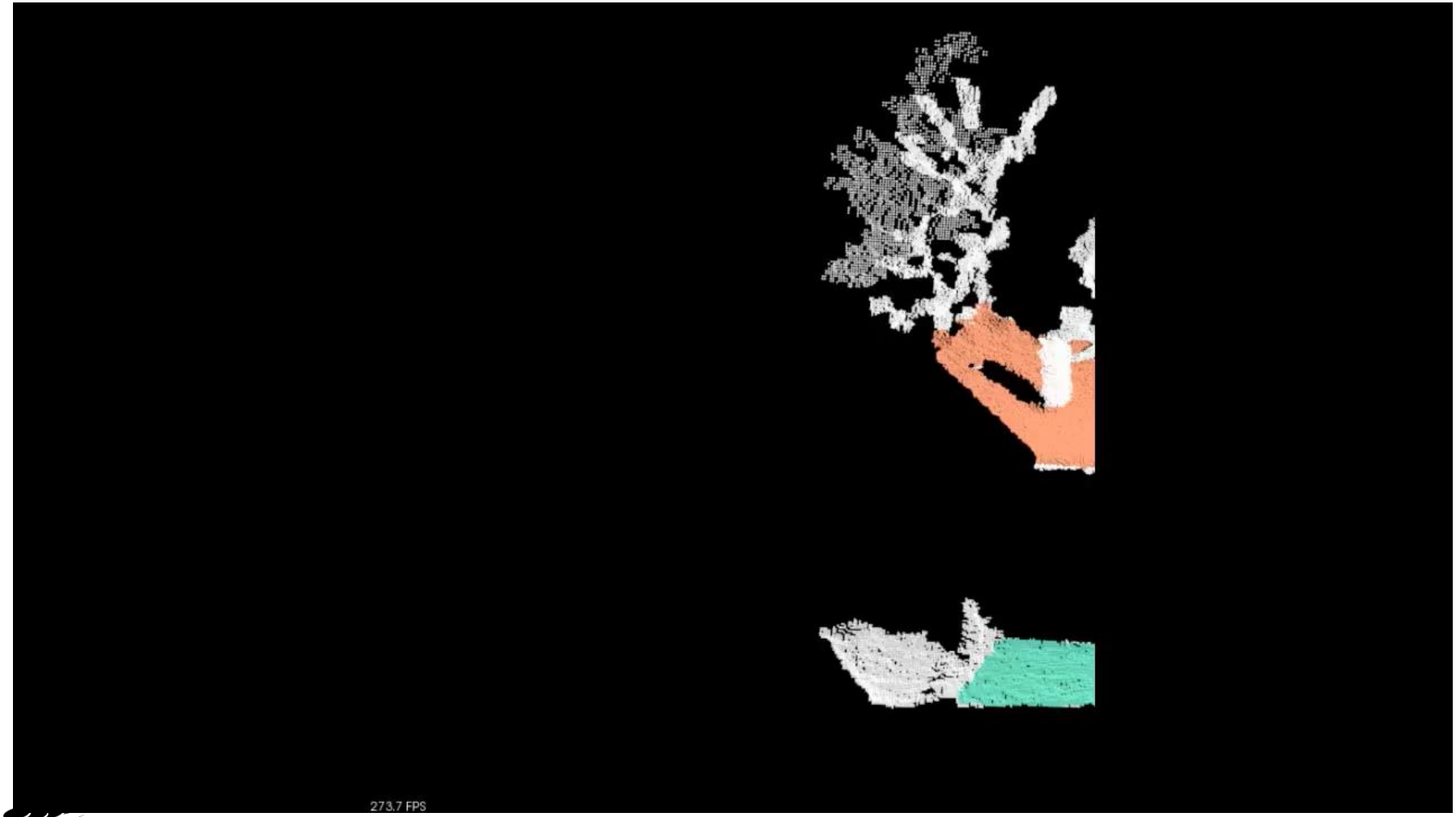
PCL Point Cloud Library

- Definition:
  2D-3D point cloud processing

- Properties
  Filtering
  Feature estimation
  Surface reconstruction
  Model fitting
  Segmentation …

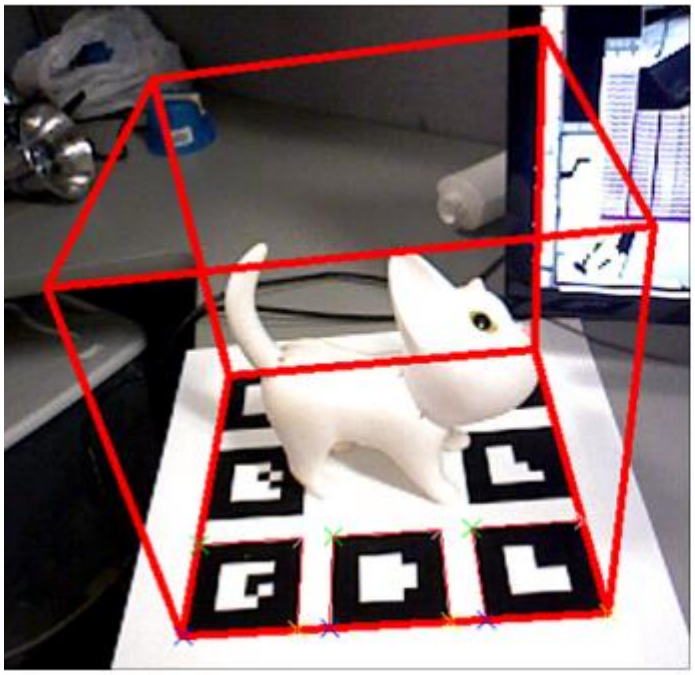


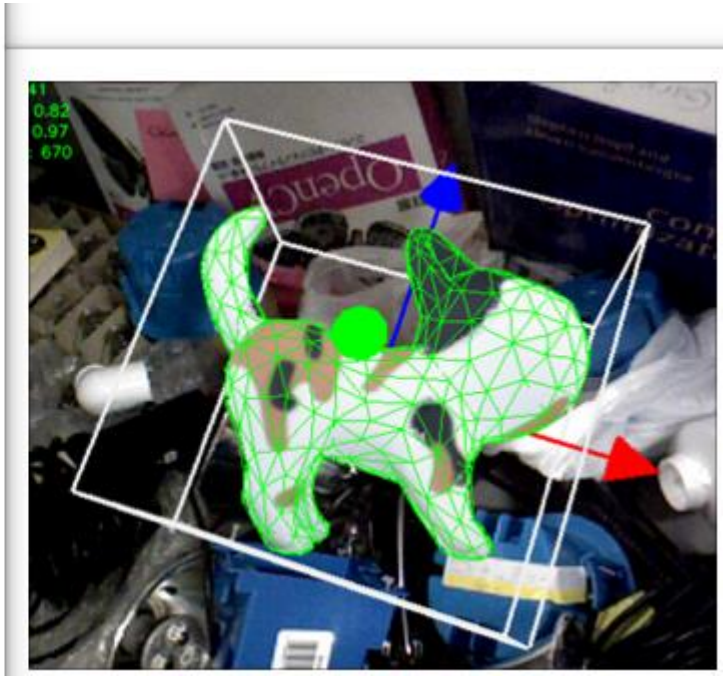
PCL Point Cloud Library



OpenCV

- Definition :
  an Open source Computer Vision and machine learning software library

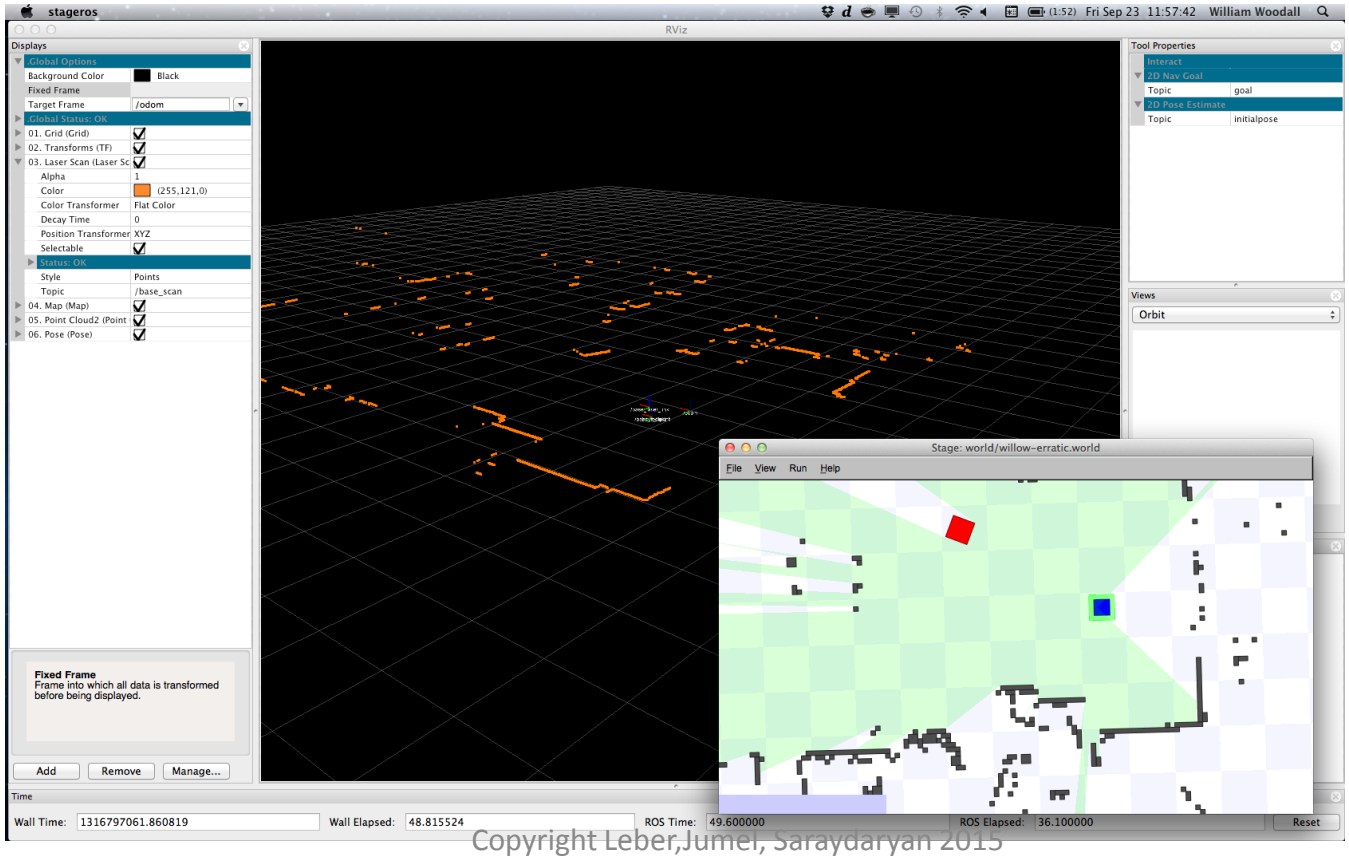




Stage simulator

- Definition :

Stage is a robot simulator. It provides a virtual world populated by mobile robots and sensors, along with various objects for the robots to sense and manipulate.



Gazebo simulator

- Definition:
  Gazebo is a simulator of complex robots, probe and environment simulator

Gazebo simulator

Our ROS Feedback

Ros Usage

Robocup@work 2014

Pick Action

Precision place Action

Smart Robotic Room

People activity follow up

Ros Usage

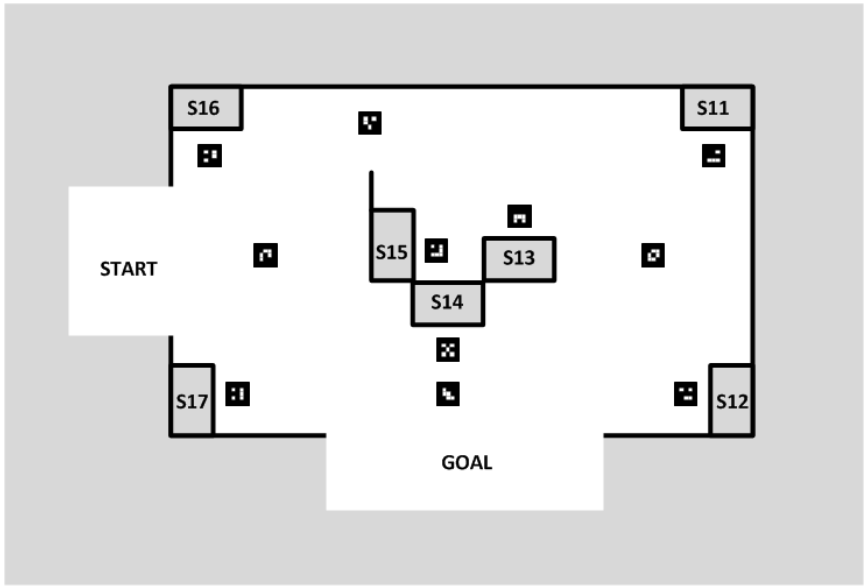
Pros

- Free and open-source
- Lot of robots and equipments compliant
- Tools box impressive
- Very Active community (lots of examples and new tools)
- More and more mature (stability, easy to use, customization)

Cons

- Inequal contribution maturity
- Currently lot of difficulty to use package between different versions
- Design for a single core on a single computer (need to synchronise time between computers, no official multi-core support)
- Security and reliability need to be evaluated

Robocup@work Usecase



Working area

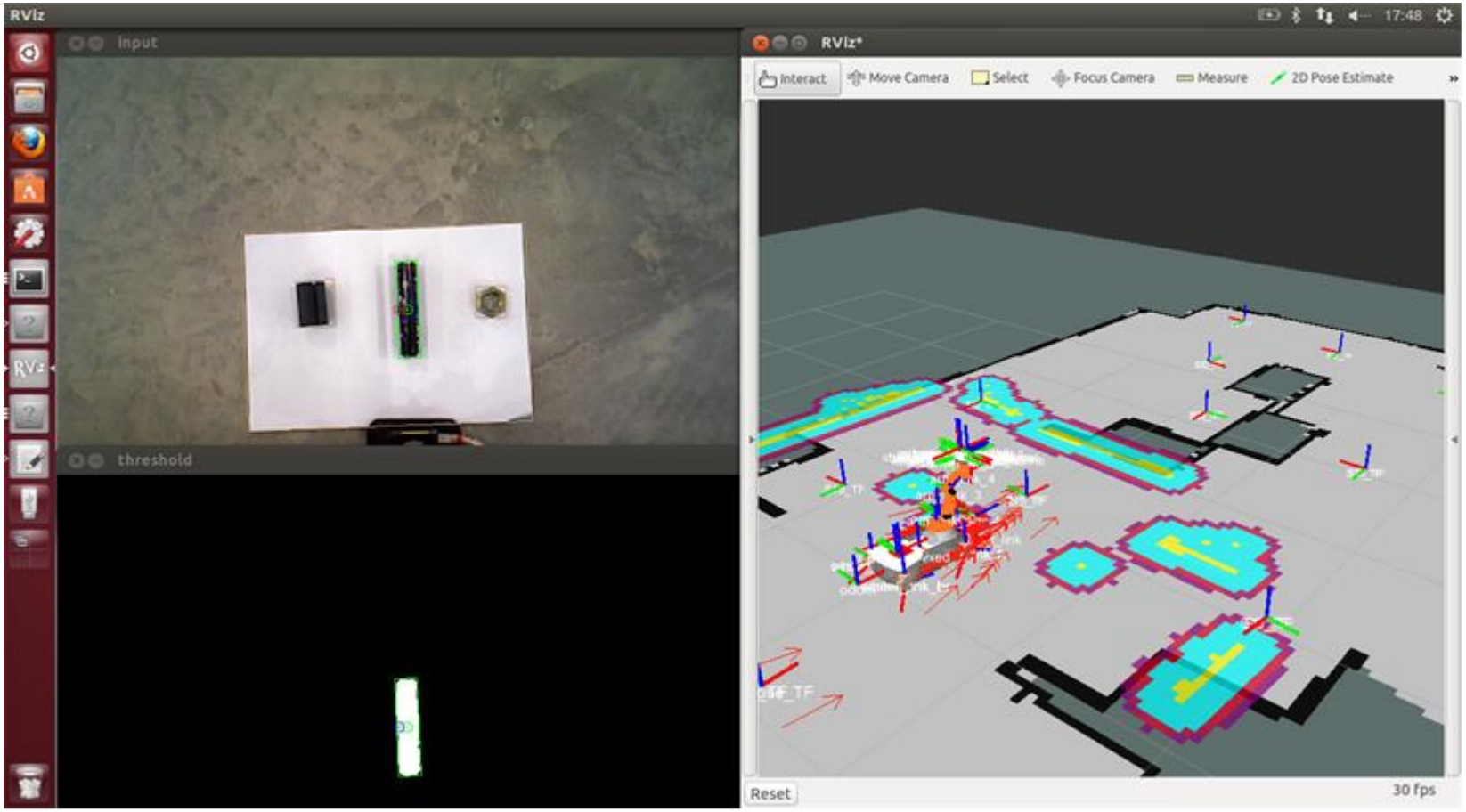
KUKA Youbot Platform

- omni-directional mobile platform
- 5-degree-of-freedom manipulator
- 2-finger gripper
- real-time EtherCAT communication
- open interfaces (ROS compliant)
- basic control software
- arm and platform can be used independently

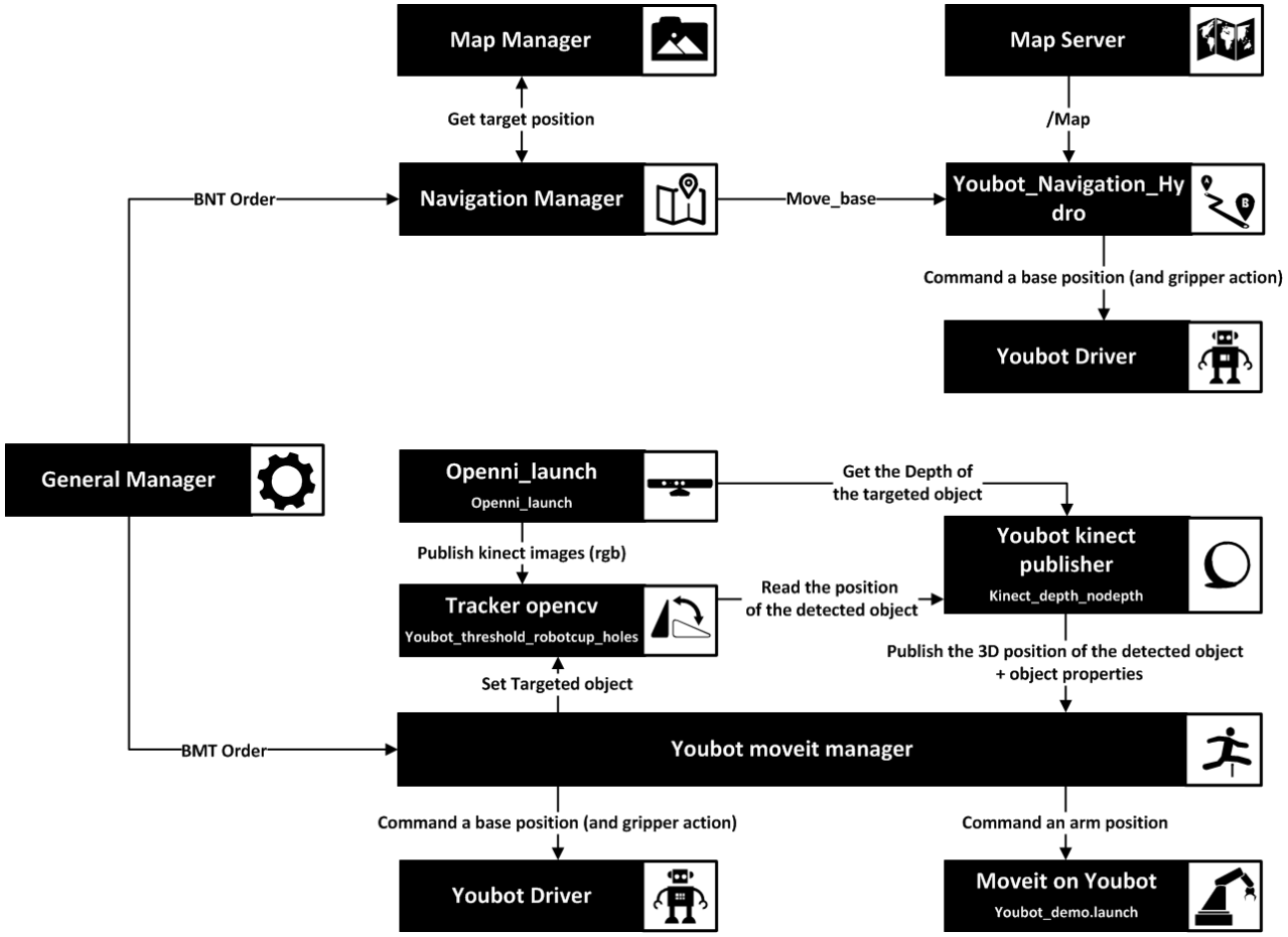
Pick Action

Precision place Action

Robocup@work Usecase



Robocup@work Usecase



References

References

http://www.ros.org/history/
http://wiki.ros.org/Robots
http://www.ros.org/about-ros/
http://wiki.ros.org/ROS/Tutorials
http://wiki.ros.org/rqt
http://wiki.ros.org/tf
http://wiki.ros.org/ROS/Introduction
http://wiki.ros.org/ROS/CommandLineTools
http://wiki.ros.org/opencv2
http://osrf.github.io/www.ros.org/features/
http://www.cnblogs.com/kekec/archive/2011/06/20/2085035.html
http://moveit.ros.org/documentation/concepts/
http://code.opencv.org/projects/gsoc2013/wiki
http://playerstage.sourceforge.net/doc/Stage-3.2.1/

- Ros Workshop , Lorenz Mösenlechner, Technishe Universität München, 2012
- Advanced Techniques for Mobile Robotics ROS, Wolfram Burgard, Cyrill Stachniss, Kai Arras, Maren Bennewitz
- Summer course on ROS framework 2013 (Instituto Superior Técnico, Lisbon)

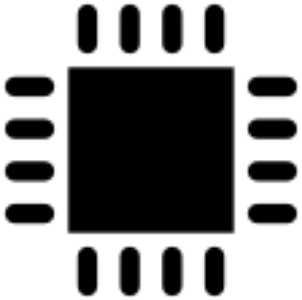
Images

http://alligatographe.blogspot.fr/2012/12/robocop-verhoeven.html
http://www.theoldrobots.org/images3/manufacturing4.JPG
http://www.cnblogs.com/kekec/archive/2011/06/20/2085035.html
http://williamjwoodall.files.wordpress.com/2011/09/screen-shot-2011-09-23-at-11-57-36-am.png
http://wiki.ros.org/tf
http://moveit.ros.org/documentation/concepts/
http://answers.ros.org/upfiles/13855650235971625.png
http://osrf.github.io/www.ros.org/img/rqt.png

Videos

Ros 5 years
https://www.youtube.com/watch?v=PGaXiLZD2KQ
TF
http://www.youtube.com/watch?v=dQyIpsMUQSw
Gazebo
https://www.youtube.com/watch?v=RvfuKP5m0w0
PCL
https://www.youtube.com/watch?v=7J-9pN96mBY
moveit
https://www.youtube.com/watch?v=dblCGZzeUqs

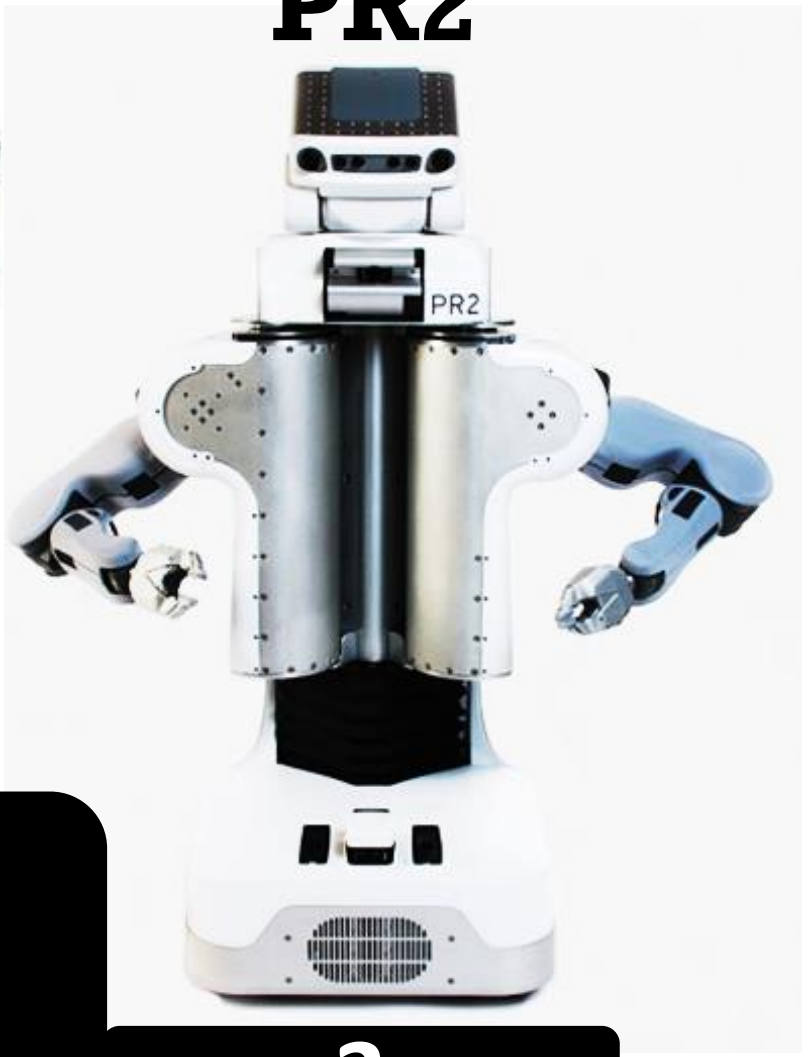
Hardware solutions for robotics of service



Emblematic Robots

Nao

PR2



Neato

2 1 3

Nao

Nao

Proprioceptive sensors ?

Exteroceptive sensors ?

Actuators ?

Proprioceptive sensors

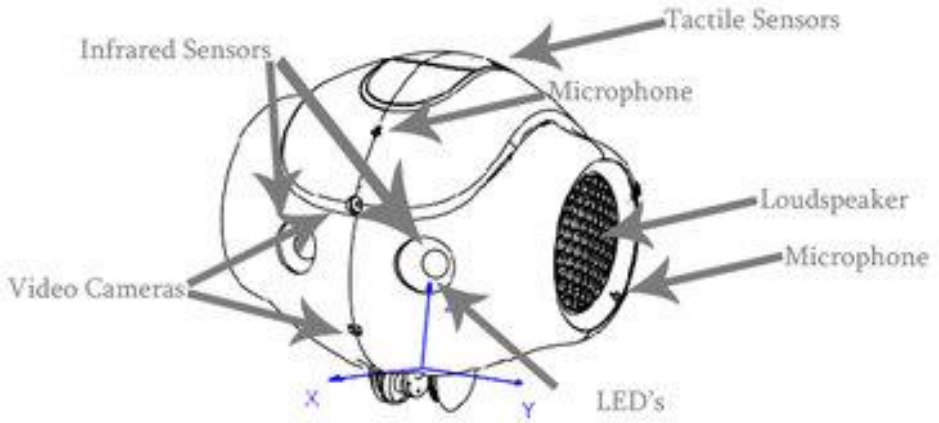
- MRE (Magnetic Rotary Encoder)
- Temperature sensor
- Current sensors

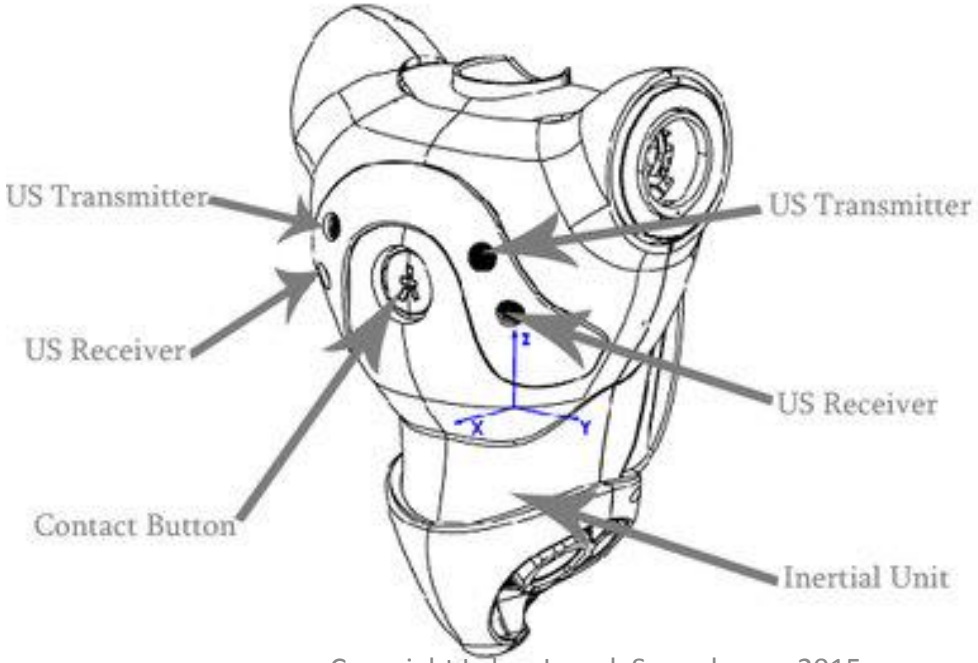
Exteroceptive sensors

- FSR (Force Sensing Resistor)
- Bumpers
- Inertial Unit (accelero, gyro)
- Sonar
- User button
- Cameras x2
- Microphones x4
- Infrared receiver
- Capacitive sensors

Actuators

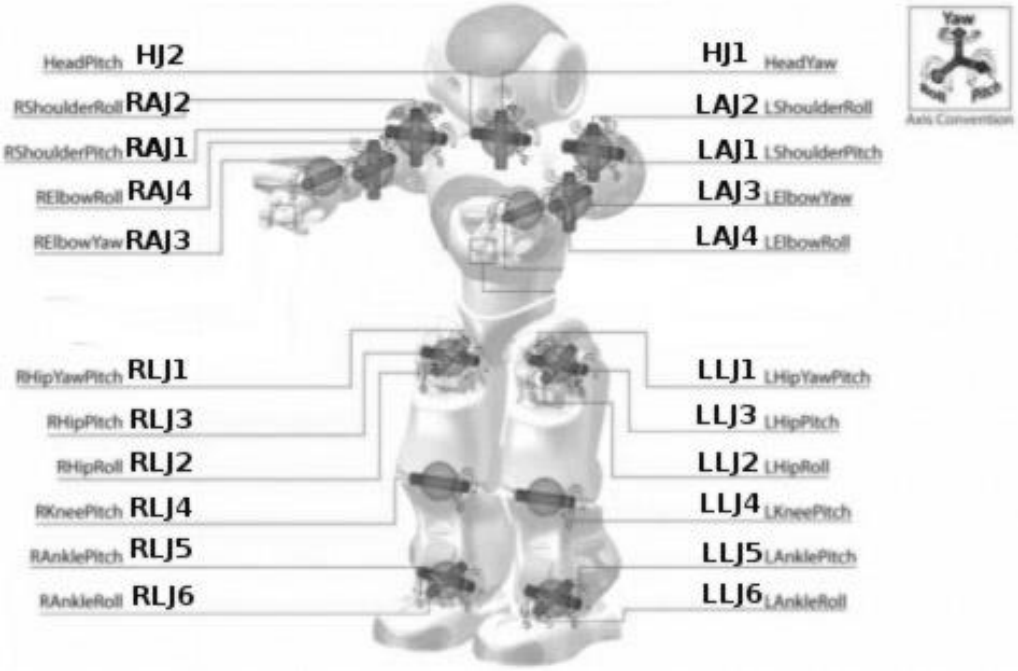
- Motors
- LEDs (visible colors)
- LEDs (infrared)
- Loudspeaker



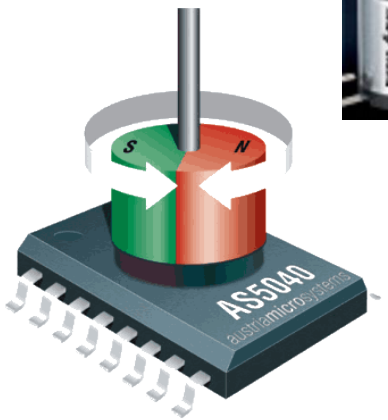


Motors + MRE subset

- Nao has 25 motors
- Nao has 34 MRE
- Very good cost/performance/size solution



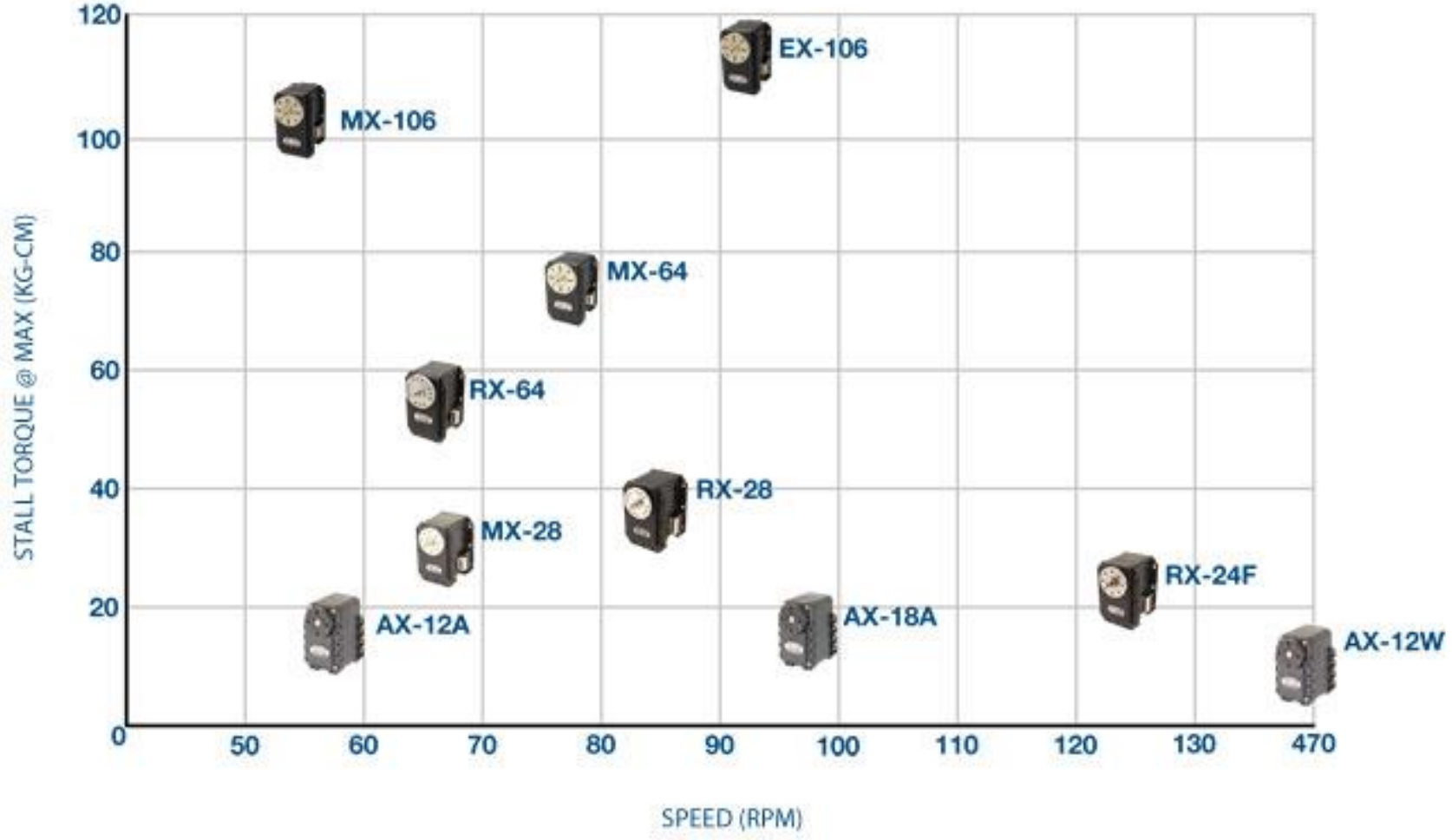
Motors + MRE subset



Vs

Classical servo motor

Example of dynamixel motors



Motor recommandations in mobile robotics

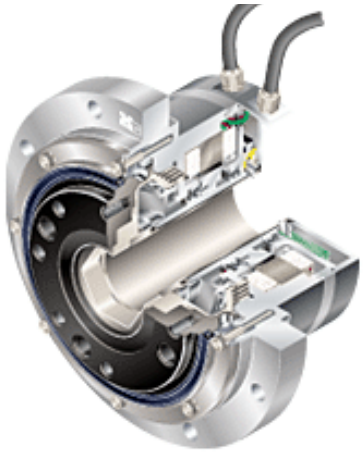
- Brushed (permanent magnets) or Brushless DC motors
  - Optimize size/integration
  - Join effectiveness and simplicity of control
- Consider thermal dissipation
  - If your robot …
    - … makes recurent movements
    - … makes lots of accelerations
    - … is position loop-controled with a « significant » torque to stand
  - Electrothermal model → Choose right product / Use as state observer

Coupling and cables ?

- Moving parts and cables… big issue!
- Two main solutions with many variants:
  - Loop/loops with cables

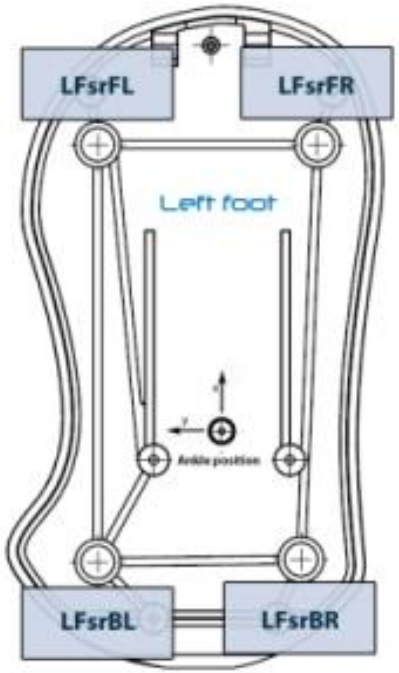
  - Use of harmonic drives

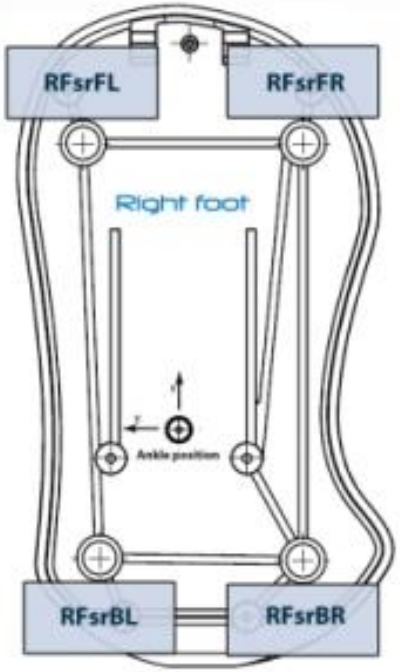
(rather for industrial arm)



- Recommendations:
  - Chose appropriate cables
  - Make endurance tests

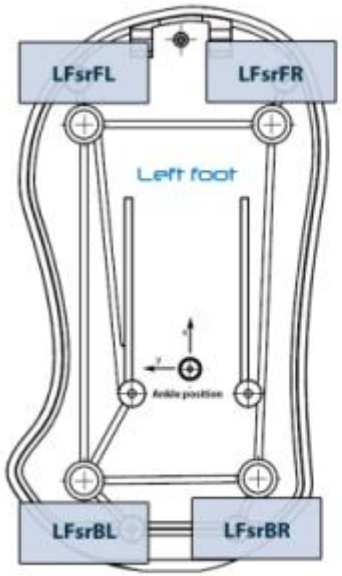
FSR (Force Sensing Resistor)

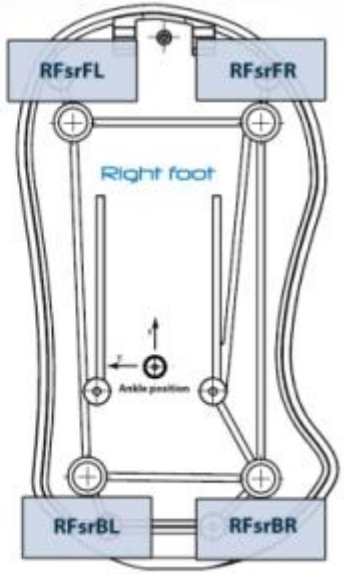




FSR (Force Sensing Resistor)

- Nao has 4 FSRs per foot
- FSR is the most accessible technologie to measure pressure
- Capacitive technology more appropriate but expensive





Neato

Proprioceptive sensors

- Optical wheel encoders
- Battery « manager »

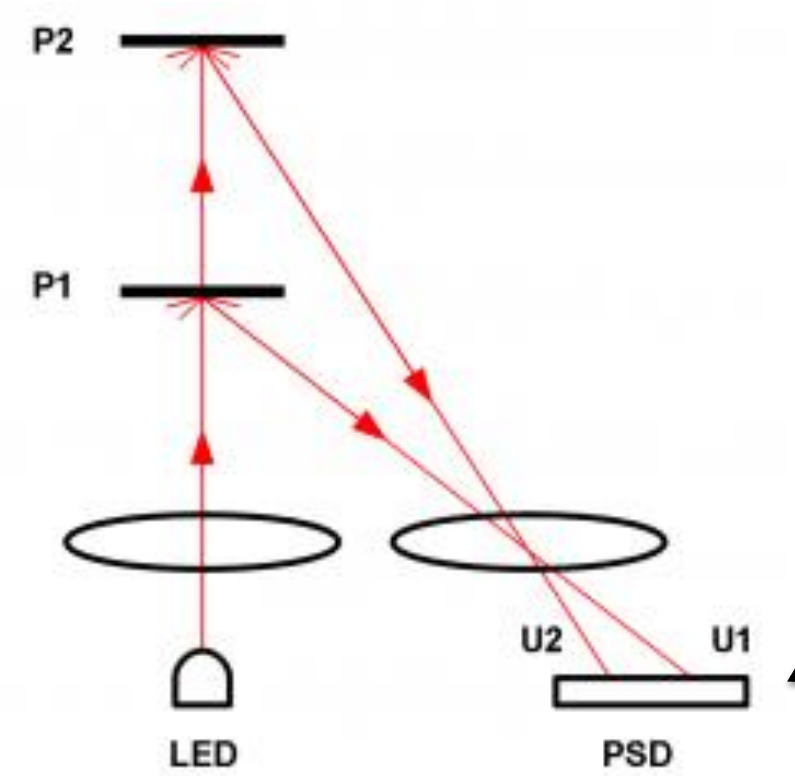
Exteroceptive sensors

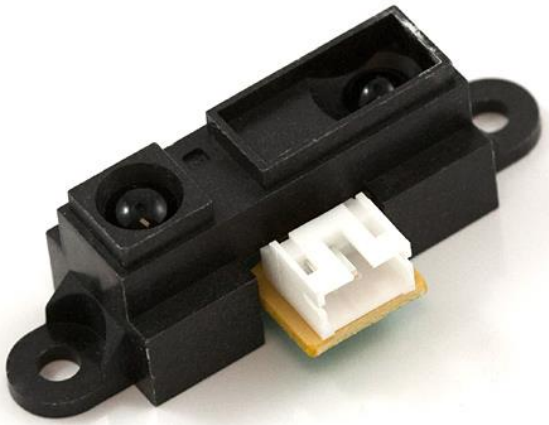
- Sharp (infrared ranger)
- Bumper
- User buttons
- Magnetic band sensor
- Lidar

Actuators

- Motors (2 wheels, fan, brush)
- LEDs (visible colors)
- Display

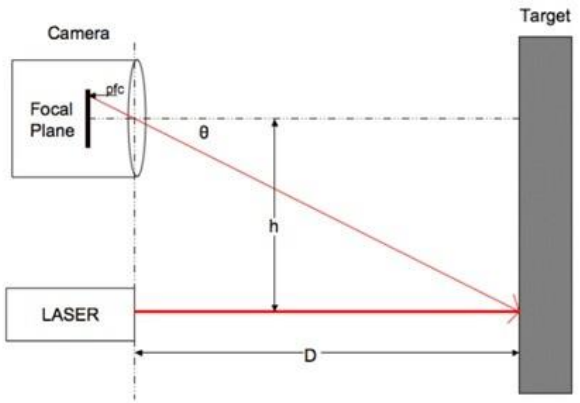
Sharp infrared ranger



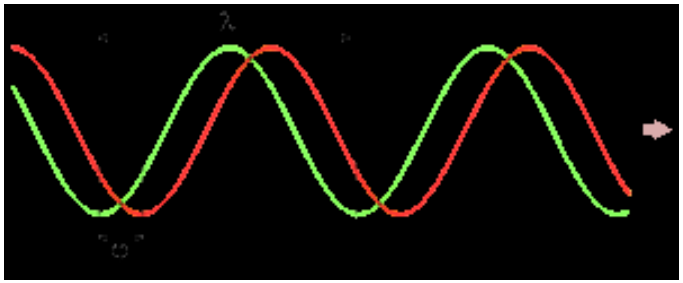


Position sensitive photo-transistors

Lidar (sharp principle)

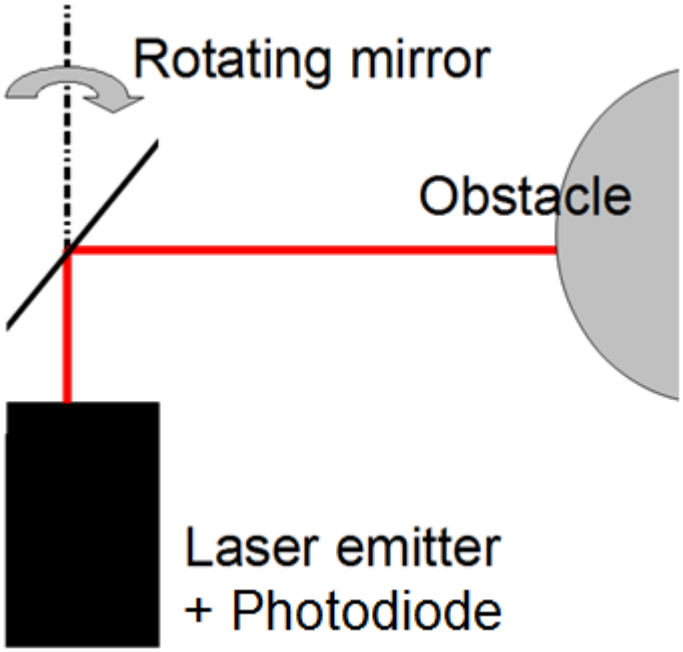


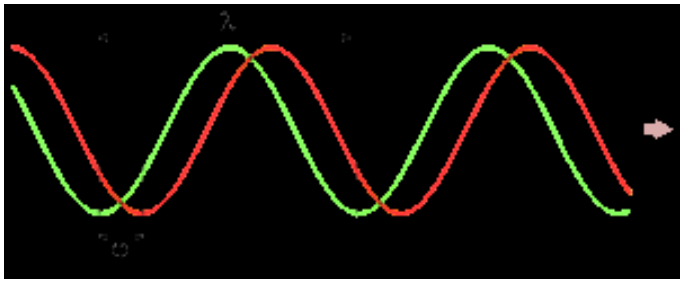
Vs



Lidar (phase difference)

Lidar (phase difference)





$$S(t) = A.\sin(w.t)$$

$$R(t) = B.\sin(w.(t - dt))$$

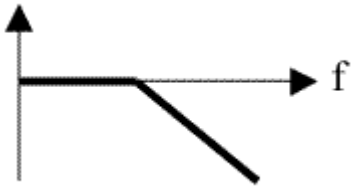
$$M(t) = S(t).R(t)$$

$$M(t) = A.B.\frac{\cos(w.dt) - \cos(2.w.t - w.dt)}{2}$$

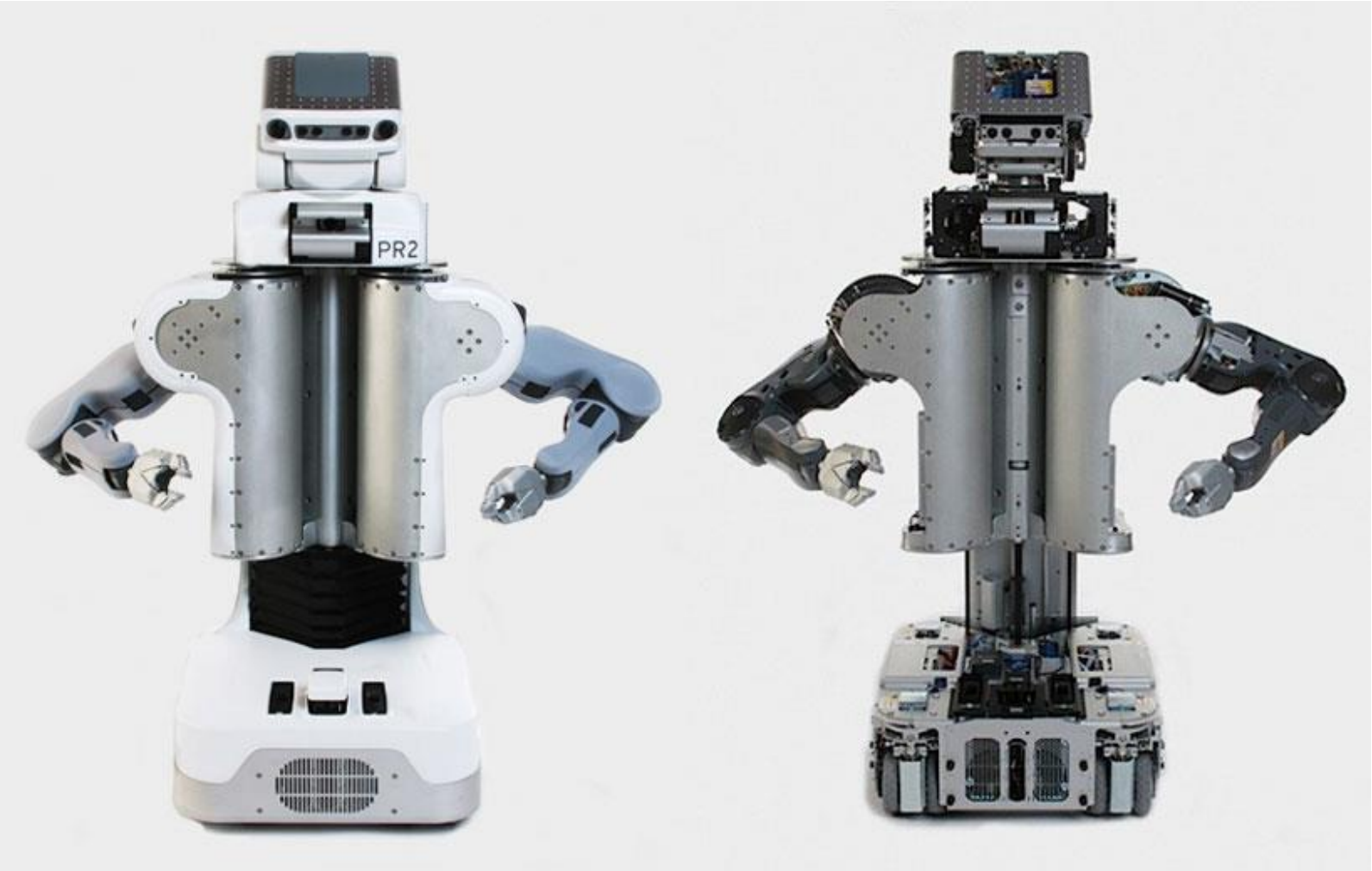
Constant for a certain distance

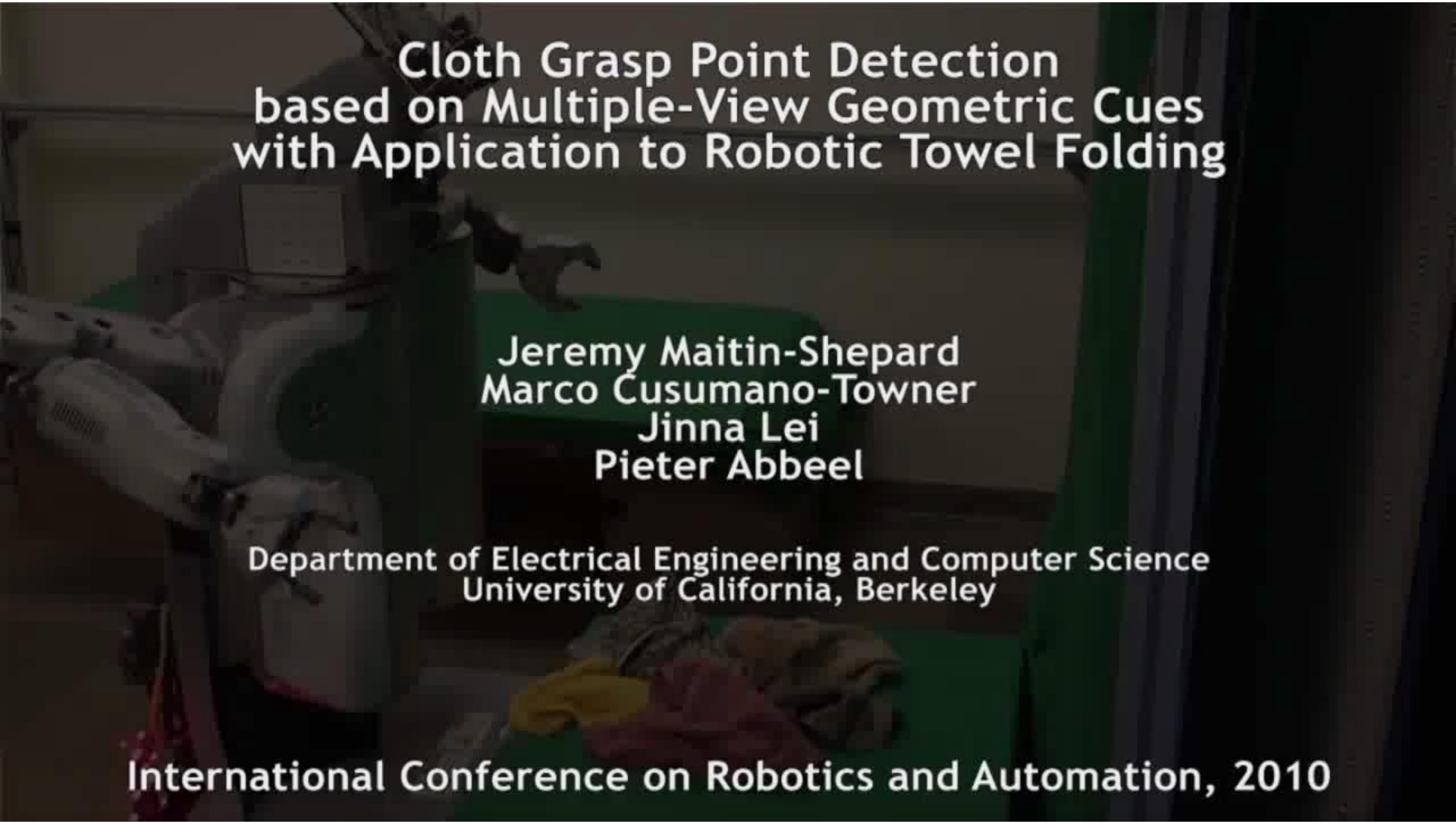
High Frequency

Extract value with low-pass filter



PR2





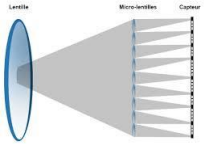
3D Cameras

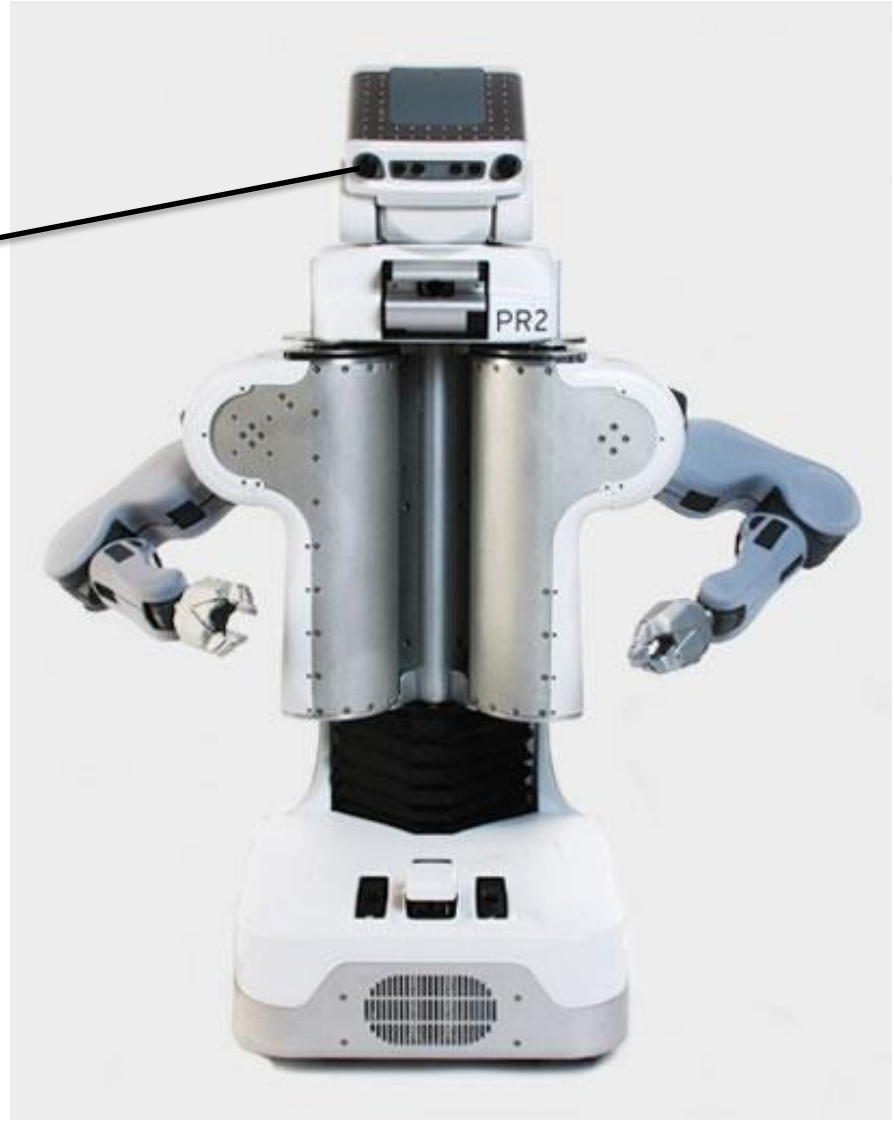
- Stereo camera

- Structured light camera

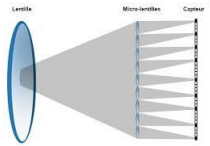
- Time of Flight (ToF) camera

- Plenoptic camera





3D Cameras

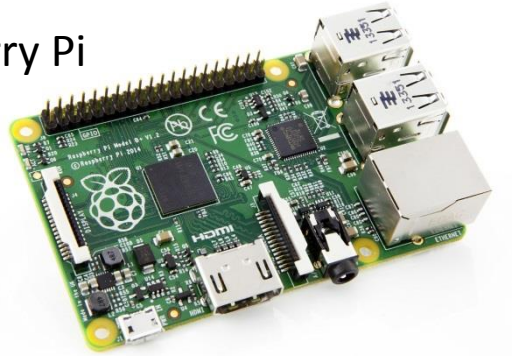


	Range	Witdh Resolution	Depth Resolution	Outside	Price
Stereo	O<--->O ~50m	~cm	~cm		
Structured light	~10m	~cm	~cm		
ToF	~50m	~mm	~cm		
Plenoptic	~1m	~mm	->10cm (not linear)		

Robotic process units (prototyping case study)

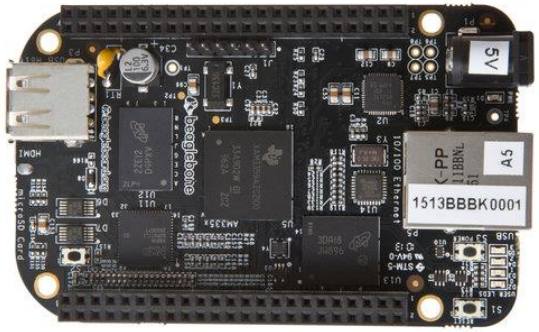
- Raspberry Pi

  B+



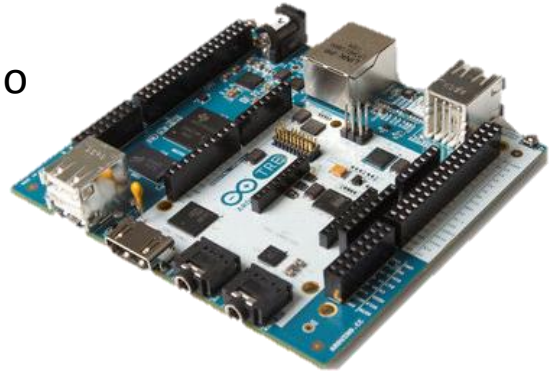
- BeagleBone

  Black



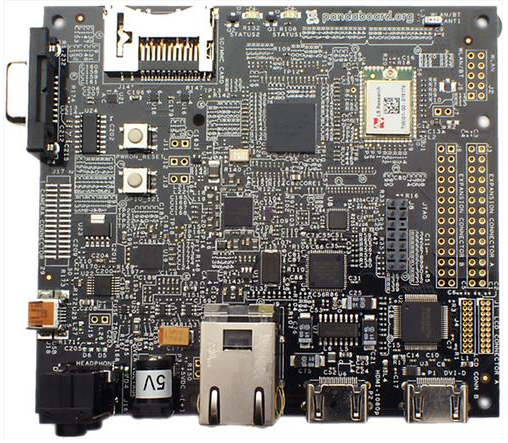
- Arduino

  Tre



- Panda Board

  ES

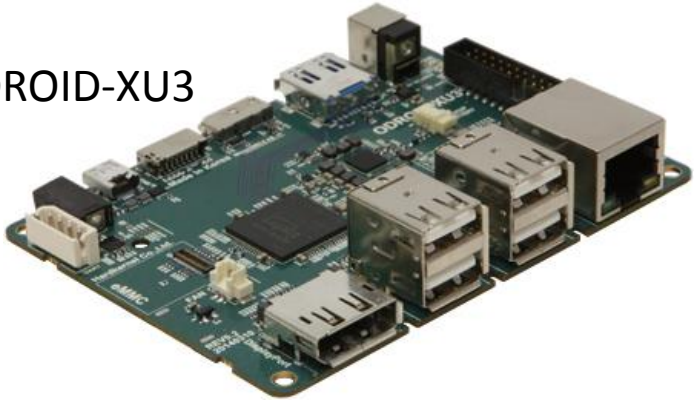


Robotic process units (prototyping case study)

- ODROID-U3

- UDOO

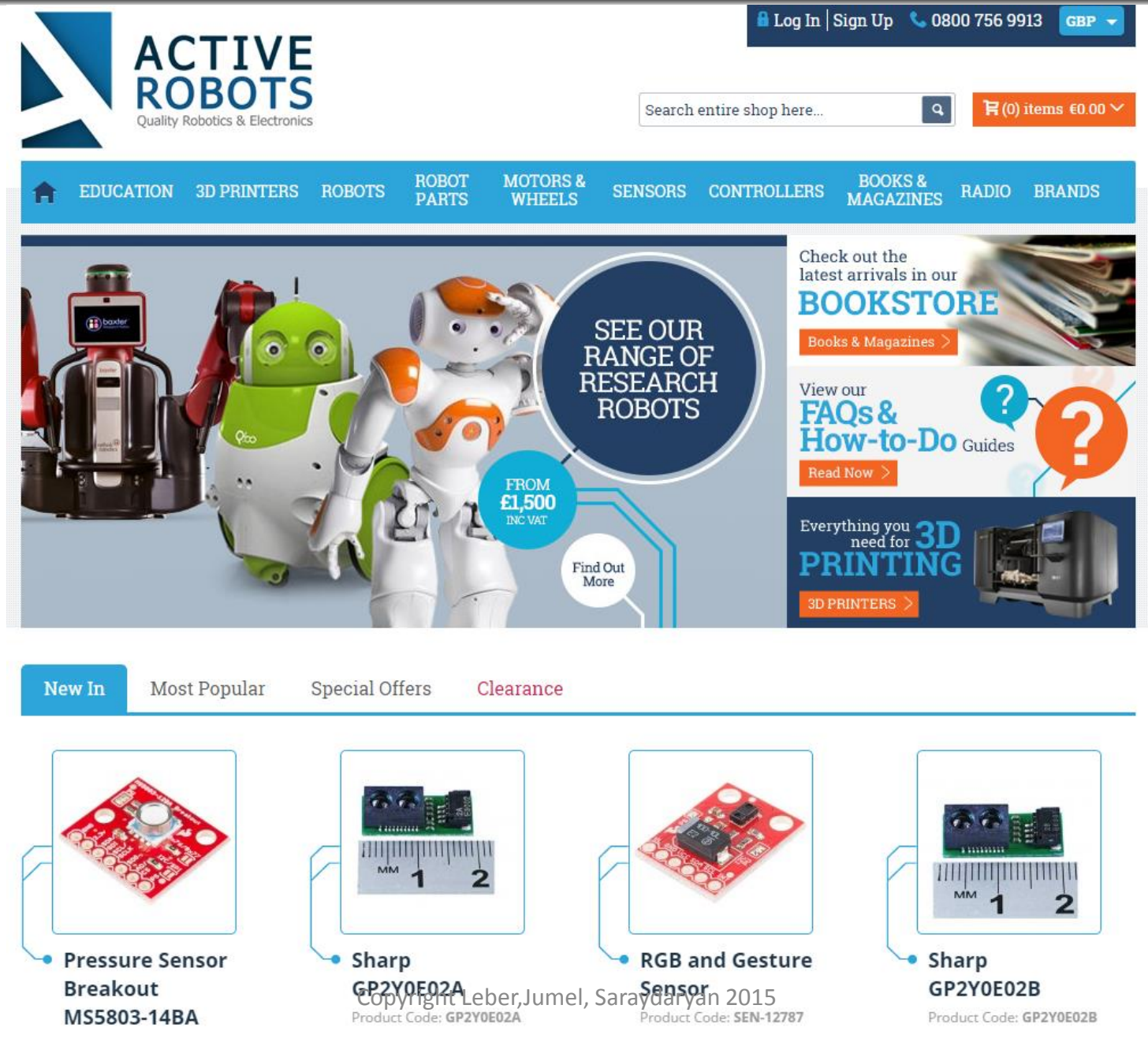
- ODROID-XU3



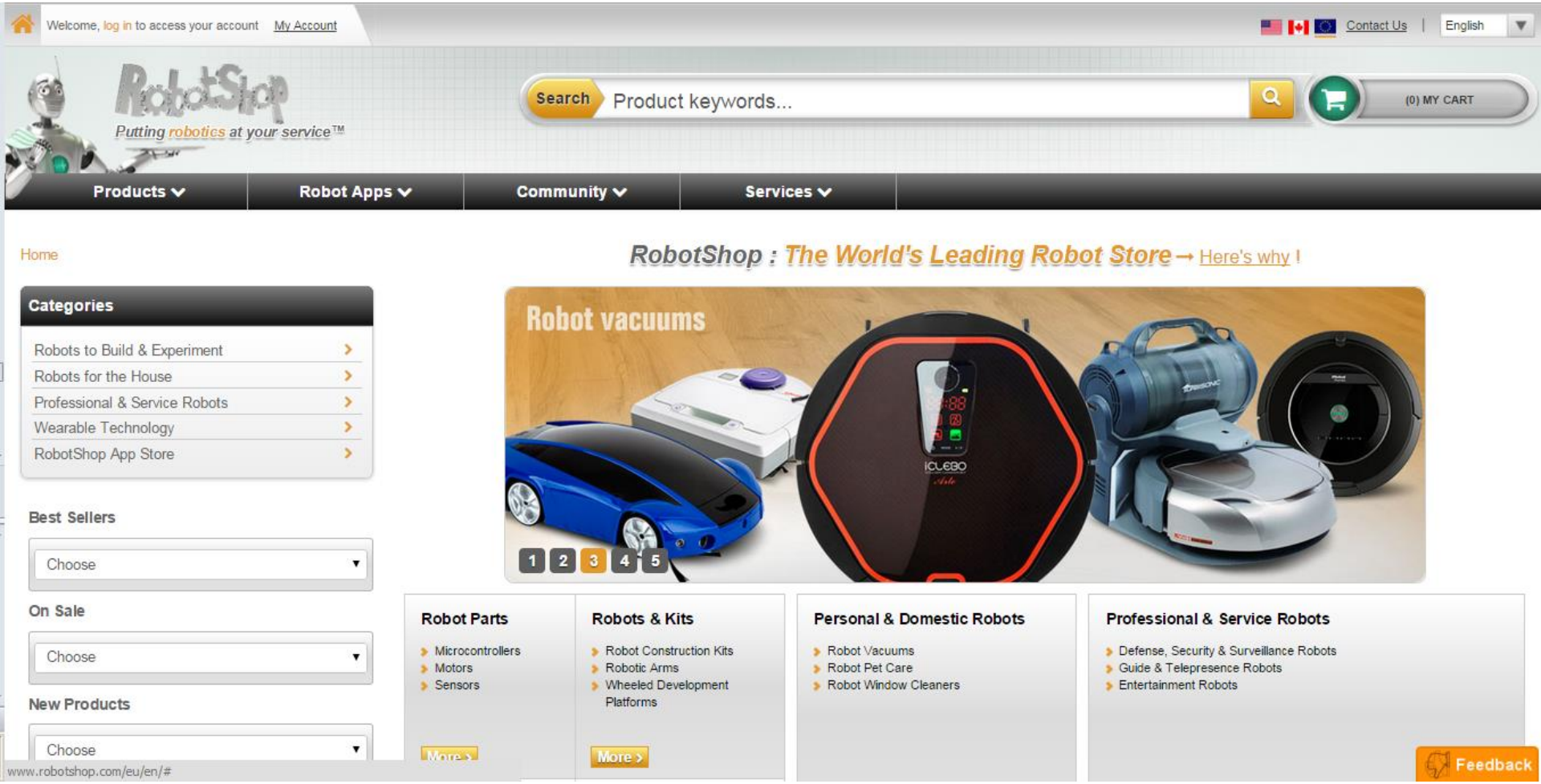
- WandBoard

Robotic process units (prototyping case study)

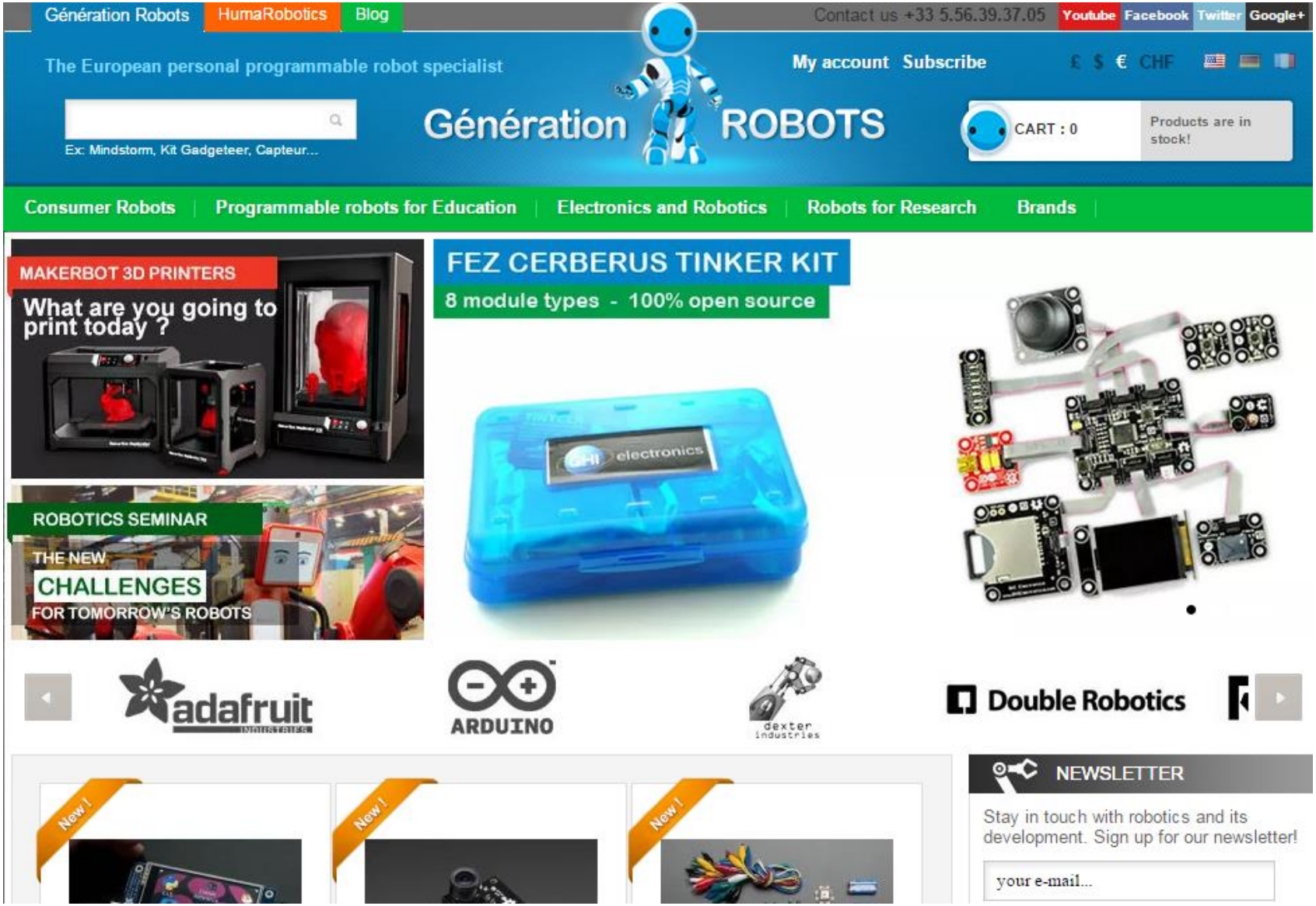
	Raspberry Pi – B+	Arduino Tree	BeagleBone Black	Panda Board ES	UDOO	Wand Board	ODROID U3	ODROID XU3
CPU	ARM710 700 Mhz	A-8 1 Ghz	A-8 1 Ghz	A-9 1,2Ghz	A-9 Quad 1Ghz	A-9 Quad 1Ghz	A-9 Quad 1,7Ghz	A-15 Quad 2Ghz
RAM	512 MB	512 MB	512 MB		1 GB	2 GB	2 GB	2 GB
Con.	+	++	++	+++	++++	++++	+	++++
Price	32 $	Coming soon	55 $	185 $	135 $	129 $	69 $	179 $



Hardware solutions for robotics of service



Hardware solutions for robotics of service



References

Images

http://www.intorobotics.com/inside-nao-next-gen-designed-for-future-home-robotic-applications/
http://home.roboticlab.eu/en/examples/sensor/ir_distance
http://www.generationrobots.com/en/content/52-localize-with-a-hokuyo-laser-range-finder
http://www.sensorsportal.com/HTML/DIGEST/april_06/Rotary_Encoder.htm
http://brooklynhobbies.com/hs-5735mg-digital-servo-hitec-35735
http://robocup.bowdoin.edu/public/software/nao/NaoQi/1.10/nao-1.10/doc/site_en/reddoc/hardware/FSR.html

Videos

https://www.youtube.com/watch?v=gy5g33S0Gzo

creative commons

Check Mark by aguycalledgary from The Noun Project
Gauge by David Waschbüsch from The Noun Project
Cocktail by Okan Benn from The Noun Project
Tools by Mister Pixel from The Noun Project
Lego by jon trillana from The Noun Project
book by Les vieux garçons from The Noun Project
Tree by Grant Taylor from The Noun Project
Caution by Jardson A. from The Noun Project
Globe by Richard Schumann from The Noun Project
Cloud by Edward Boatman from The Noun Project
Question by Jardson A. from The Noun Project
Robot by jon trillana from The Noun Project
Robot by Dylan Thompson from The Noun Project
Robot by Ricardo Moreira from The Noun Project
Robot by William Hollowell from The Noun Project
Sustainable Energy by Simple Icons from The Noun Project
Quote by irene hoffman from The Noun Project

CPE LYON ÉCOLE SUPÉRIEURE DE CHIMIE PHYSIQUE ÉLECTRONIQUE DE LYON

Raphael Leber
raphael.leber@cpe.fr

Fabrice Jumel
fabrice.jumel@cpe.fr

Jacques Saraydaryan
Jacques.saraydaryan@cpe.fr