Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

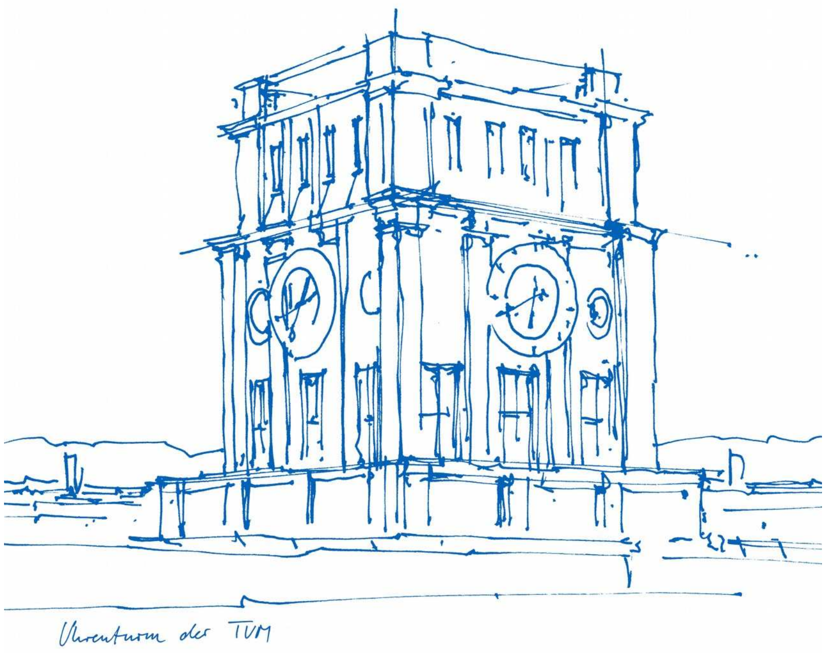
RoboCup@Home Practical course

Tutorials

Dr. Karinne Ramirez-Amaro
Dr. Emmanuel Dean
Dr. Pablo Lanillos
M.Sc. Roger Guadarrama
M.Sc. Constantin Uhde
Dr. Gordon Cheng

Technical University of Munich

Chair for Cognitve Systems



Definition of teams

1) **Hands-on tutorials:** introductory tutorials to get familiar with the equipment (robot & sensors) used for this course.

2) **Group definition tasks:** form groups according to the students' knowledge to address different challenges of the competition. The students designate a team leader.

3) **Development and test phase:** design and implementation of algorithms to solve the problems defined for your working team.

4) **Final phase:** test real scenarios on a mobile robot to evaluate the performance of the robots abilities.

Definition of teams

2) **Group definition tasks:** form groups according to the students' knowledge to address different challenges of the competition.

- Designate a **team leader**.

Responsibilities of the team leader:

- Coordinate the work of the group. Make sure that the work is correctly distributed.
- Be responsible for the key of the laboratory
- Direct communication with the supervisor(s)

Definition of topics

Category III: This category includes:

- following a human,
- indoor navigation in crowded environments,
- recognizing & grasping alike objects,
- find a calling person (waving or shouting), etc.
- deal with incomplete information

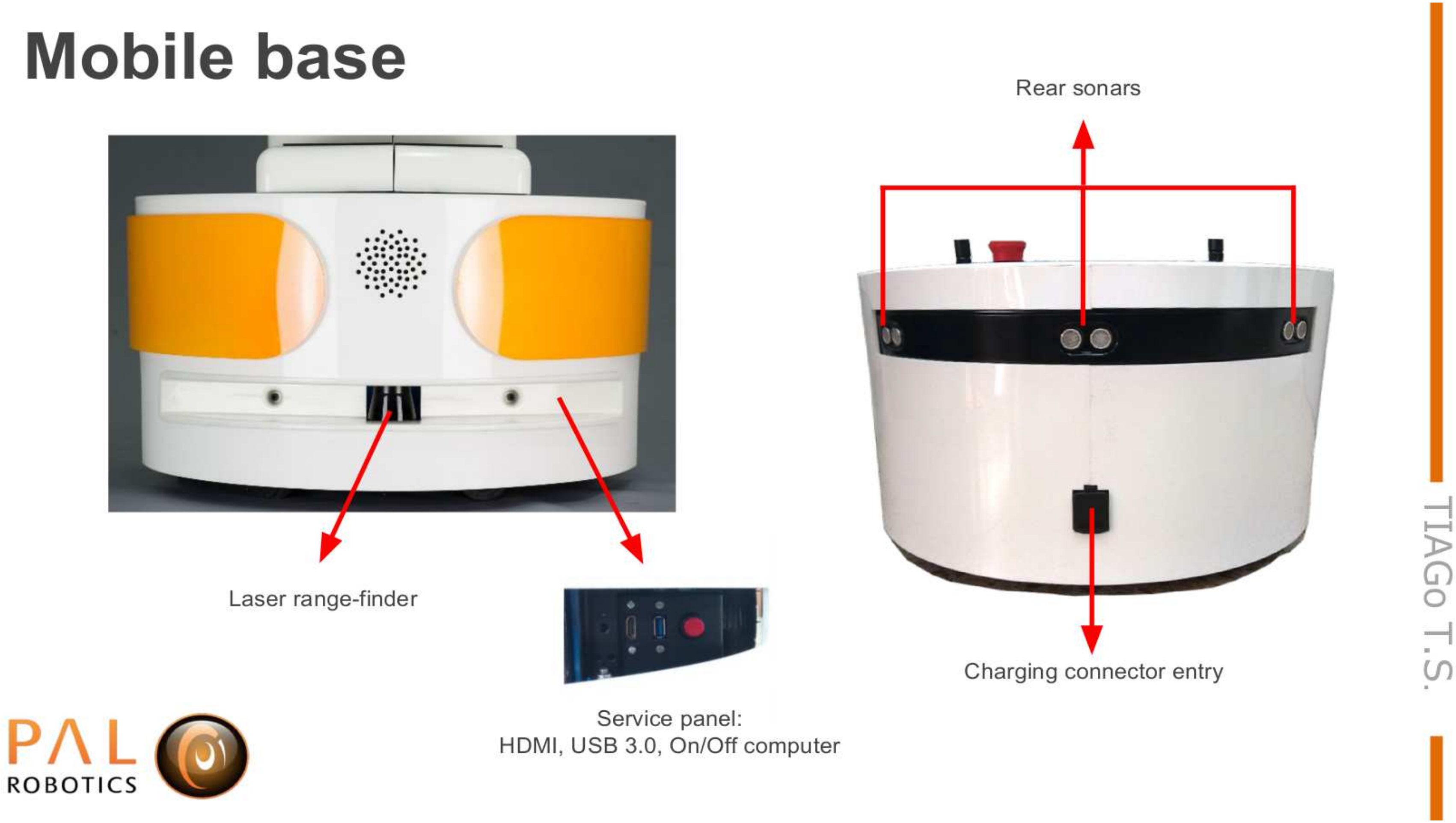
IMPORTANT: Define the strongest capabilities of your team: Control, navigation, perception, and/or learning.

Tutorial 5: TIAGo – getting started

Tutorial 5: TIAGo – getting started

©Pictures taken from PAL Robotics documentation

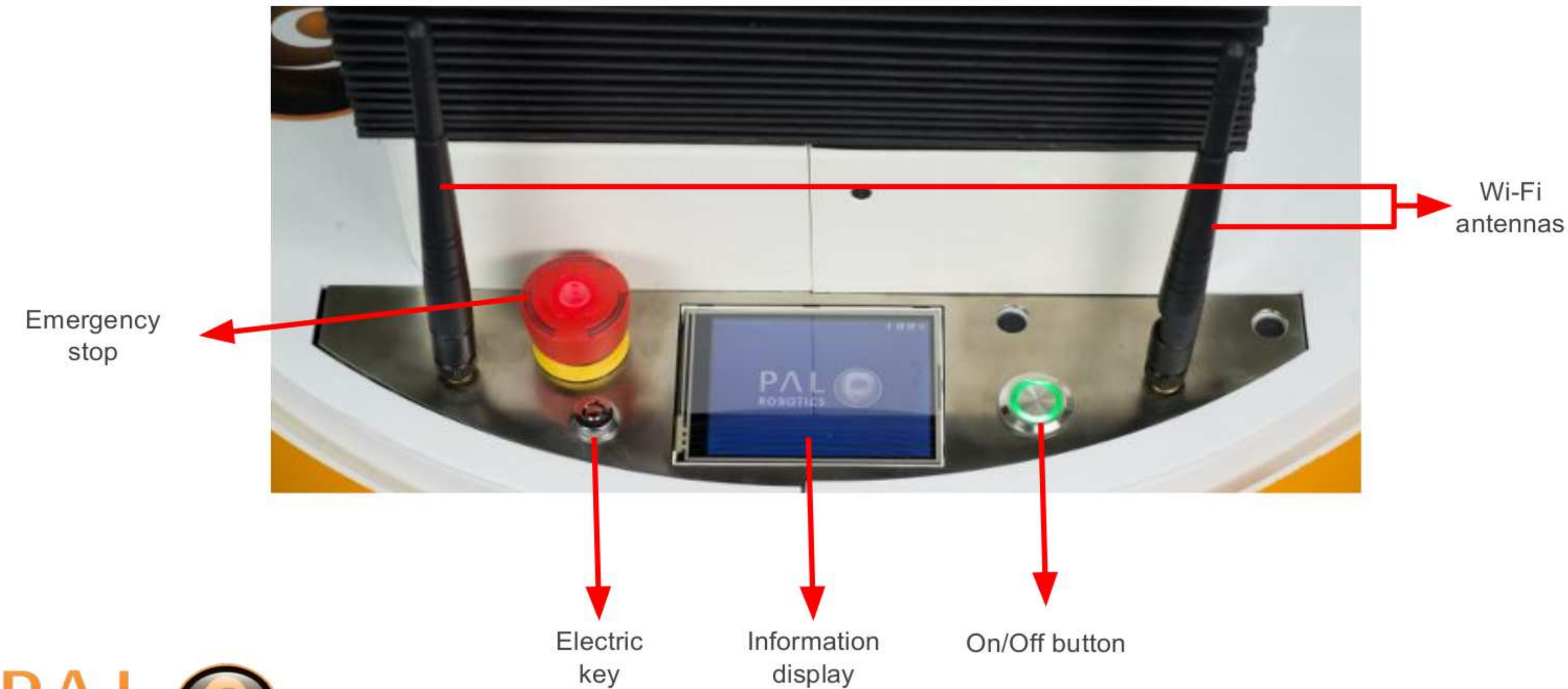
Tutorial 5: TIAGo – getting started



©Pictures taken from PAL Robotics documentation

Tutorial 5: TIAGo – getting started

User panel



PAL ROBOTICS

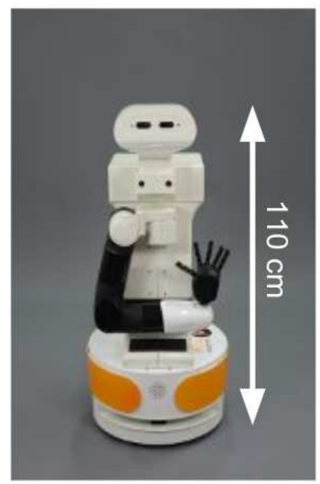
TIAGo T.S.

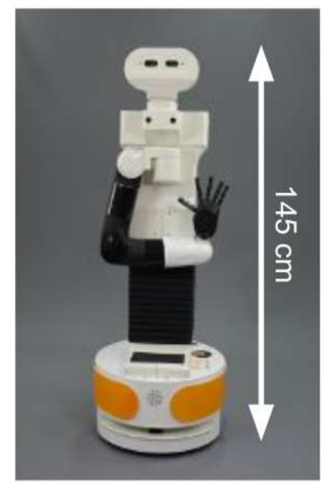
©Pictures taken from PAL Robotics documentation

Tutorial 5: TIAGo – getting started

Torso lifter

Stroke: 350 mm
Max speed: 50 mm/s





TIAGo T.S.

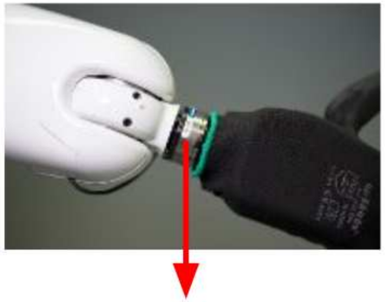
PAL ROBOTICS

©Pictures taken from PAL Robotics documentation

Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

Tutorial 5: TIAGo – getting started

Arm and end-effector



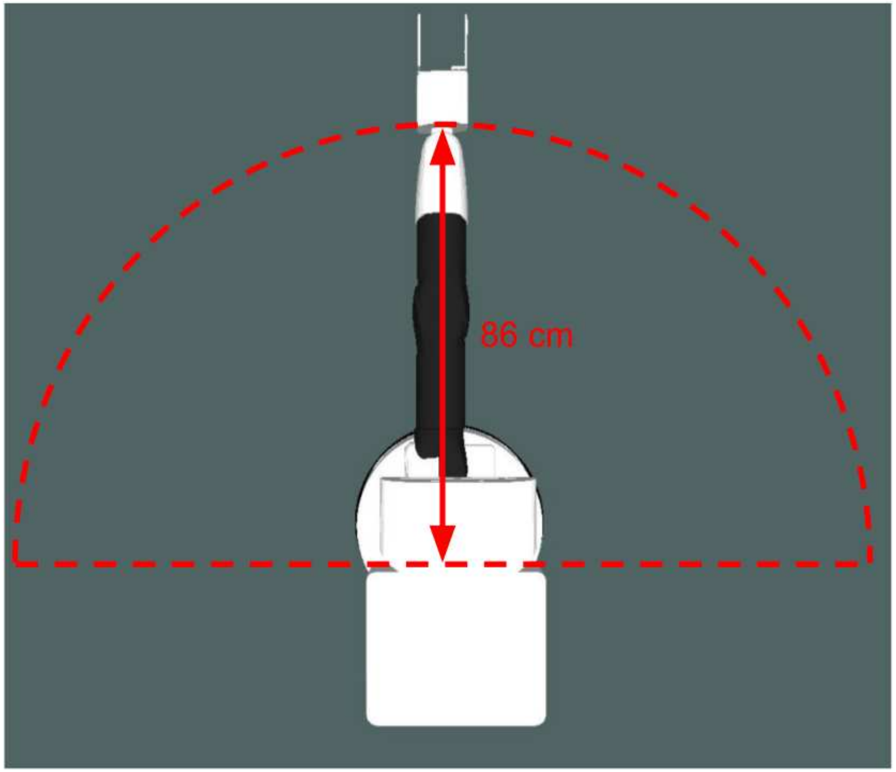
PAL ROBOTICS

TIAGo T.S.

©Pictures taken from PAL Robotics documentation

Tutorial 5: TIAGo – getting started

Arm reach

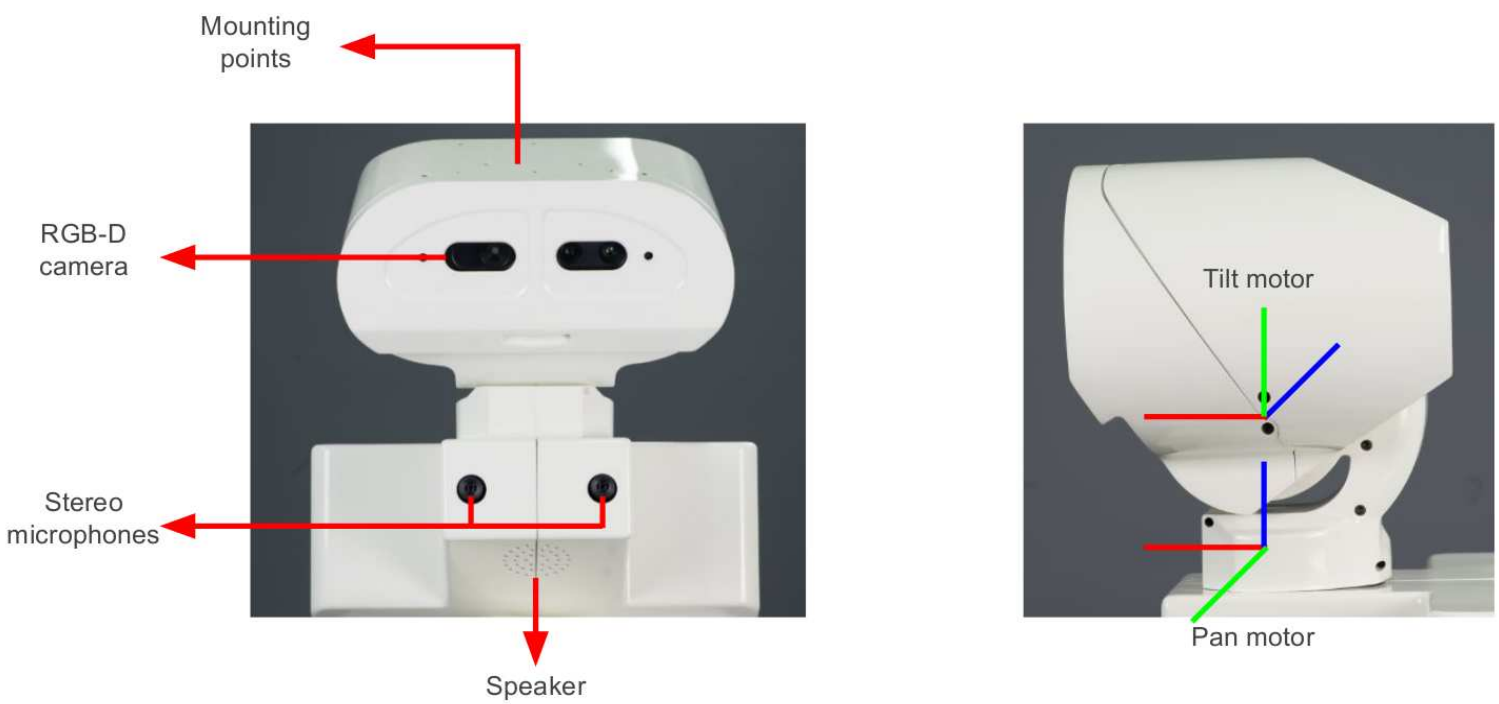


©Pictures taken from PAL Robotics documentation

Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

Tutorial 5: TIAGo – getting started

Head



PAL ROBOTICS

TIAGo T.S.

©Pictures taken from PAL Robotics documentation

Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

Tutorial 5: TIAGo – getting started

Start the robot

Turn the electric key

Release the emergency button

Press On button during 1 second

TIAGo T.S.

PAL ROBOTICS

©Pictures taken from PAL Robotics documentation

Tutorial 5: TIAGo – getting started

Charging TIAGo

- Use the charger supplied with TIAGo in order to charge the robot

Open the lid located in the rear part

Insert charging connector with metal lock facing up, push until you hear a ‘click’

The duration of the charging is about 4 h for 1 battery completely discharged, and about 8 h for 2 batteries

Insert charging connector with metal lock facing up, push until you hear a ‘click’

PAL ROBOTICS

TIAGo T.S.

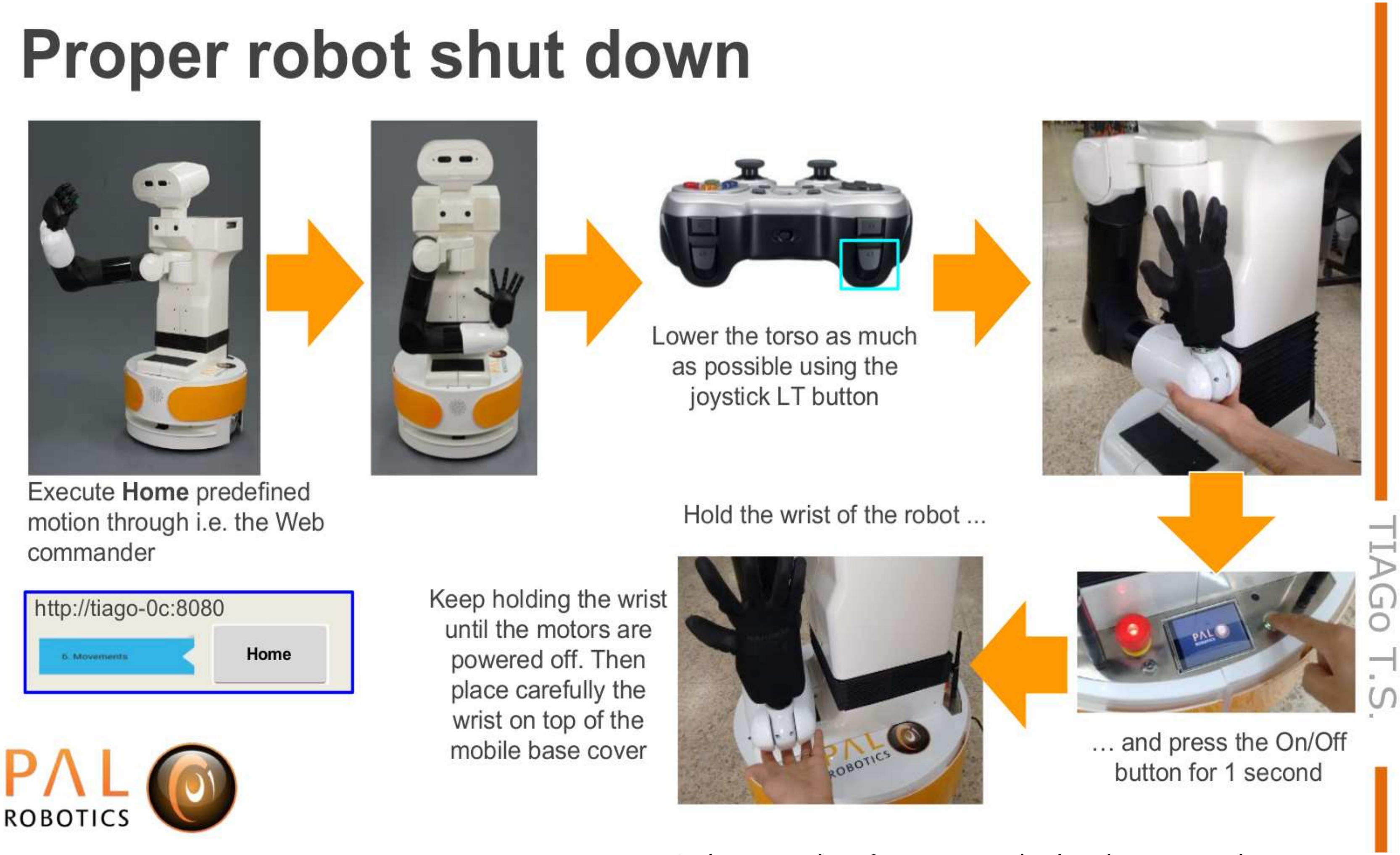
©Pictures taken from PAL Robotics documentation

Tutorial 5: TIAGo – Safety measures

Safety measures

Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

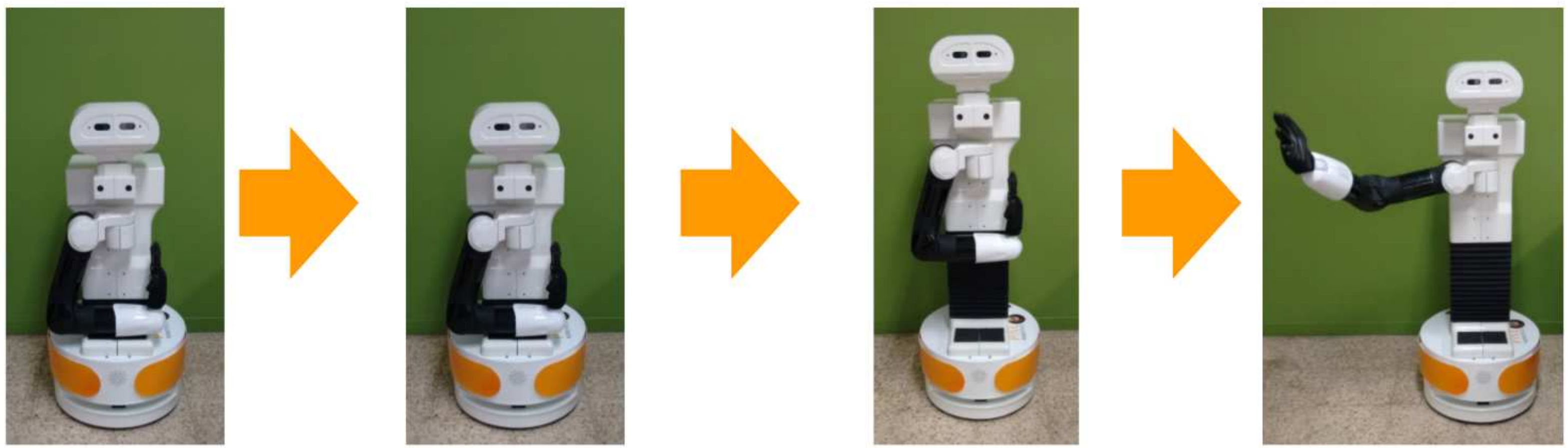
Tutorial 5: TIAGo – Safety measures



©Pictures taken from PAL Robotics documentation

Tutorial 5: TIAGo – Safety measures

How to get the arm out of Home safely



Get the arm out of collision:

Raise the torso to its maximum height with the joystick

Execute the movement **unfold_arm** through i.e. the Web commander

PAL ROBOTICS

©Pictures taken from PAL Robotics documentation

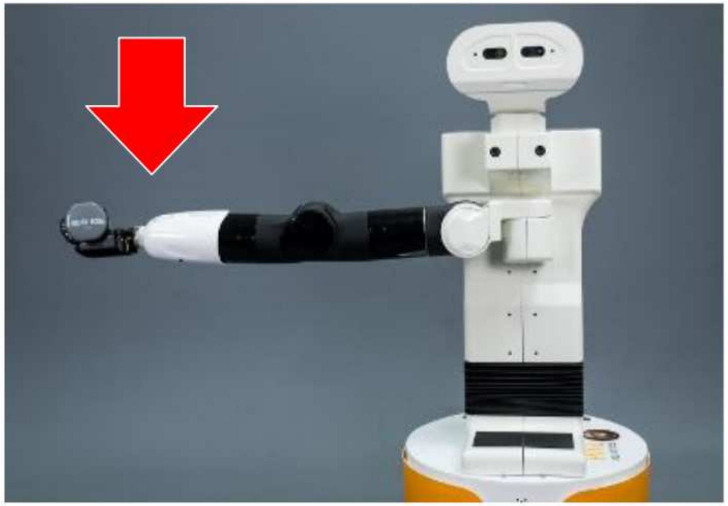
TIAGo T.S.

Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

Tutorial 5: TIAGo – Safety measures

Measures to prevent the robot falling (I)

- The robot has been designed to be stable even when the arm is holding its maximum payload in its most extreme kinematic configuration
- Nevertheless, some measures need to be respected in order to avoid the robot falling



Do not apply external forces to the arm when it is extended and holding a weight

TIAGo T.S.

©Pictures taken from PAL Robotics documentation

Tutorial 5: TIAGo – Safety measures

Measures to prevent the robot falling (II)

Do not navigate at **high speeds** with the arm extended and holding a weight specially when the torso is extended

TIAGo T.S.

©Pictures taken from PAL Robotics documentation

Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

Tutorial 5: TIAGo – Safety measures

Measures to prevent the robot falling (III)

- Do not navigate on floors with slope higher than 6º
- And in that case navigate with the torso at its lowest level and with the arm folded, i.e. in **Home** configuration

PAL ROBOTICS

TIAGo T.S.

©Pictures taken from PAL Robotics documentation

Tutorial 5: TIAGo – Safety measures

Measures to achieve a safe navigation

- When carrying a weight:

It is highly recommended to navigate with the arm folded and the torso at low extension, like in the **Home** configuration

- Reduces the footprint of the robot and lowers the probability to collide with the environment with the arm
- Ensures the stability of the robot

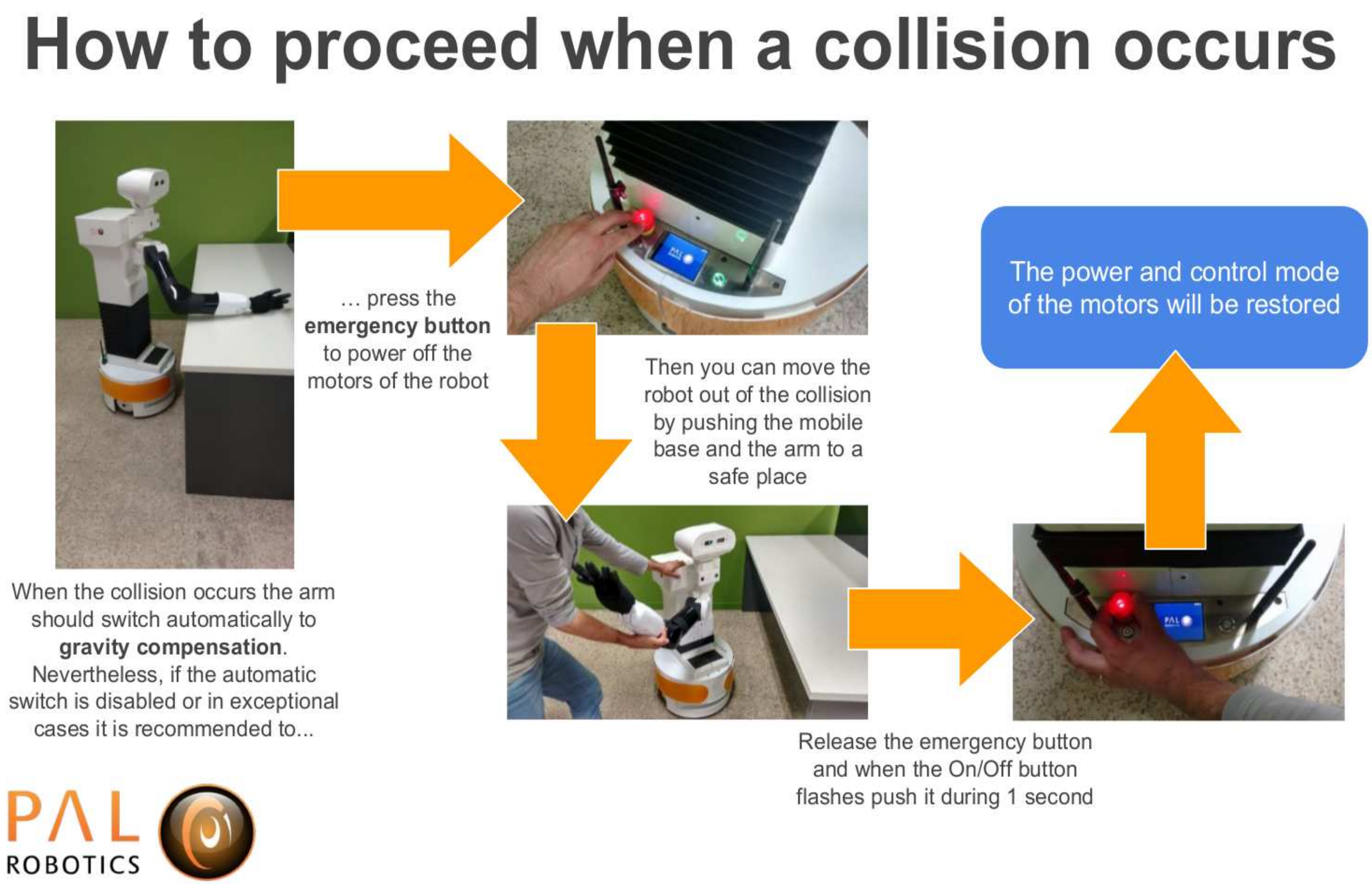
PAL ROBOTICS

TIAGo T.S.

©Pictures taken from PAL Robotics documentation

Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

Tutorial 5: TIAGo – Safety measures



TIAGo T.S.

©Pictures taken from PAL Robotics documentation

Tutorial 5: TIAGo – Software

Software architecture overview

- Operating system:

  Ubuntu 14.04 LTS 64-bit

  Xenomai real-time framework → real-time control software

- Robotics middleware:

  Orocos 2.8

  ROS Indigo

  PAL dubnium → ROS packages developed by PAL Robotics

Tutorial 5: TIAGo – Software

ROS packages

- **ROS** Indigo packages:

  Installation path: **/opt/ros/indigo**

- **PAL** dubnium packages:

  Installation path: **/opt/pal/dubnium**

- **User** packages:

  Installation path: **/home/pal/deployed_ws**

Sub-folder	Description
bin	nodes (executables)
include	package header files
lib	package dynamic libraries and python files
share	packages cmake, launch and config files
etc	other files

All the ROS software in the robot is executed with the user **pal**

PAL ROBOTICS

TIAGo T.S.

©Pictures taken from PAL Robotics documentation

Tutorial 5: TIAGo – Deploy new software in the robot

Connect to the robot ROS topics from the remote PC

On you local PC execute following commands:

1) export ROS_MASTER_URI=http://tiago-24c:11311

2) export ROS_IP=*ip_of_your_PC*

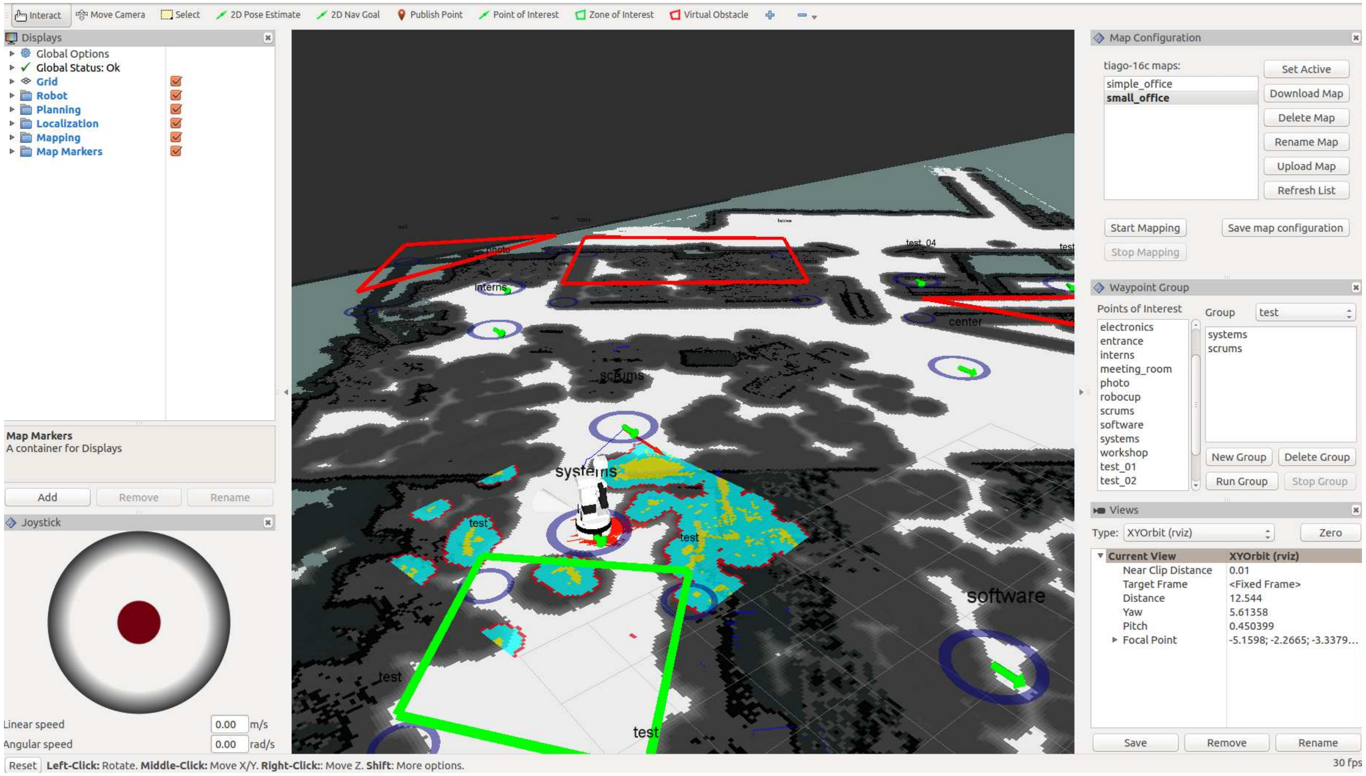
Tutorial 5: System integration on TIAGo

- **Finish all the tutorials of TIAGo**
- **Get really familiar with the Navigation package of TIAGo**

Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

Tutorial 5: System integration

Exercise 1: Build a map of one of the floors from ICS. You can choose between the 2nd floor or the basement. The robot should be able to navigate within these rooms.



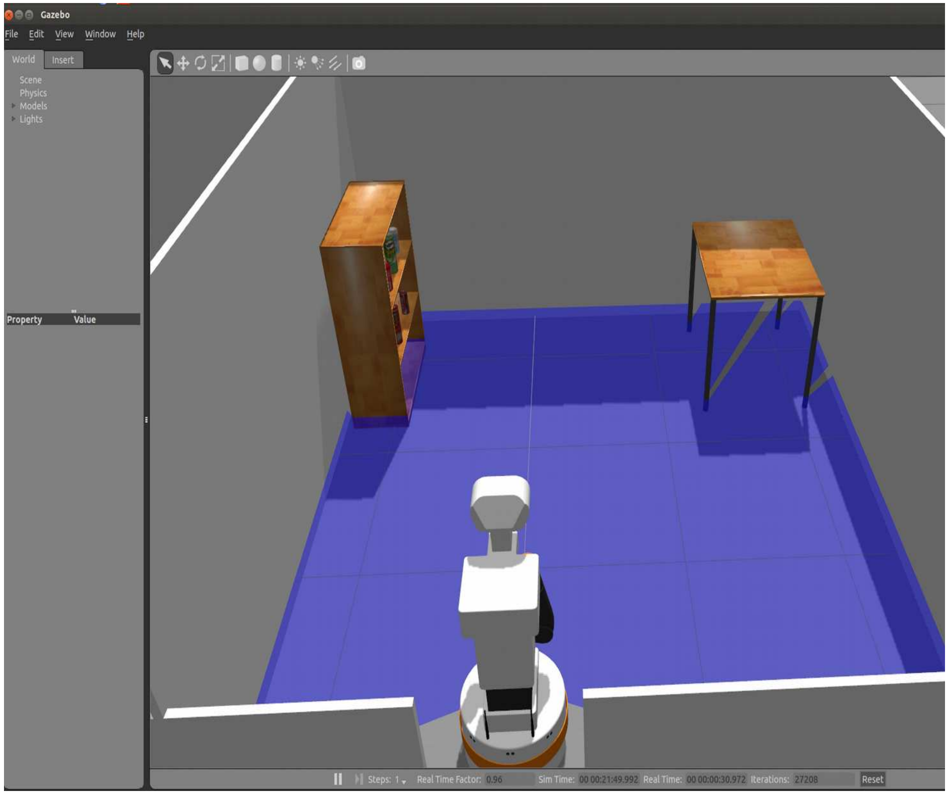
Tutorial 5: System integration

Exercise 2

- **Object Recognition**
- **Robot Manipulation**
- **Mapping and Navigation**

Tutorial 5: System integration

Exercise 2: The robot needs to move to the shelf grab one item and move it to the table.



Perception:
- 3D object localization
- Object detection
- Identify the shelf/table

Robot manipulation:
- Command the robot arm to a desired pose
- Grasp the object

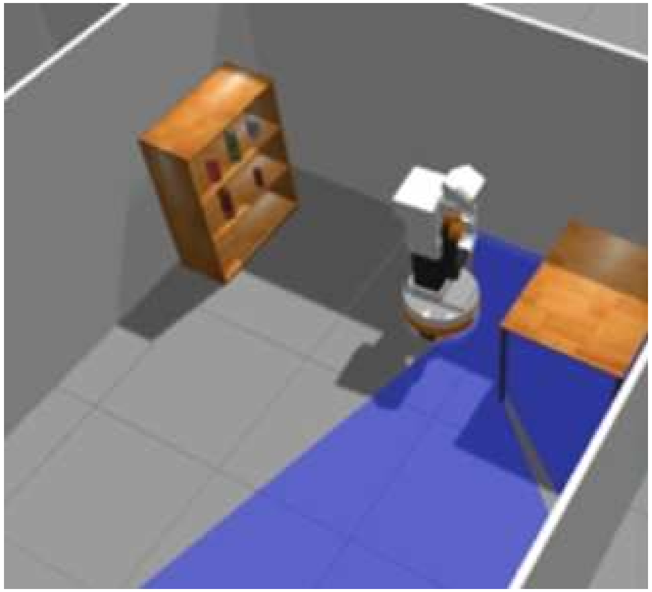
Navigation:
- Move to the shelf and avoid collisions
- Move to the table

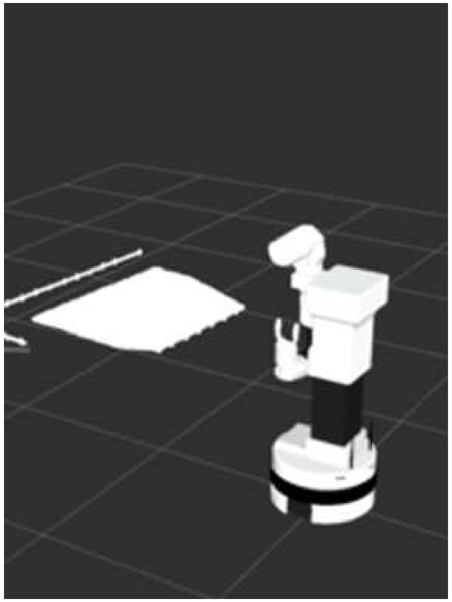
Tutorial 5: System integration

- **Object Recognition**

Perception

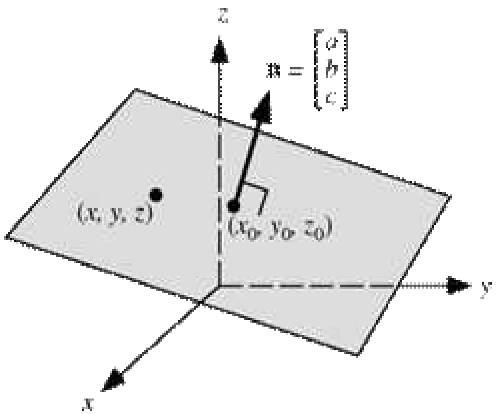
From plane segmentation to table computation





`pcl::ModelCoefficients::Ptr`

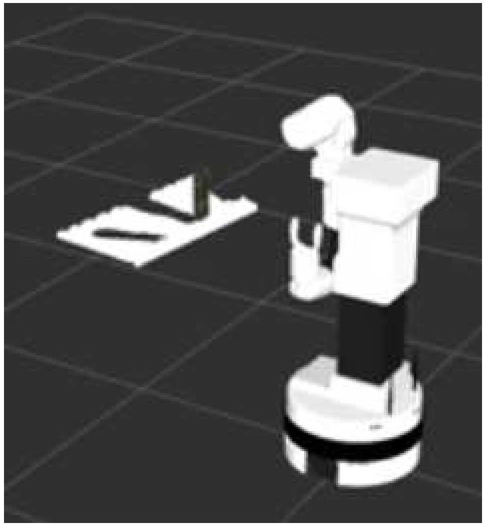
$$ax + by + cy + d = 0$$



Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

Perception

Segmenting the objects on the bookshelf

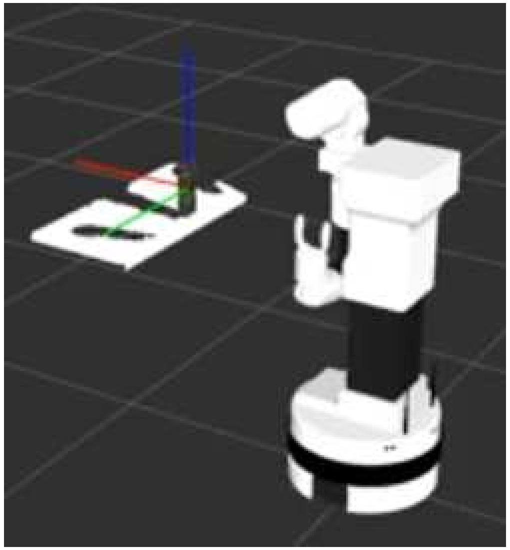


- Transform to frame if needed (pcl_ros::transformPointCloud)
- Filter limits (Passthrough)
- Downsample (VoxelGrid)
- Remove the planes (pcl::SACSegmentation)
- Find the cylinders (pcl::SACSegmentationFromNormals)
- Remove outliers (pcl::StatisticalOutlierRemoval)
- Publish the list of poinclouds or the one needed

Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

Perception

What else we can get from the Point Cloud data?



- Shape coefficients: cylinder [point_on_axis (x,y,z), axis_direction, (x,y,z) radius]
- 3D centroid
- Orientation
- Bounding box: min (x,y,z) and max (x,y,z) | centre (x0,y0,z0)
- Colored Texture: VoxelGrid.setDownsampleAllData(true);
- Descriptors
- Distance to the object

In order to grasp we need the **pose** of the object.

Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

Perception

Tips for visualization in rviz:

- One object: send the PointCloud2
- Several objects: use markers (visualization_msgs/MarkerArray.msg)

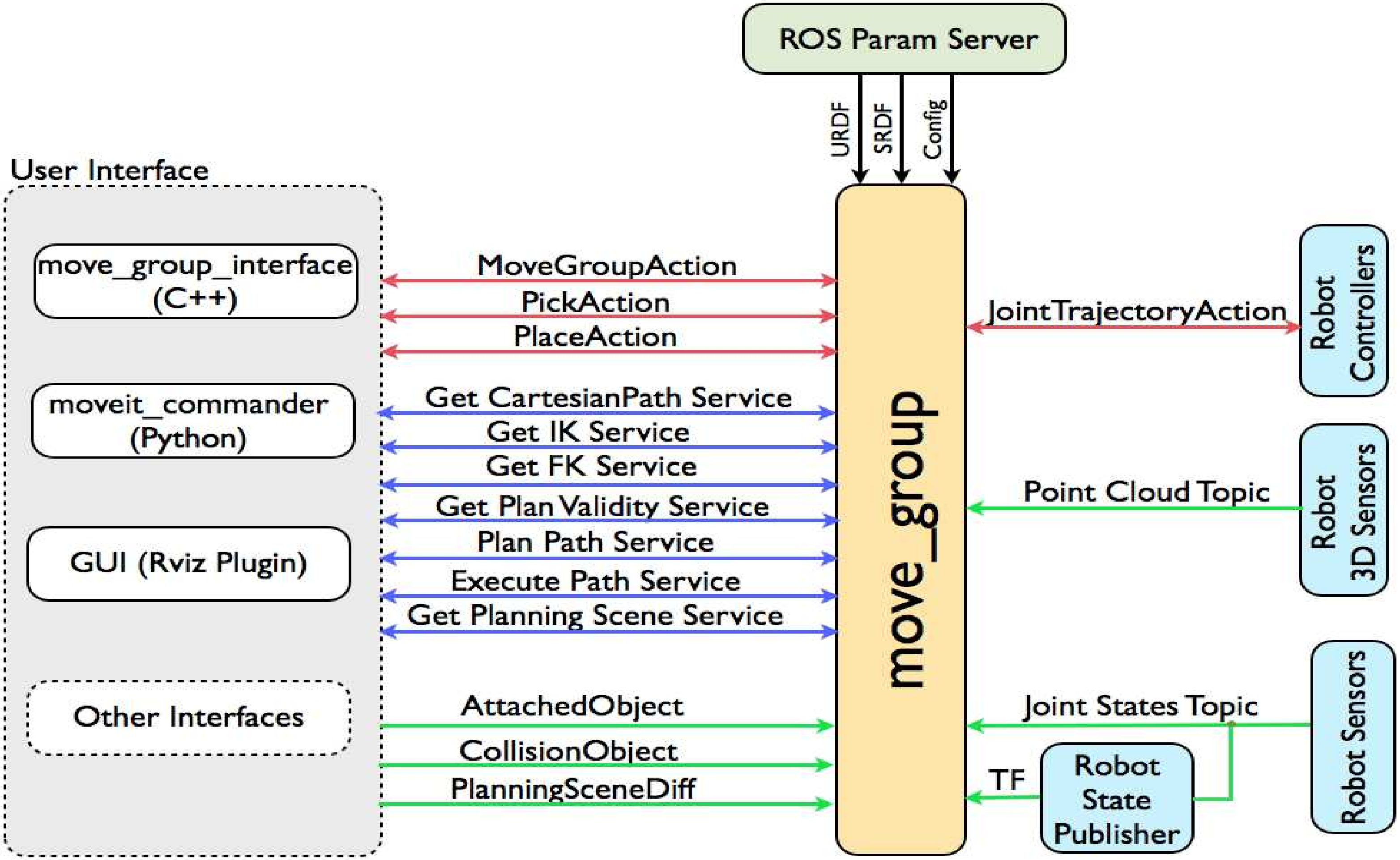
```
#ifdef PUBLISH_MARKERS
visualization_msgs::Marker ObjectClustering::getCloudMarker(const pcl::PointCloud<PointType>::Ptr  cloud, int id)
{
  //create the marker
  visualization_msgs::Marker marker;
  marker.header.frame_id = processing_frame_;
  marker.header.stamp = ros::Time();
  marker.action = visualization_msgs::Marker::ADD;
  marker.lifetime = ros::Duration(5);
  marker.ns = "segmentation";
  marker.id = id;
  marker.pose.orientation.w = 1;

  marker.type = visualization_msgs::Marker::POINTS;
  marker.scale.x = 0.002;
  marker.scale.y = 0.002;
  marker.scale.z = 1.0;

  marker.color.r = ((double)rand())/RAND_MAX;
  marker.color.g = ((double)rand())/RAND_MAX;
  marker.color.b = ((double)rand())/RAND_MAX;
  marker.color.a = 1.0;

  for(size_t i=0; i<cloud->size(); i++) {
    geometry_msgs::Point p;
    p.x = (*cloud)[i].x;
    p.y = (*cloud)[i].y;
    p.z = (*cloud)[i].z;
    marker.points.push_back(p);
  }

  return marker;
}
#endif
```

Tutorial 5: System integration

- **Object Recognition**
- **Robot Manipulation**

Chair for Cognitive Systems
Department of Electrical and Computer Engineering
Technische Universität München

MoveIt! Motion planner



Grasping with MoveIt! planner

1) Define object position (From perception node)
2) Feed the object position and geometry restrictions to the move_group node.
3) Request a planning service.
4) Wait for response.
5) Execute the motion.

6) Check MoveIt! Tutorial from http://wiki.ros.org/Robots/TIAGo/Tutorials

Tutorial 5: System integration

- **Object Recognition**
- **Robot Manipulation**
- **Mapping and Navigation**

Navigation and Mapping

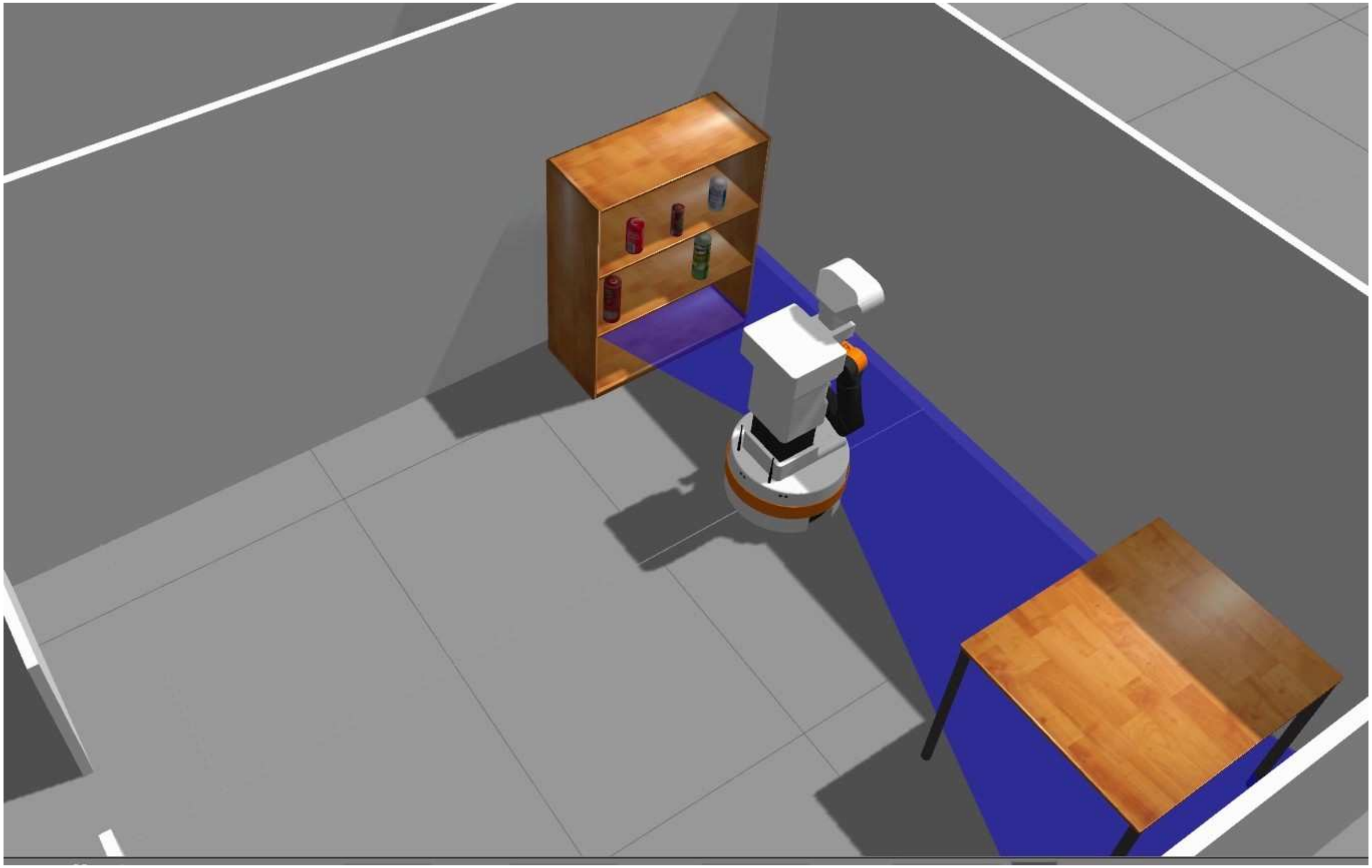
Hints:

- Generate a map for the testing environment.
- Use the obtained map to do navigation.
- Use MoveBaseGoal to move the robot with respect to the base_link.

If you want to navigate in the unknown environment (without previously building a map) you can check another SLAM packages :

http://wiki.ros.org/hector_slam or http://wiki.ros.org/slam_karto

Tutorial 5: System integration



Thank you

robocup.atHome.ics@tum.de

www.ics.ei.tum.de