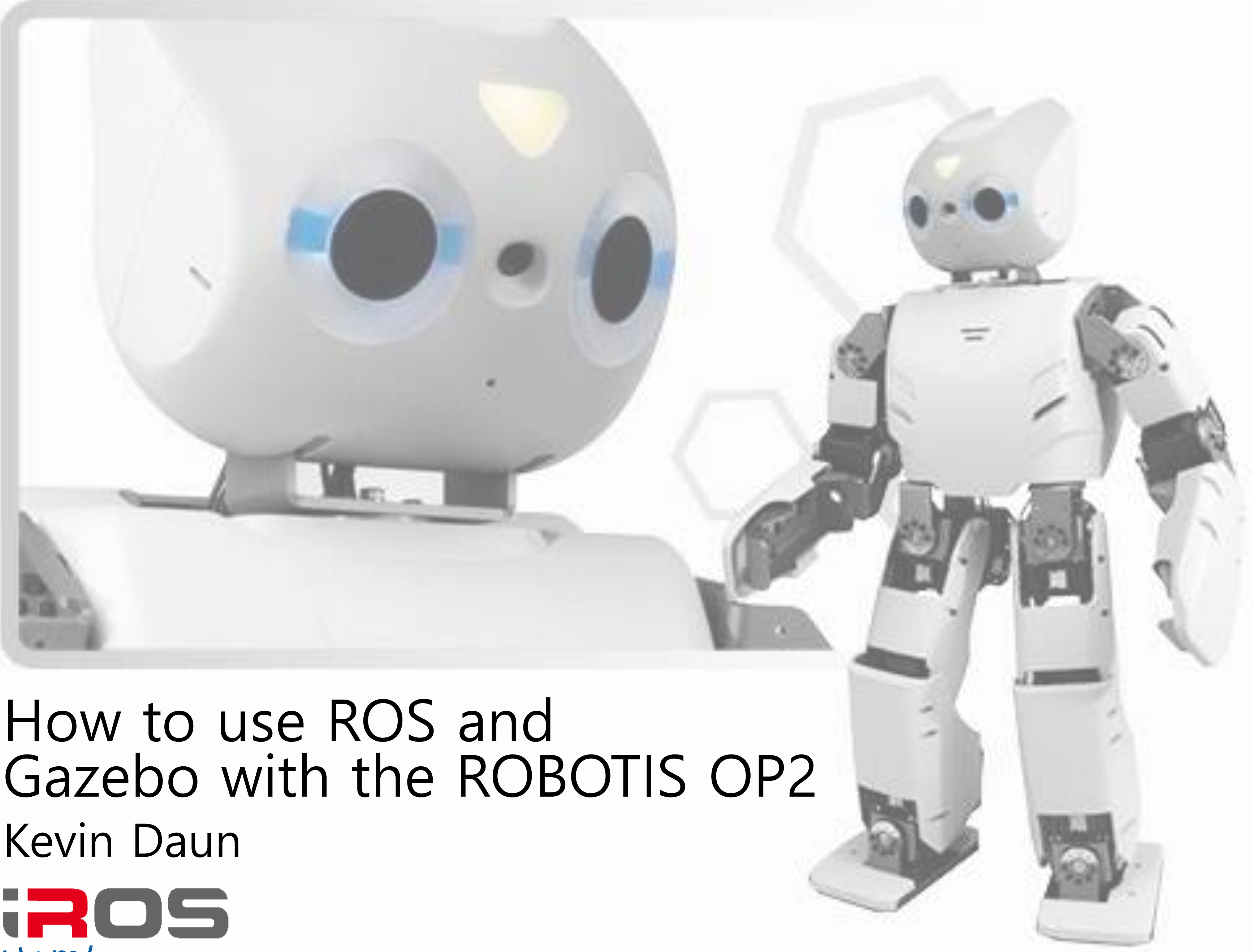


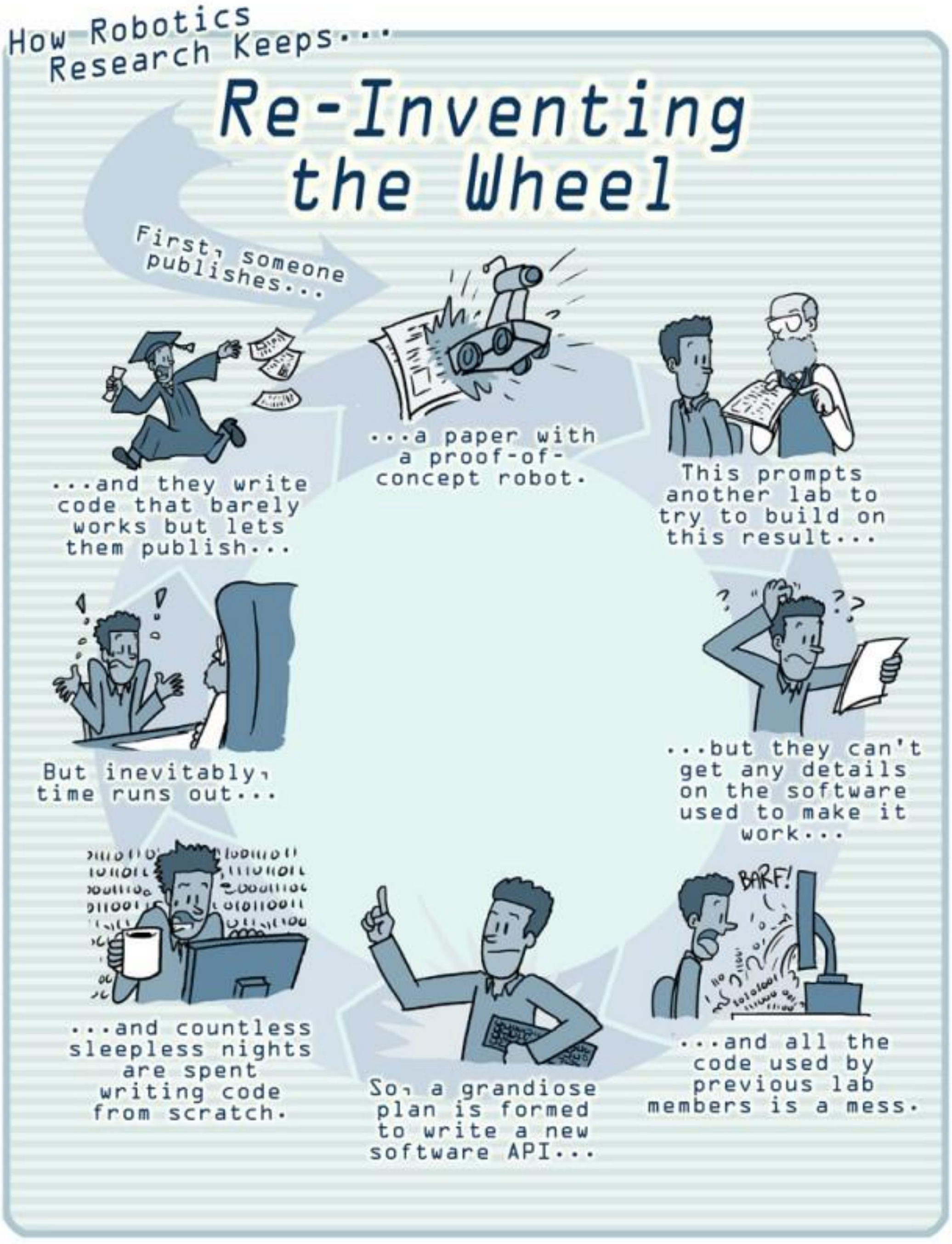
How to use ROS and Gazebo with the ROBOTIS OP2

Kevin Daun

iROS
Hamburg 2015

Before we start…

- Who has used Linux before?
- Who has used ROS before?
- Who has used the Darwin OP/ ROBOTIS OP2 before?


How Robotics Research Keeps...
Re-Inventing the Wheel
First, someone publishes...
...a paper with a proof-of-concept robot.
This prompts another lab to try to build on this result...
...but they can't get any details on the software used to make it work...
BARF!
...and all the code used by previous lab members is a mess.
So, a grandiose plan is formed to write a new software API...
...and countless sleepless nights are spent writing code from scratch.
But inevitably, time runs out...
...and they write code that barely works but lets them publish...

By Jorge Cham, Ph.D. Comics.

Contents

- What will you learn today?
  - What is ROS? What is Gazebo?
  - What are the key components?
  - Which basics are important to know?
  - How does the Robotis OP2 ROS Interface look like? How do I use it?
  - How do I implement a ball follower with the Robotis OP2 using ROS? → Hands-on project

Contents

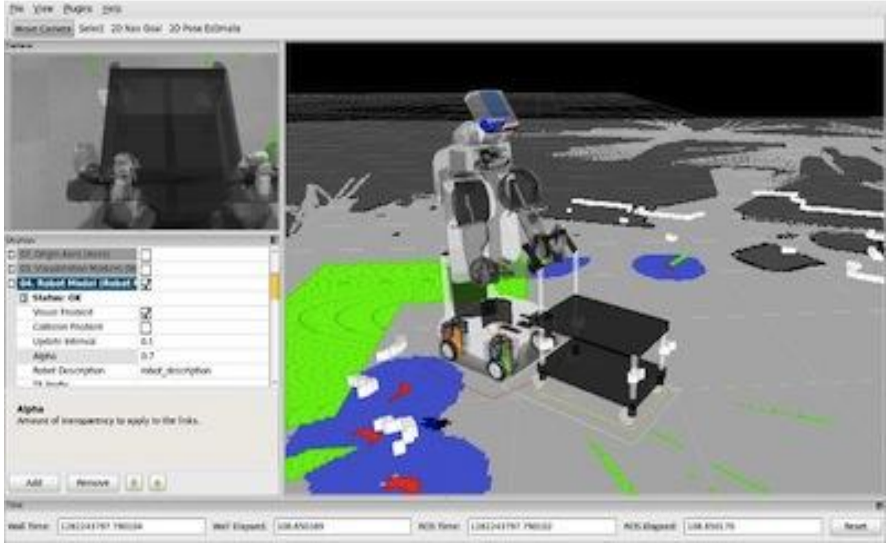
- What will you learn today?
  - **What is ROS? What is Gazebo?**
  - What are the key components?
  - Which basics are important to know?
  - How does the Robotis OP2 ROS Interface look like? How do I use it?
  - How do I implement a ball follower with the Robotis OP2 using ROS? → Hands-on project

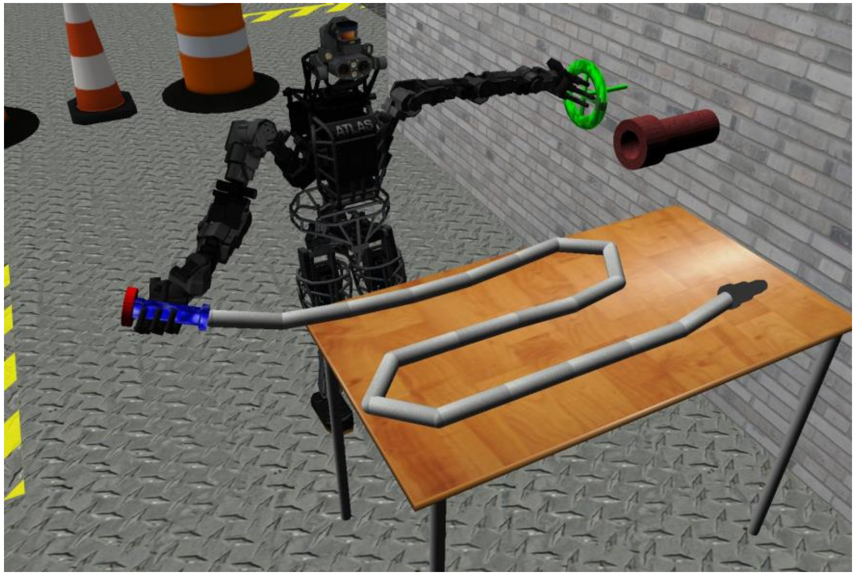
ROS - Robot Operating System

- What is ROS?
  - A collection of
    - Software libraries
    - Tools
    - Conventions
- What is ROS not?
  - An operating system like e.g. Ubuntu or Windows
- Why use it?
  - High reusability/ Easy collaboration
  - Collection of powerful development tools
  - Popular in academics and industry

ROS Core Elements

- Communication Structure
- Tools
  - Rviz
  - Rqt
- Integration
  - Gazebo
  - OpenCV
  - PCL
  - MoveIt





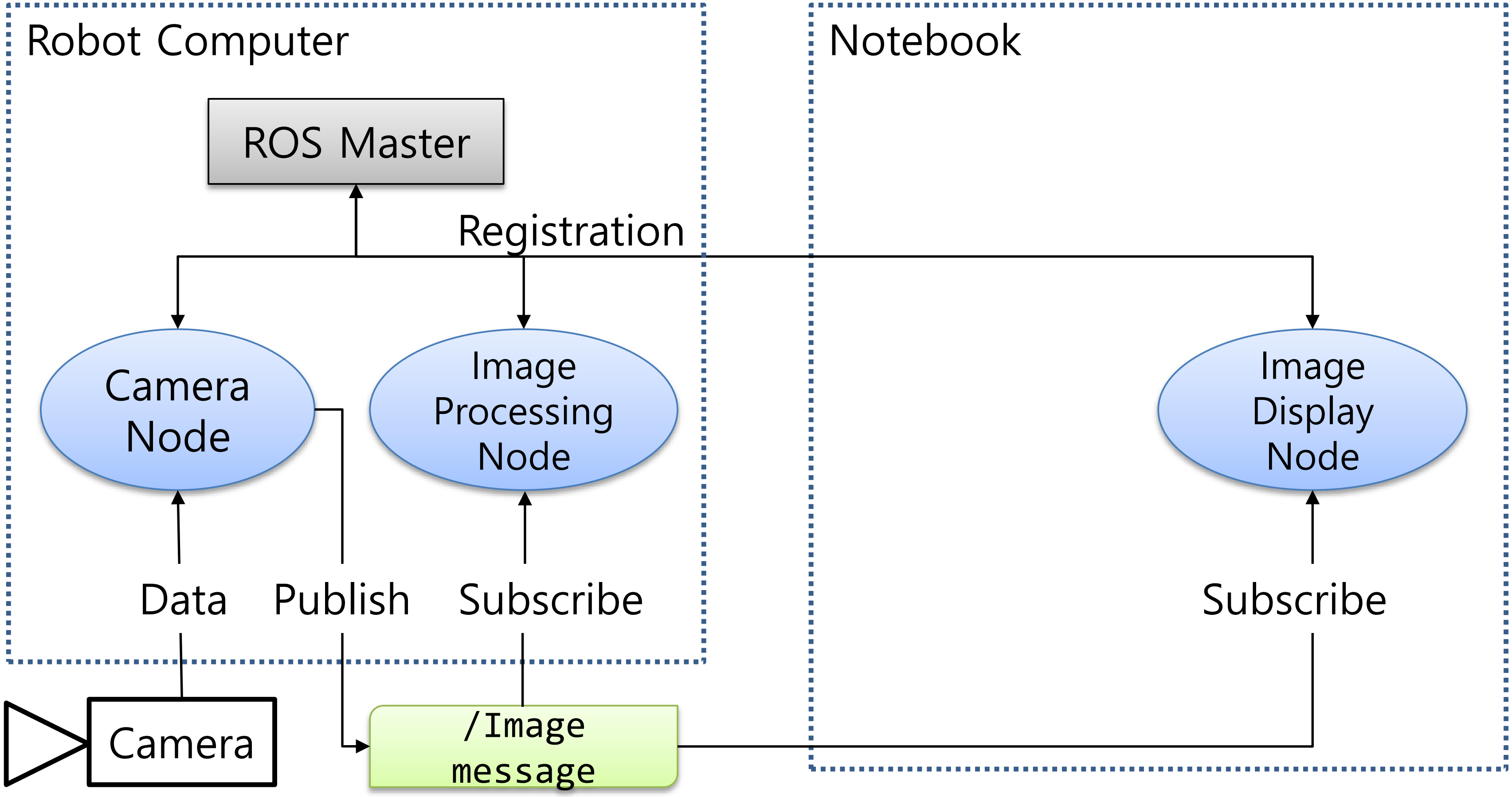
Contents

- What will you learn today?
  - What is ROS? What is Gazebo?
  - **What are the key components?**
  - Which basics are important to know?
  - How does the Robotis OP2 ROS Interface look like? How do I use it?
  - How do I implement a ball follower with the Robotis OP2 using ROS? → Hands-on project

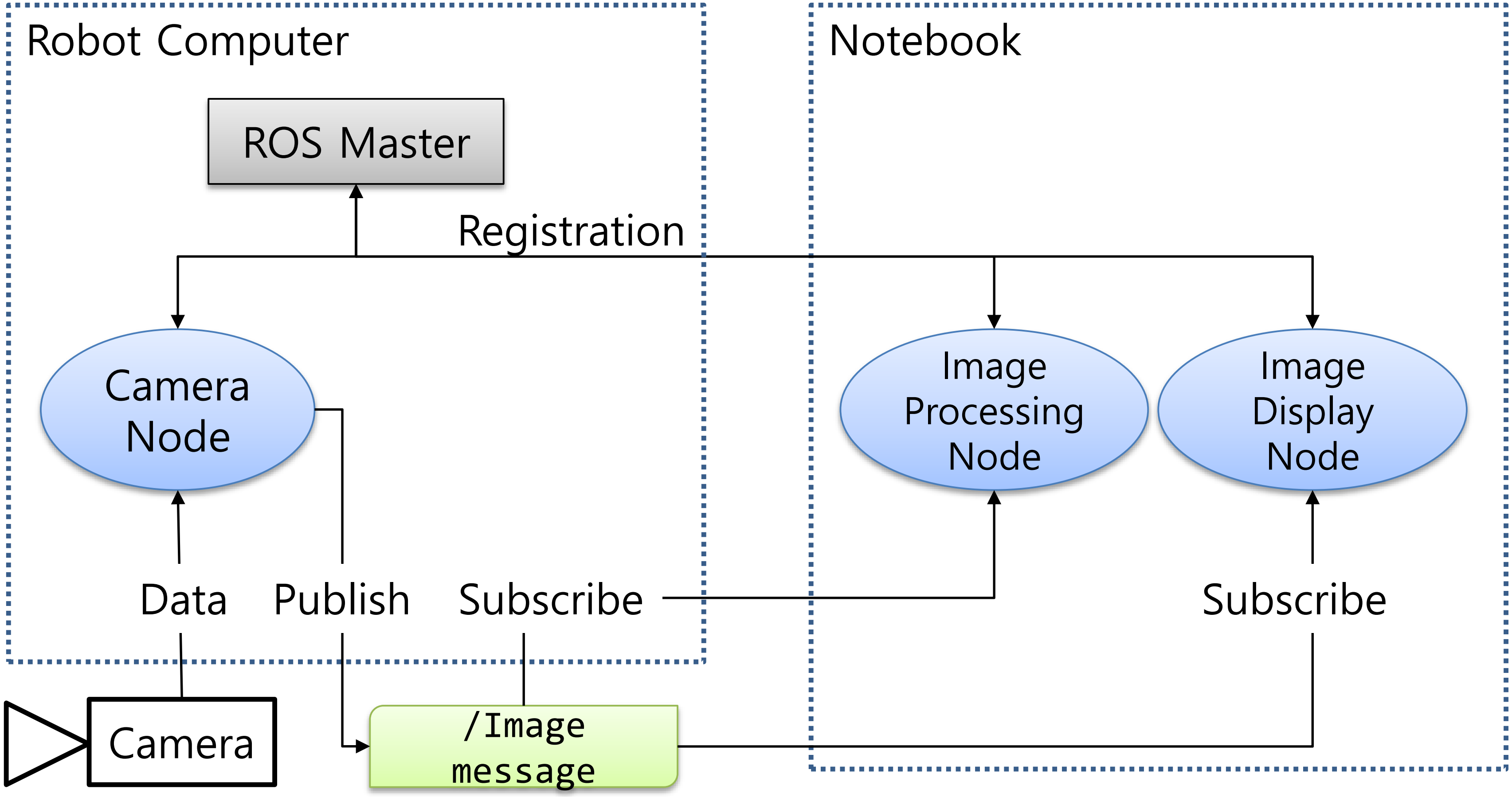
Key concept

- Nodes
    - Modular separated programs → “loose coupling”
- Master
    - Main node, e.g. manages address spaces
- Parameter server
    - Stores data
- Messages
    - Data structures to exchange information
- Topics
    - Message “channels”. Nodes **subscribe** to topics to receive messages or **publish** on topics to send messages

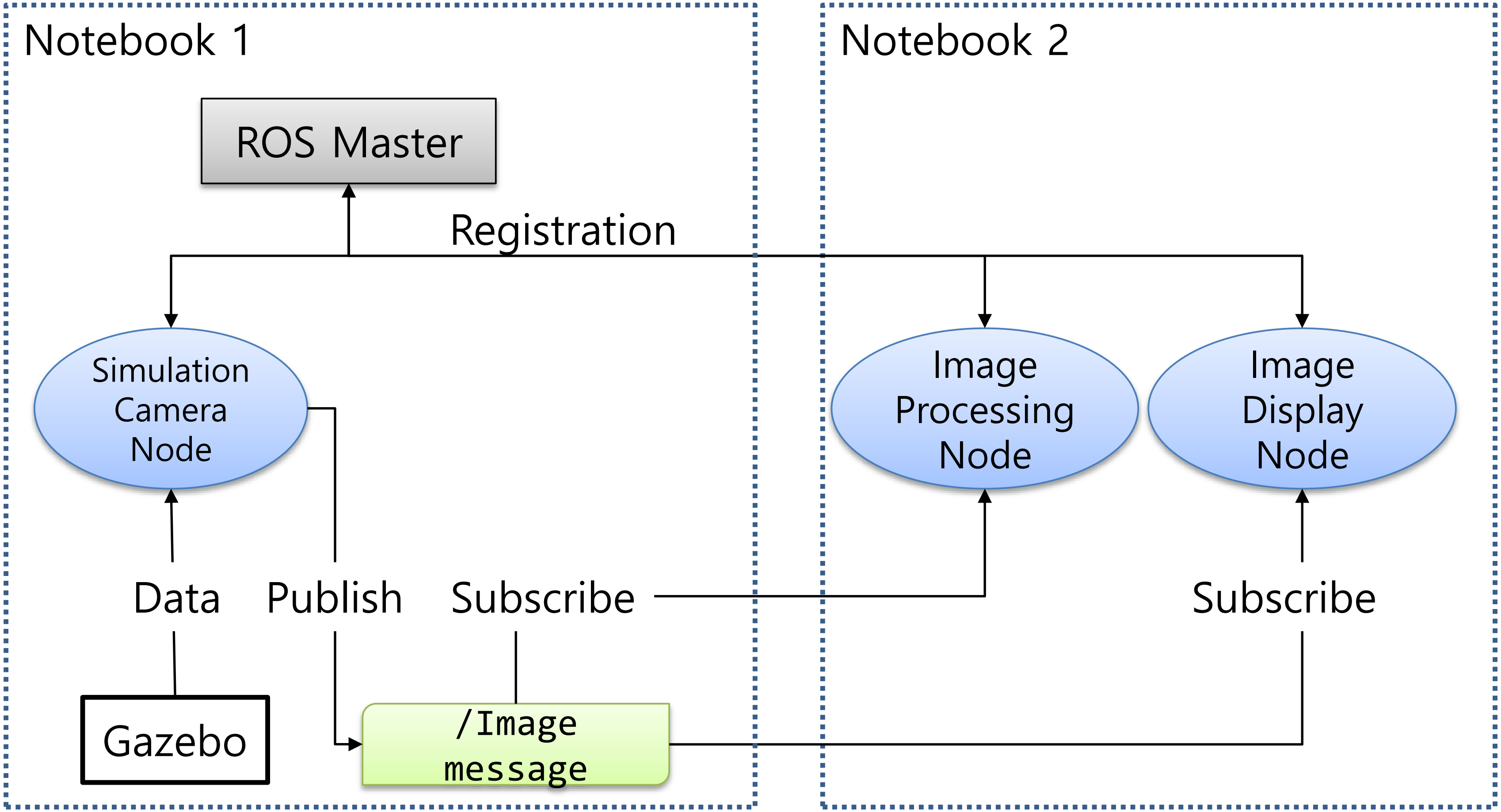
Robot Operating System (ROS)



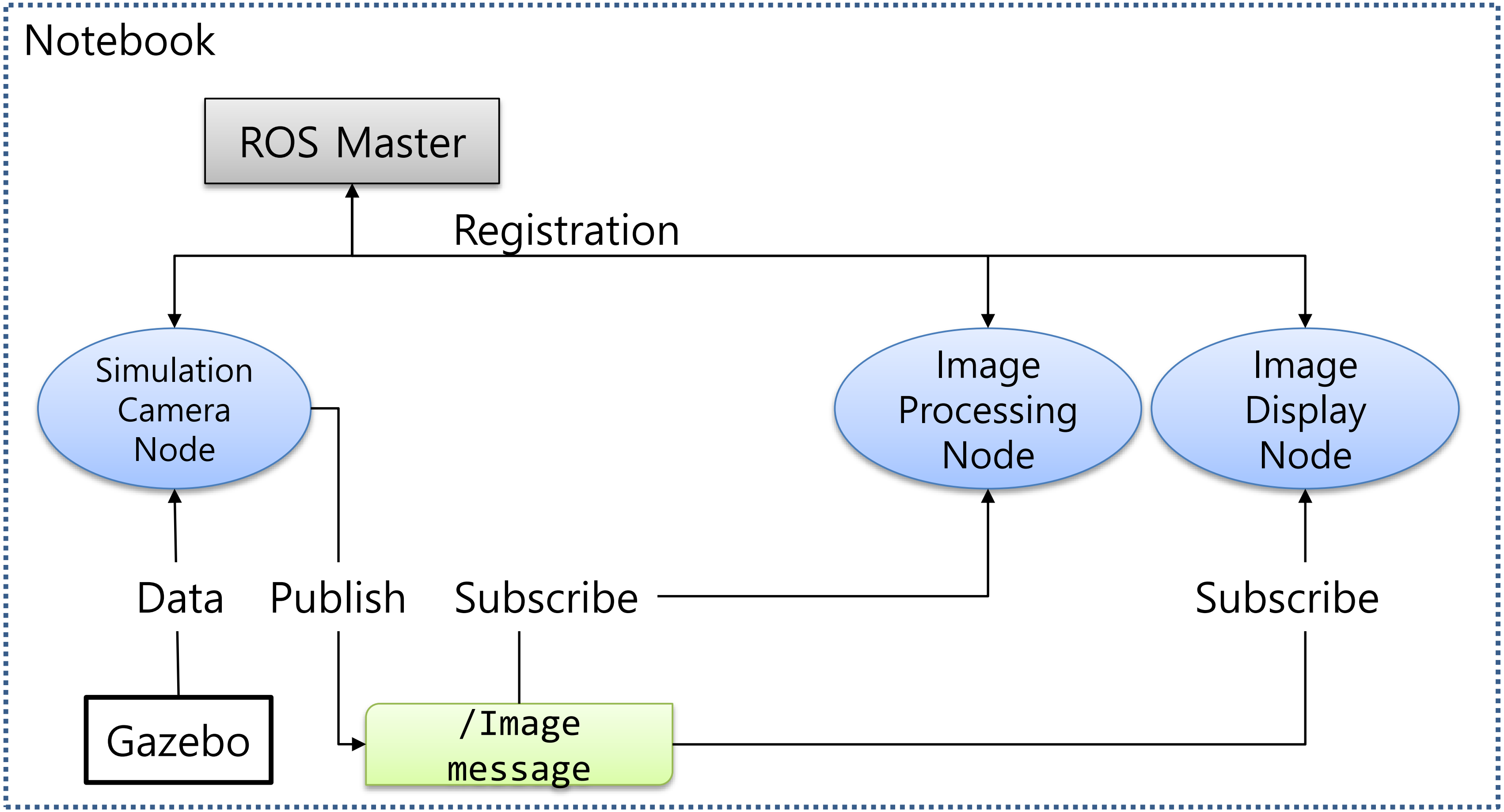
Robot Operating System (ROS)



Robot Operating System (ROS)



Robot Operating System (ROS)



Publisher Example

```
//Initializing Publisher
ros::Publisher vel_pub_;
vel_pub_ = nh_.advertise<geometry_msgs::Twist>("robotis_op
/cmd_vel", 1);
```

message type

topic

```
//Sending message
geometry_msgs::Twist vel;
    vel.angular.z = a_scale_*(joy->axes[axis_angular_r_] -
joy->axes[axis_angular_l_]);
    vel.linear.x = l_scale_*joy->axes[axis_linear_x_];
    vel.linear.y = l_scale_*joy->axes[axis_linear_y_];
    vel_pub_.publish(vel);
```

Subscriber Example

```
// Initializing Subscriber
ros::NodeHandle nh_;
ros::Subscriber image_sub_;
image_sub_ = nh_.subscribe("/robotis_op/camera/image_raw",
100, &RobotisOPBallTrackingNode::imageCb, this);
```

topic

message type

```
//Receiving Image Callback
void RobotisOPBallTrackingNode::imageCb(const sensor_msgs:
:Image& msg)
{
   cv_bridge::CvImagePtr image_ptr;
   image_ptr = cv_bridge::toCvCopy(msg,sensor_msgs::image_
encodings::RGB8);
   […]
}
```

Repetition

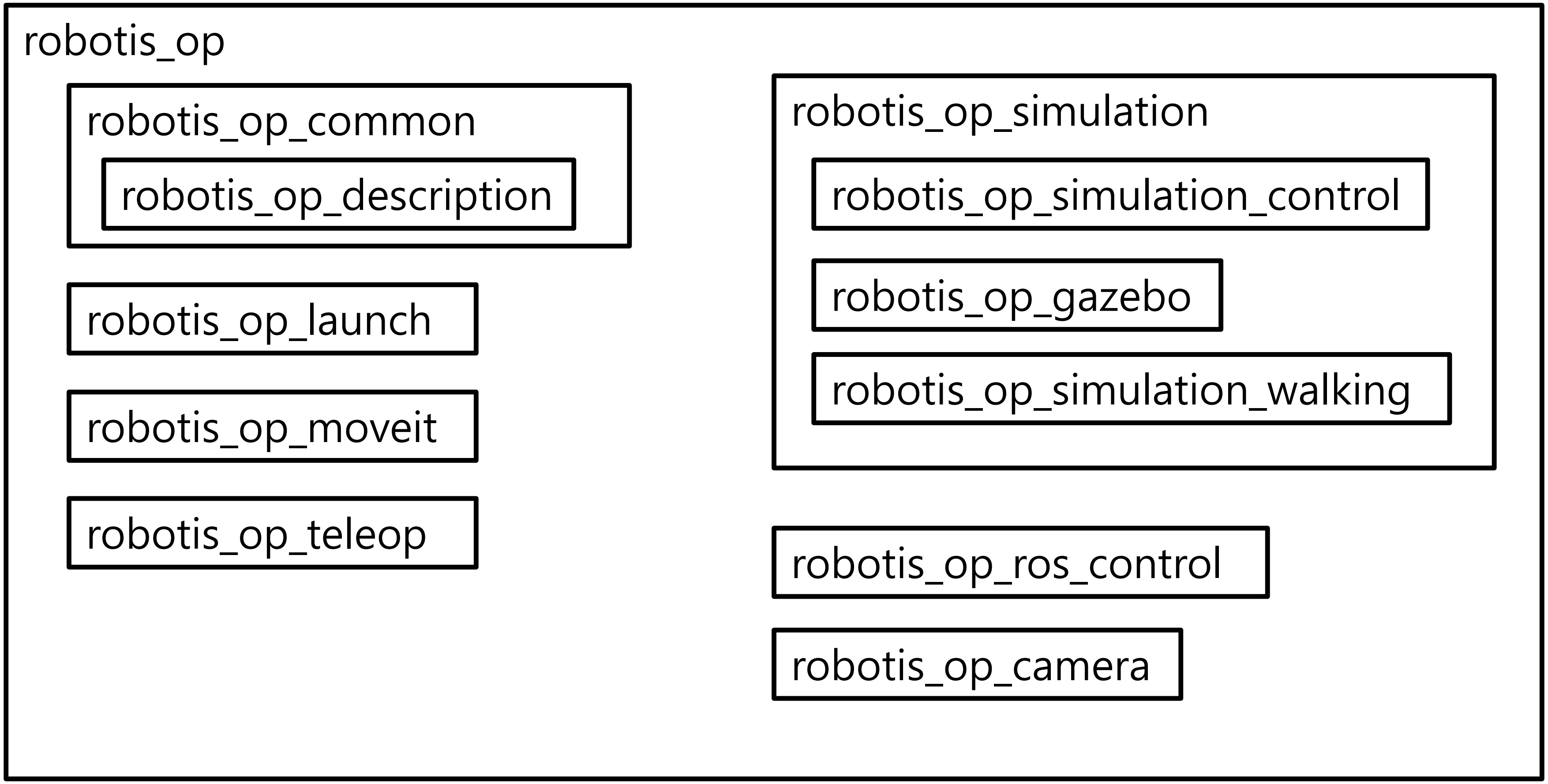
- Nodes
  - Modular separated programs → “loose coupling”
- Master
  - Main node, e.g. manages address spaces
- Parameter server
  - Stores data
- Messages
  - Data structures to exchange information
- Topics
  - Message “channels”. Nodes **subscribe** to topics to receive messages or **publish** on topics to send messages.

ROBOTIS

Contents

- What will you learn today?
  - What is ROS? What is Gazebo?
  - What are the key components?
  - Which basics are important to know?
  - **How does the Robotis OP2 ROS Interface look like? How do I use it?**
  - How do I implement a ball follower with the Robotis OP2 using ROS? → Hands-on project

OP2 ROS Packages



robotis_op

- robotis_op_common
  - robotis_op_description
- robotis_op_launch
- robotis_op_moveit
- robotis_op_teleop
- robotis_op_simulation
  - robotis_op_simulation_control
  - robotis_op_gazebo
  - robotis_op_simulation_walking
- robotis_op_ros_control
- robotis_op_camera

How to use it

- Gazebo – physics simulator with OP2 model

```
roslaunch robotis_op_gazebo robotis_op_gazebo_position_control_soccer_field.launch
```

- Rviz – monitoring tool (image, robot state, …)

```
rosrun rviz rviz
```

- Dynamic reconfigure – dynamic parameter configuration

```
rosrun rqt_reconfigure rqt_reconfigure
```

- Build

```
cd ~/catkin_ws/
catkin_make
```

- Starting the ball tracker node

```
rosrun robotis_op_ball_tracker_tutorial robotis_op_ball_tracker_tutorial_node
```

Contents

- What will you learn today?
  - What is ROS? What is Gazebo?
  - What are the key components?
  - Which basics are important to know?
  - How does the Robotis OP2 ROS Interface look like? How do I use it?
  - **How do I implement a ball follower with the Robotis OP2 using ROS? → Hands-on project**

Project I

- Ball tracking with Gazebo and real OP2
- Ball detection
  - Receive image as sensor_msgs::Image on the topic /robotis_op/camera/image_raw
  - Process with OpenCV

    ```
    bool RobotisOPBallTrackingNode::detectCircles(const
    sensor_msgs::Image& msg, cv::Point& offset)
    ```
- Track movement
  - According to ball detection in the image
  - Publish pan and tilt position on as std_msgs::Float64 on the topics /robotis_op/j_pan_position_controller/command and /robotis_op/j_pan_position_controller/command

Project II

- Try walking towards the ball
  message type: geometry_msgs::Twist
  topic: robotis_op/cmd_vel

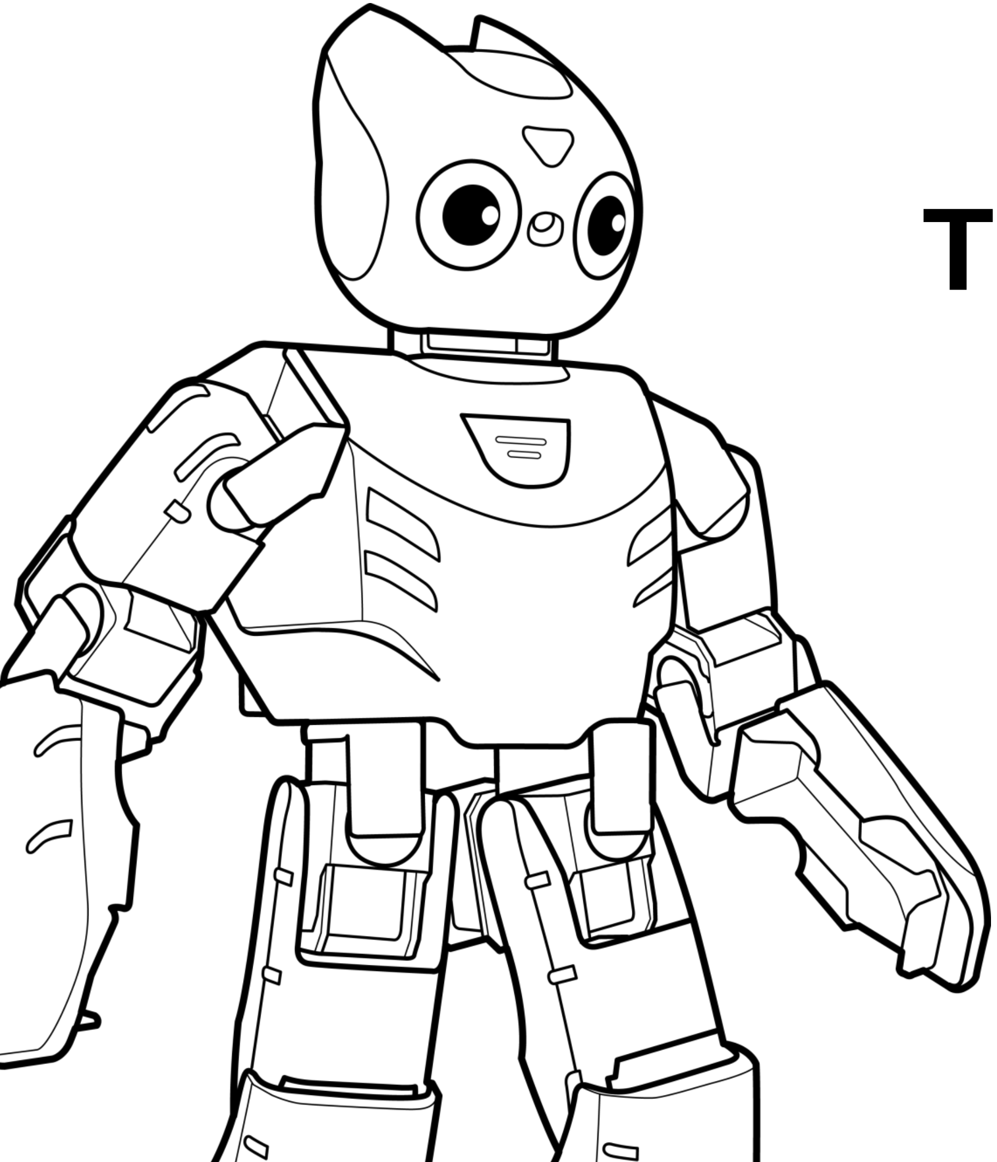
Project III

- Ball tracking real OP2
  - Copy your code to the robot
  - Connect to robot and launch robot

    ```
    ssh robotis@192.168.123.1
    sudo killall demo
    roslaunch robotis_op_onboard_launch robotis_op_whole_robot.launch
    ```

  - On your notebook

    ```
    export ROS_MASTER_URI=http://192.168.123.1:11311
    ```



THANK YOU