Humanoids 2016

THORMANG3 :
Modular, Full-Size Humanoid Platform with ROS

Tutorial
@Humanoids 2016

ROS Enabled

ROBOTIS

Humanoids 2016

You can DOWNLOAD this from ROBOTIS Github:

https://github.com/ROBOTIS-GIT/ROBOTIS-Documents/tree/humanoids2016_tutorial

Tutorial
@Humanoids 2016

ROS Enabled

ROBOTIS

Agenda

- **Introduction to THORMANG3**
- **Hardware Components**
  - Robot Layout
  - Hardware Specifications
- **Basic Operation**
  - System Configuration
  - Running Basic Program
- **Q&A**

History

- 2015 : Competed in the DARPA Robotics Challenge Finals as TEAM ROBOTIS
- 2014 : Released <ROBOTIS MINI>, a 3D printable and programmable humanoid kit
- 2013 : ISO 9001/ ISO 14001 Certified
- 2013 : Selected as IP Star Business (Korean Intellectual Property Office)
- 2012 : <DYNAMIXEL> selected as "World-class Product"
- 2011 : Started <ROBOTIS KidsLab> Classes in US office
- 2010 : <BIOLOID Premium> selected as official kit at Robo-one competition
- 2009 : Released <OLLO> robot kit for beginners
- 2009 : Awarded President's Best Robot Technology Award (Korea)
- 2009 : Incorporated US office and warehouse : ROBOTIS INC
- 2009 : <DARWIN-OP> selected as Open Source Humanoid Project supported by NSF
- 2007 : Technical support for weekly TV program <EBS Robot Power>
- 2005 : Released <BIOLOID> Educational Robot Kit
- 2003 : Released <DYNAMIXEL> smart robot actuator
- 2001 : Released <DiDi-TiTi> Smart Robot Toy
- 1999 : Established ROBOTIS and R&D laboratory
- ~1999: Won numerous robot championship as amateur team at Micromouse and FIRA Robot Soccer competition

Key Facilities and Competencies



HQ is located in Seoul. We provide lectures on weekends, cultivating over 1,000 robotics instructors. Along with our own products, we also display robotic creations made by users.

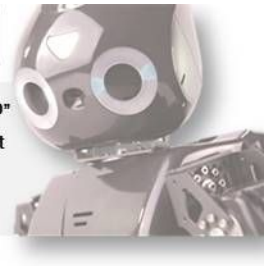

ROBOTIS R&D Laboratory consists of Mechanical, Electrical, Software, and Design section, with equipments ready to fabricate various parts and PCB. Our 2nd R&D Laboratory specializes in Precision Machining for Grade1 gear reduction modules. Up to 2014, we have accumulated 387 kinds of intellectual properties.

ROBOTIS INC: Our US Office located in Irvine, California is in charge of sales and technical support in the US.

The production facility is located near the HQ in Seoul, with the capacity to manufacture over 500,000 Dynamixels and other parts per year.

ROBOTIS
www.robotis.com
[KOREA] #1505 ACE HIGH END TOWER NO.3 145 GASAN DIGITAL 1-RO, GEUMCHEONGU, SEOUL, KOREA 153-787
Tel. +82-70-8671-2609 Fax. +82-70-8230-1336 contactus2@robotis.com
[USA] 1 TECHNOLOGY DR, SUITE F213 IRVINE, CA 92618, USA Tel. +1-949-333-3635 Fax. +1-949-242-5112 america@robotis.com
[JAPAN] japan@robotis.com [CHINA] Tel. +86-10-5798-0703 china@robotis.com

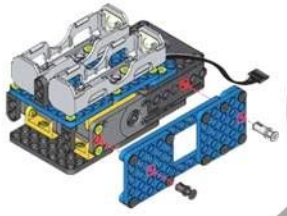
ROBOTIS - We Provide Creativity



ROBOTIS was found in 1999, in response to the question "What is a robot?" Having our foundation in CREATIVITY, we consider it to be the keyword that helps develop childhood dreams into reality and the best solution for designing the future that robots and human live together.

ROBOTIS KIDSLAB

The smart actuators for robots
DYNAMIXEL



STEAM learning tool to boost up CREATIVITY
STEAM= Science. Technology. Engineering. Art. Mathematics

Robotic solution for professional R & D that turns CREATIVE ideas into reality

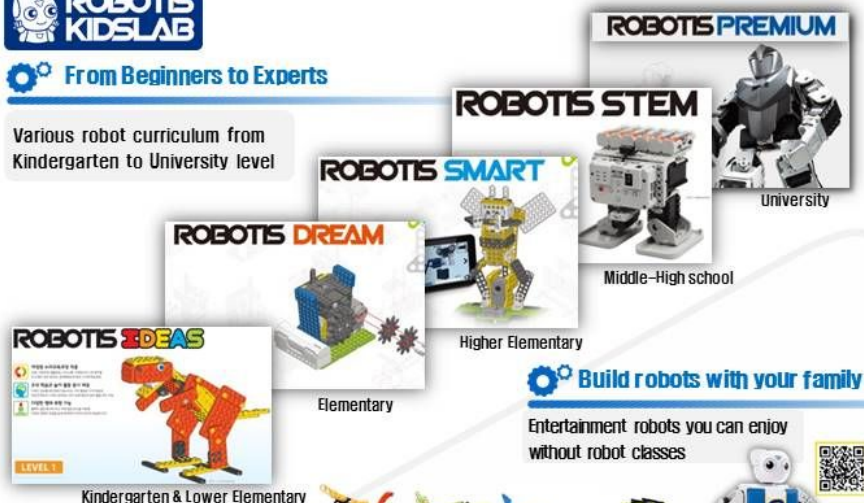
ROBOTIS provides solutions for all customer ranges. For children, the right teaching tools to develop on creativity, and for experts, the best solution to materialize creative ideas into real robots. ROBOTIS products are sold in over 60 countries globally, and is used for various field such as; Kinetic Art, STEAM Education, Robot Sports, Rescue/Surveillance, Medical/Military, etc.

ROBOTIS
www.robotis.com

ROBOTIS KIDSLAB

From Beginners to Experts

Various robot curriculum from Kindergarten to University level



Build robots with your family

Entertainment robots you can enjoy without robot classes

Open-source Project

Share the codes and know-hows from professional communities around the world, utilizing OpenCM ARM Cortex M3 board and ROBOTIS OP, which is best known as open source humanoid platform that you can fabricate from the scratch.

OpenCM9.04 | Dynamixel XL-320 | Sensor Modules | ROBOTIS OP Cross-compiling Environment

Software to Complete a Robot Solution

A Robot Solution cannot be completed with just the hardware. ROBOTIS offers various and easy to use software to fulfill a complete robot solution.

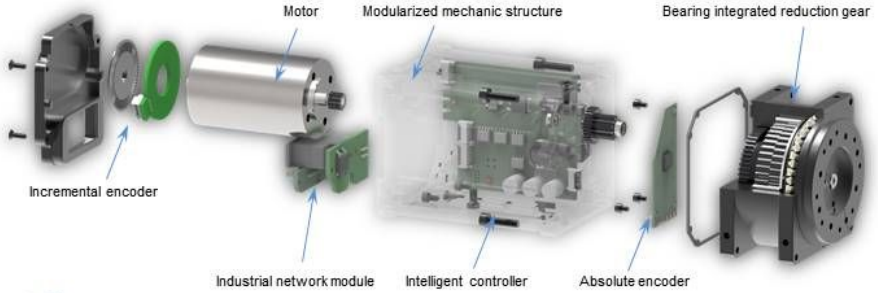
OpenCM : Arduino-style IDE | R+ Task : Programming | R+ Motion: Motion Editor | R+ Design : 3D Assembly S/W

Robot Smart Actuator, Dynamixel

DYNAMIXEL

Key Strengths

1. Robot Exclusive : Compact and lightweight for building robots. Best for battery-operated mobile robots that needs to be durable from collision.
2. All-in-one : sensors, drivers, network and reduction gears are integrated into the rectangular form factor
3. Modular : Modular structure for easy assembly and maintenance like block toys
4. Network driven : Driven through a network, allowing systematic control for complicated systems



Lineup

DYNAMIXEL

- Spur Gear
- Position control based on 12bit Sensor feedback
- TTL 485 communication
- Output up to 10W at 12V

DYNAMIXEL PRO

- Cycloid type detachable reduction gear
- Current based torque control
- Output 10~200W at 24V DC

Applicable Fields and Platforms

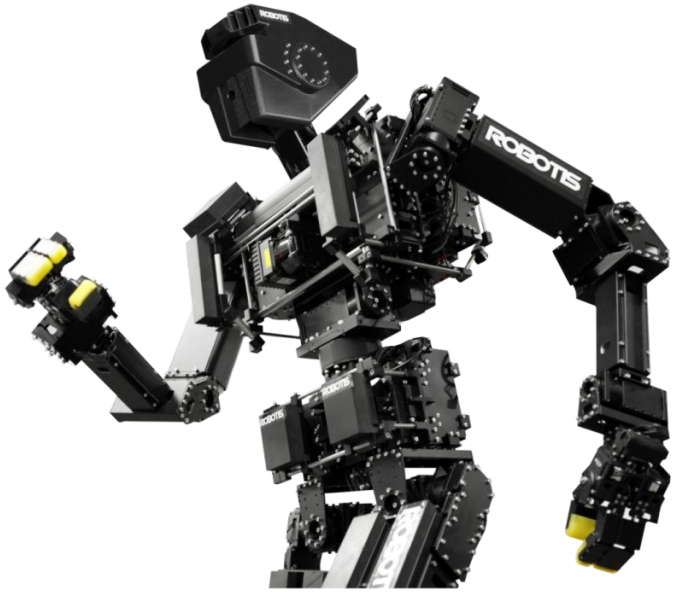
Team ROBOTIS @ DRC Final in 2015

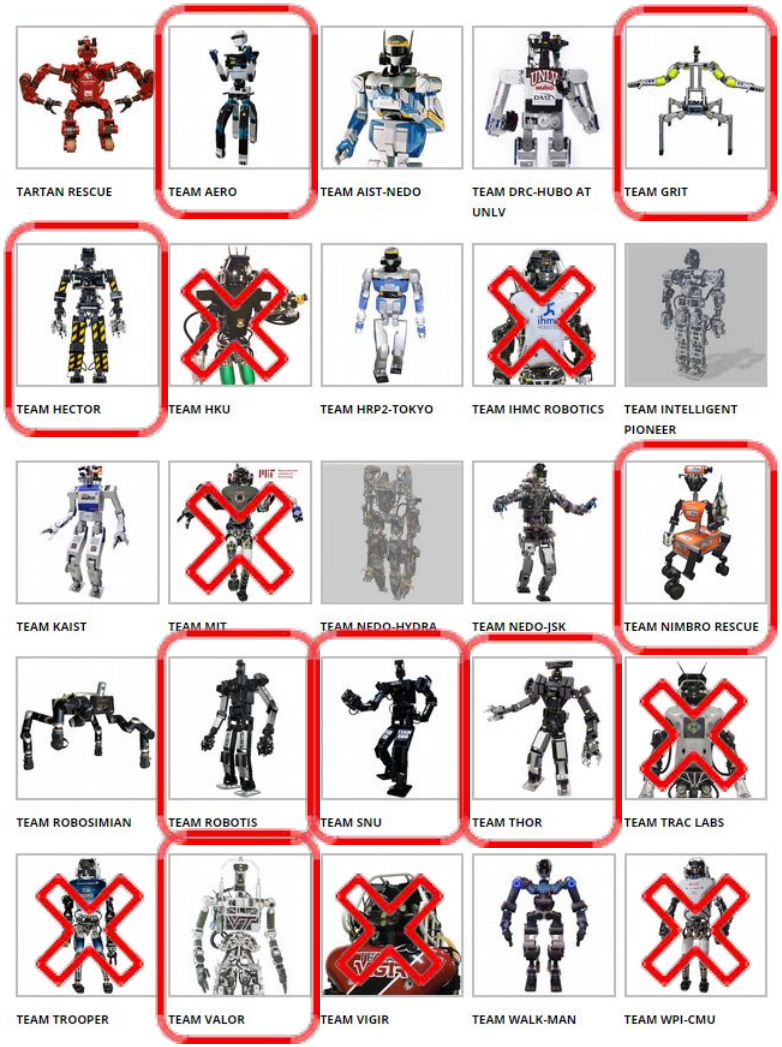
ROBOTIS-Most Widely Used Platform

Platform Provider

- 8 of 15 teams who

built their own platforms were using the THORMANG platform or DXL-Pros



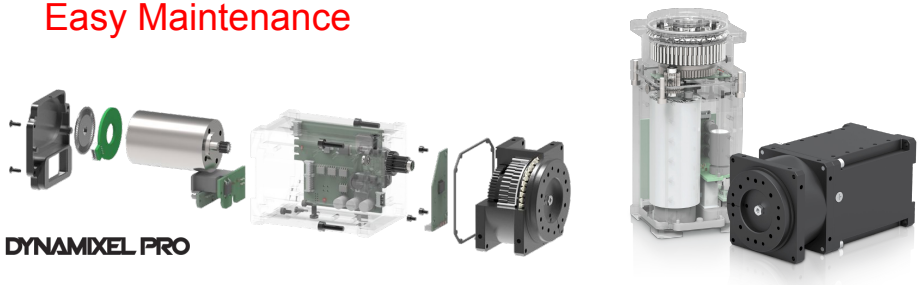


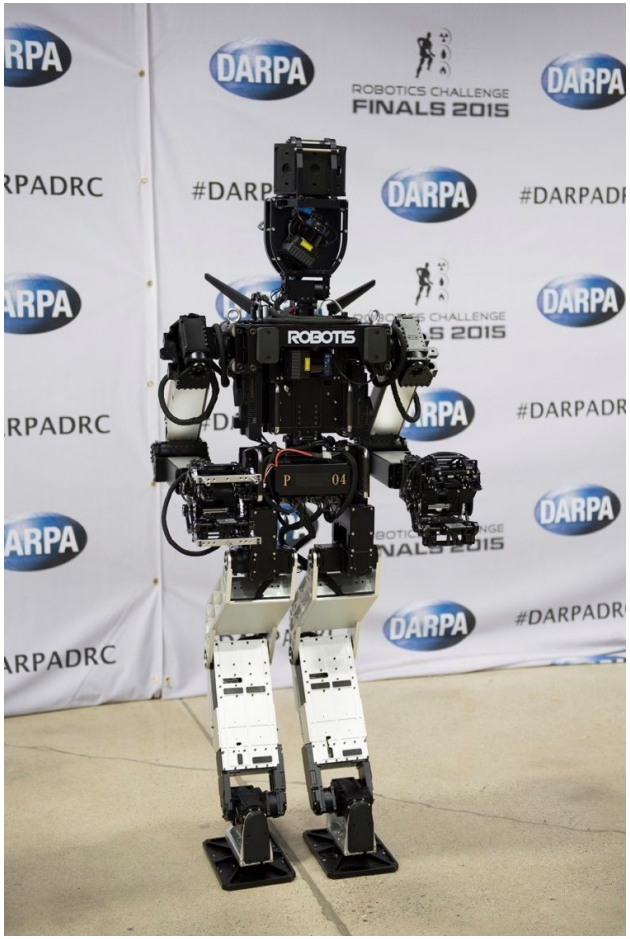
ROBOTIS

THORMANG-Powered by Dynamixel PROs

Dynamixel Pro - Full Modular Solution

- High Power, High Precision
- Cycloidal Gear Reduction System provides proven durability and quality
- Reconfigurable, Expandable, Adaptable
- Easy Maintenance





ROBOTIS

THORMANG @ DRC Final in 2015

THORMANG @ Humanoids 2015

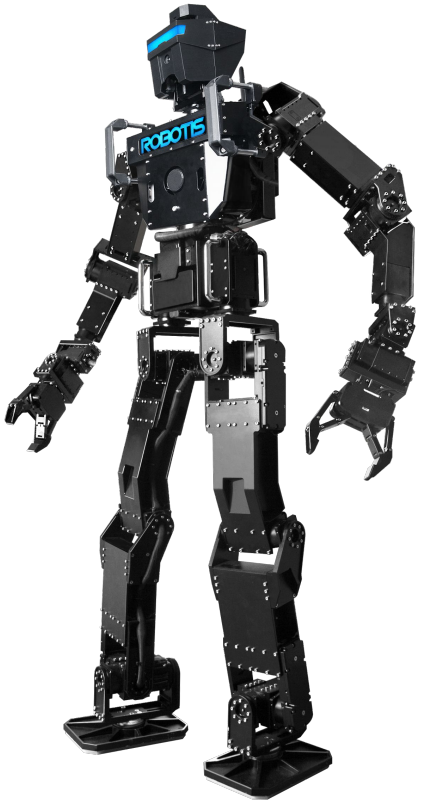
Rapid Progression: THORMANG3

- **THORMANG3**
  - Human-Friendly Scale
  - Powerful Dual PC-level Computing with Intel NUC
  - Various Sensing with Force/Torque Sensor, LIDAR/IMU/Camera
  - **Full ROS Support** with 3D CAD data and **Open-Source SDK**
  - DYNAMIXEL PRO based Modular Design

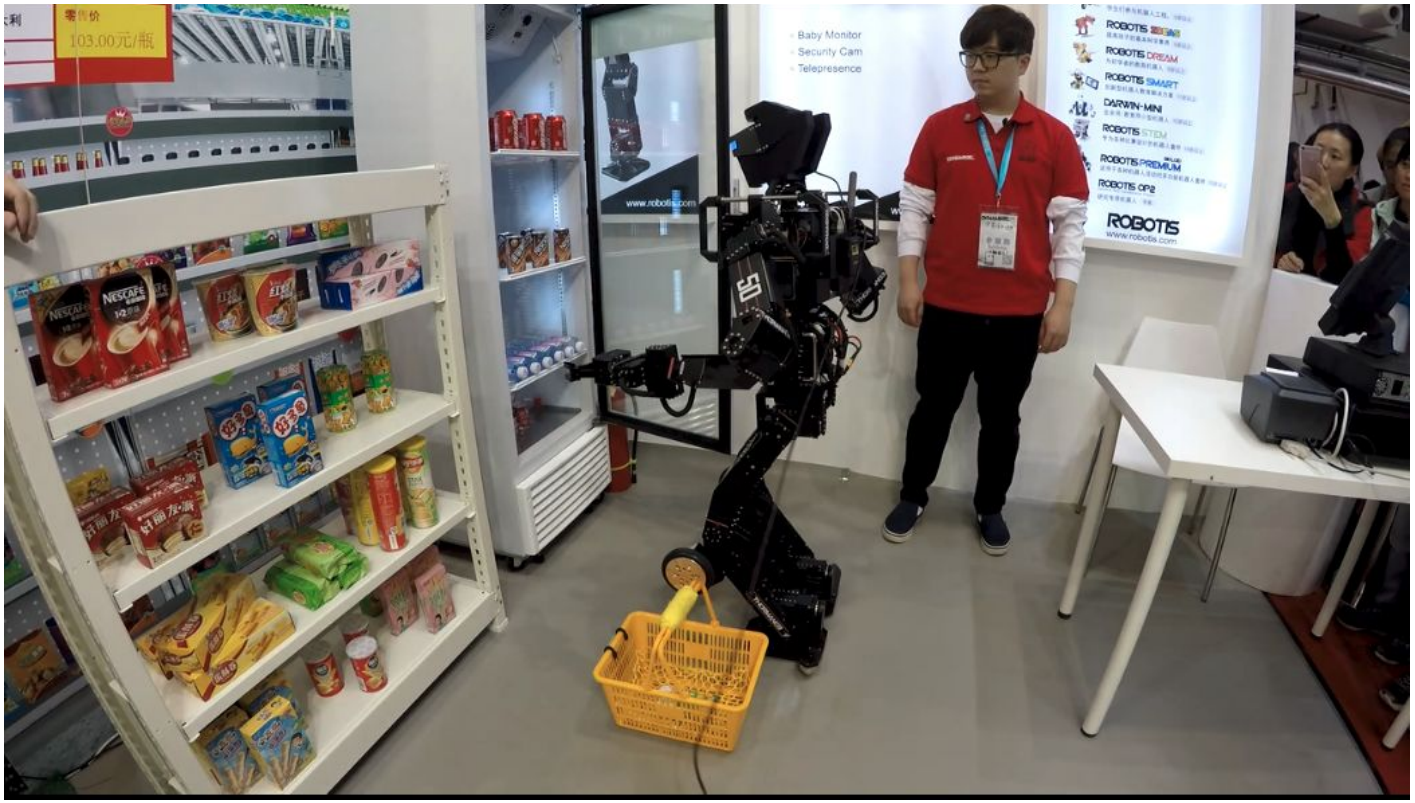
ROS Enabled

intel NUC

ROBOTIS GIT
ROBOTIS Official Github
#1505, Gasan Digital-1ro 145 (Ace High E... http://en.robotis.com/index/



Recent Approach: WRC2016 (Beijing, China)

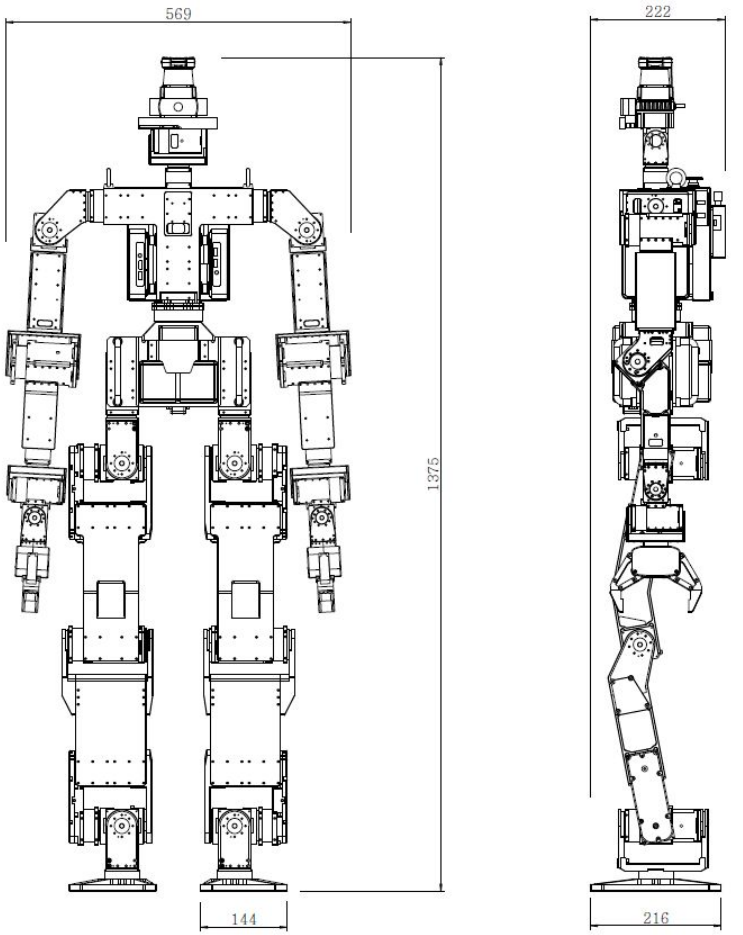


Agenda

- Introduction to THORMANG3
- **Hardware Components**
  - Robot Layout
  - Hardware Specifications
- Basic Operation
  - System Configuration
  - Running Basic Program
- Q&A

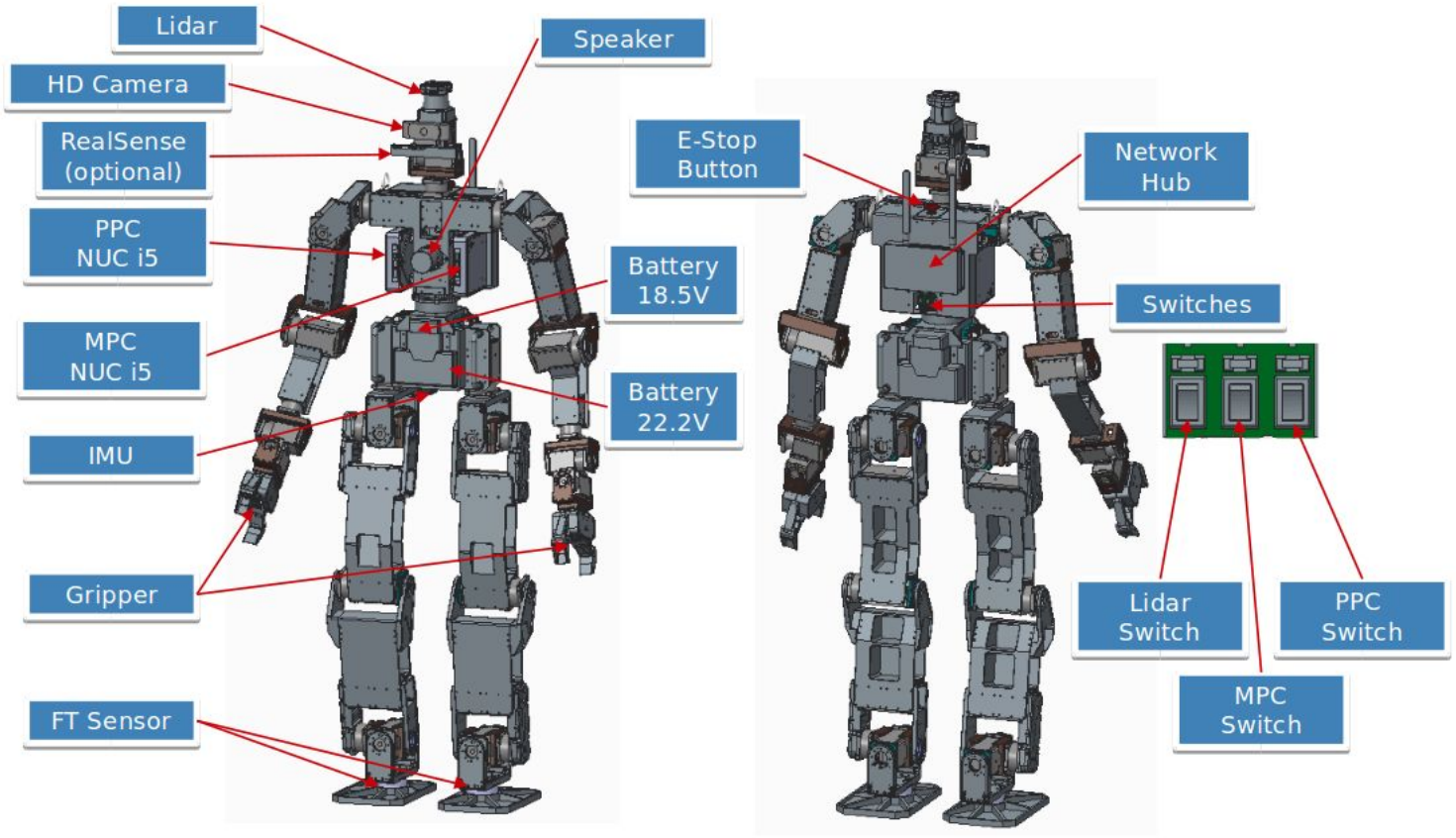
Hardware Specifications

DOF : 29 (+ 2 Gripper)
Actuator : 200W x 10 / 100W x 11 / 20W x 8
Computer : Intel NUC i5
(8GRAM / M.2 SSD) x 2
Wireless Router : D-Link DIR-806A x 1
Sensors :
Logitech C920 HD Camera x 1
Intel RealSense x 1
Hokuyo UTM-30LX-EW
F/T : ATi Mini58-SI-2800-120 x 2
IMU : MicroStrain 3DM-GX4-25 x 1
Battery : 22V, 22000mA x 1
18.5V, 11000mA x 1
Height : 137.5cm
Weight : 42Kg

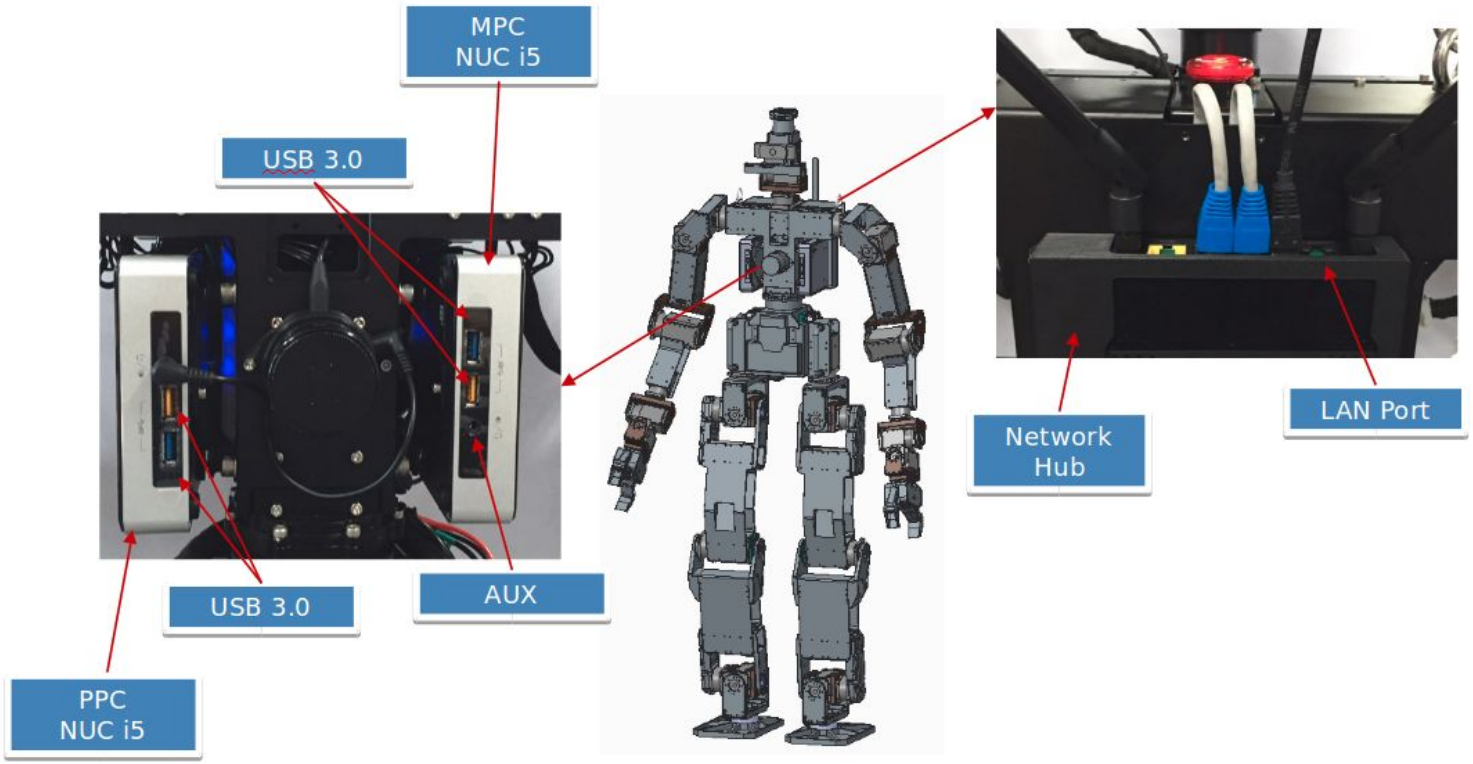


ROBOTIS

Robot Layout (1)

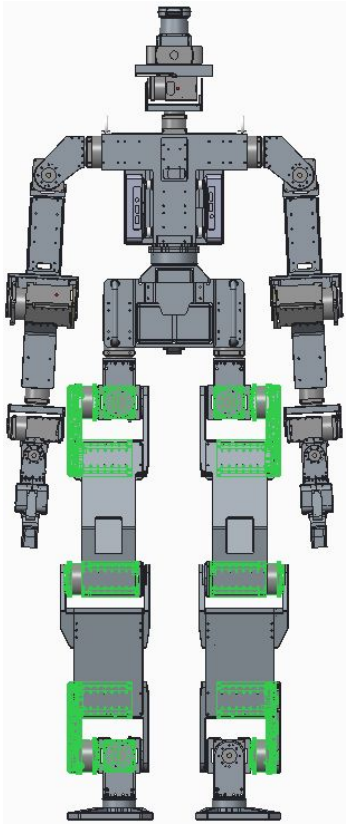


Robot Layout (2)

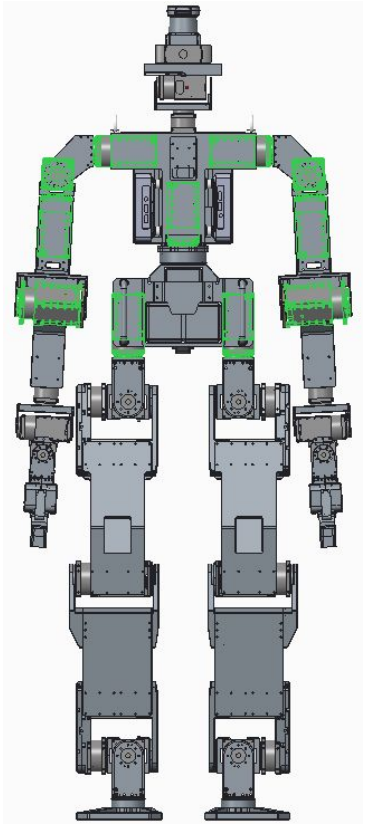


Actuator

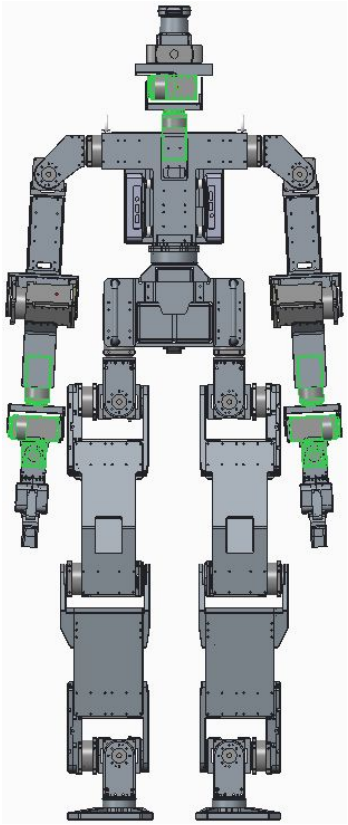
- **Dynamixel Pro 29 DOF**



Dynamixel Pro 200W X 10



Dynamixel Pro 100W X 11

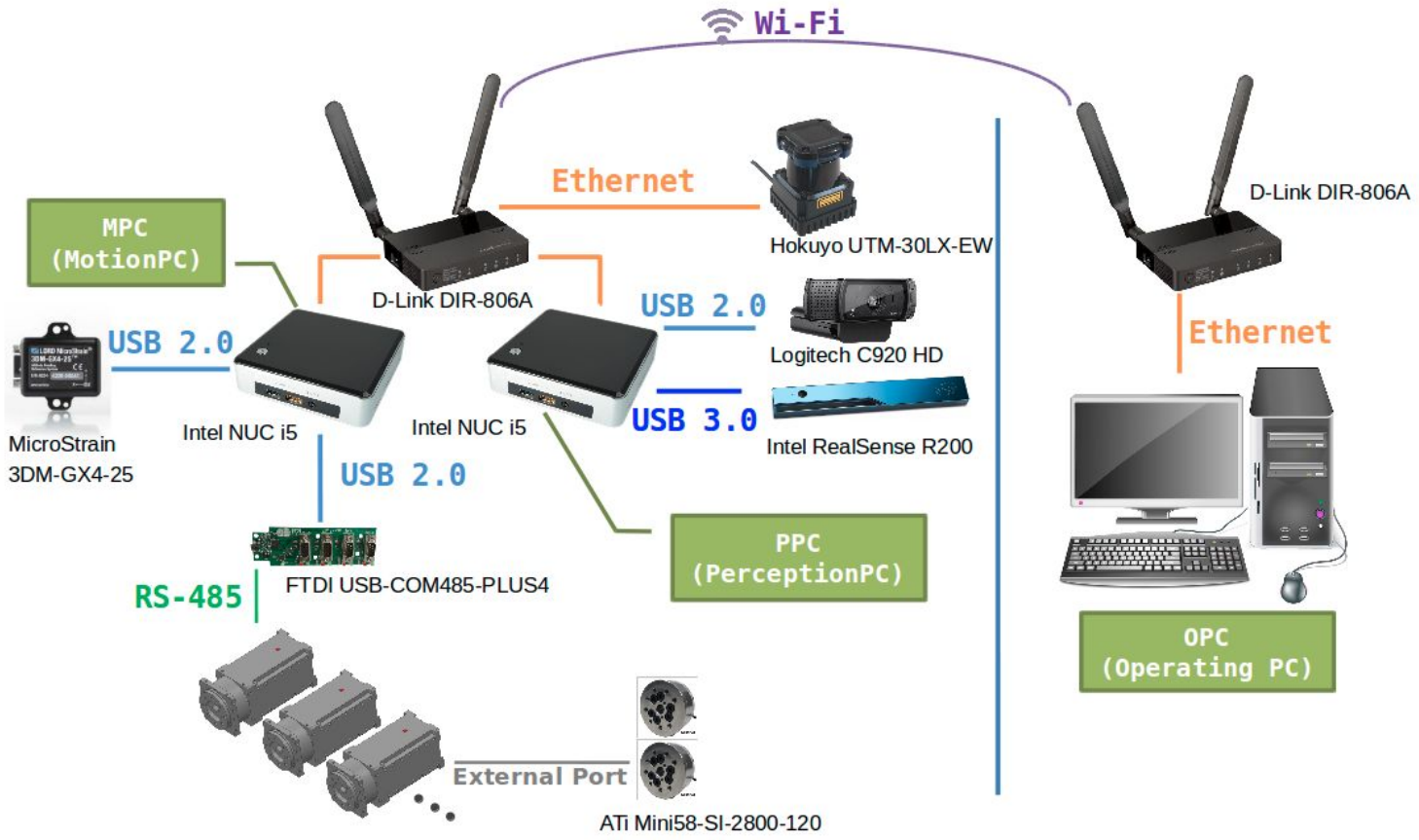


Dynamixel Pro 20W X 8

Agenda

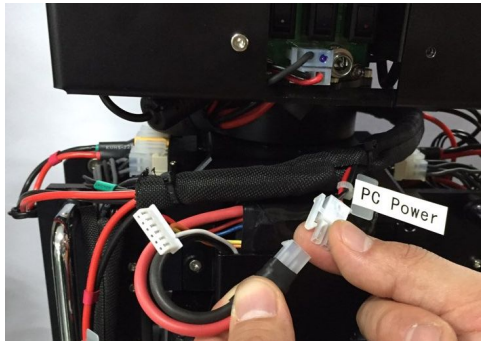
- Introduction to THORMANG3
- Hardware Components
  - Robot Layout
  - Hardware Specifications
- **Basic Operation**
  - System Configuration
  - Running Basic Program
- Q&A

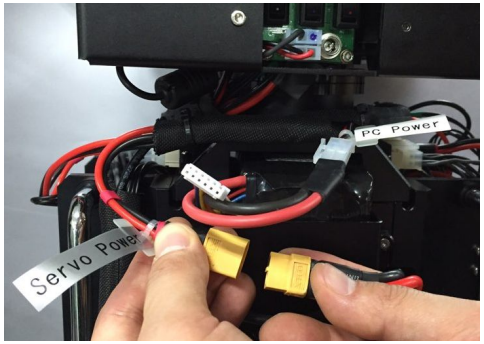
System Configuration



Powering On THORMANG3 (1)

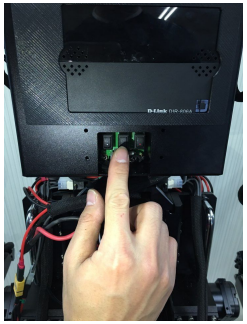
- **Case 1. Using the Power Supply**
  - Connect the power supply (18V, > 10A) to the PC Power cable to provide power to the PCs.
  - Connect the power supply (24V, >30A) to the Servo Power cable to provide power to the actuators.
- **Case 2. Using the Battery Packs**
  - Ensure the batteries are fully charged.
  - Open the battery compartment door by loosening the thumbscrew.
    Insert the battery packs. Close the compartment by tightening the thumbscrew.
  - Connect the 22.2V battery cable (yellow jack) to Servo Power and connect the 18.5V battery cable (white terminal) to PC Power.

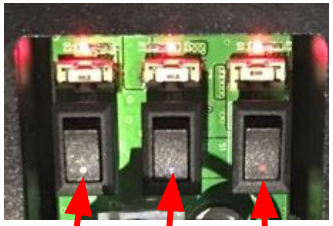




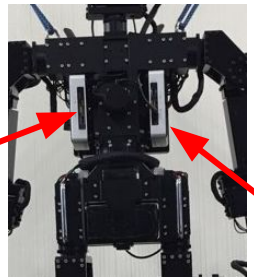
Powering On THORMANG3 (2)

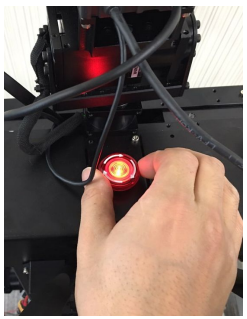
1. Flip the three switch located on the power board
2. Push the MPC and PPC power buttons.
3. Turn the E-Stop Button.
   (If the E-Stop Button is pressed,
   the DXL power is turned off)

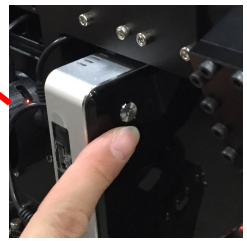


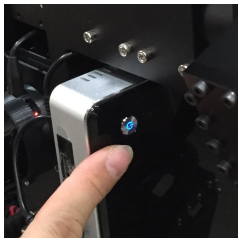


PPC Power Button









MPC Power Button

Running Basic Program (1)

Connecting THORMANG3 to your PC

From your computer go to your LAN settings and set static IP as follows: **10.17.3.xxx**

WIFI Network : THORMANG-Sxx or THORMANG-Sxx-5G

Connection Information

- MPC (Motion PC) IP : 10.17.3.30
- PPC (Perception PC) IP : 10.17.3.35
- MPC & PPC User Name : robotis
- MPC & PPC Password : 111111

Running Basic Program (2)

Connecting THORMANG3 to your PC

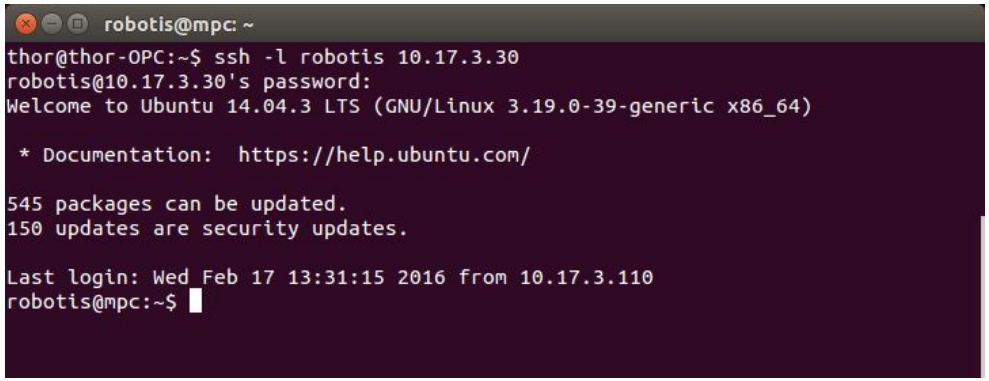
ROBOTIS recommends that users connect via an SSH client

Example: Ubuntu SSH Client

1. Open the terminal window
2. Type the following SSH command utilizing the MPC's user name and IP address :

   $ ssh robotis@10.17.3.30

3. Input the MPC's password : 111111



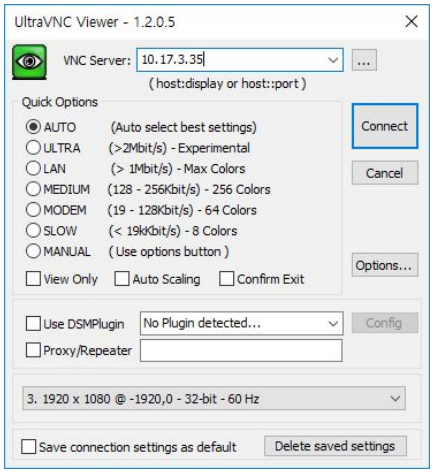
Running Basic Program (3)

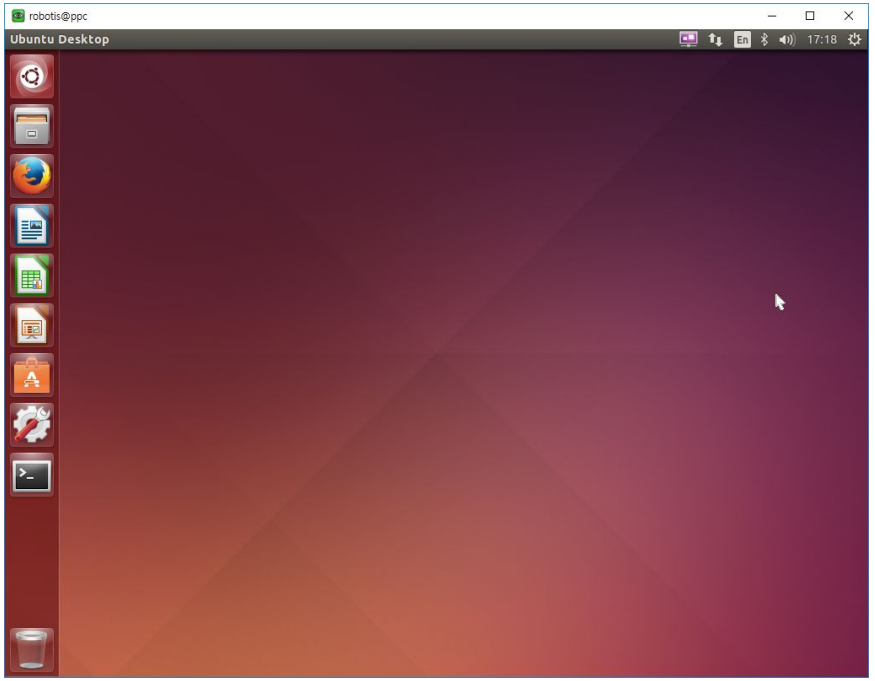
Connecting THORMANG3 to your PC

Accessing the MPC via remote desktop may result in slower performance.

Example: Windows VNC client

1. Execute VNC client program
2. Input the MPC's IP address : 10.17.3.30
3. Input the MPC's password : 111111





Running Basic Program (4)

Running roscore and THORMANG3 Manager

1. **roscore**

- Connect to the PPC via SSH client program. (**IP: 10.17.3.35**)
- **roscore** can be launched using the **roscore** executable:

  $ roscore

2. **THORMANG3 Manager**

   thormang3_manager is a base node using ROBOTIS' framework.
   thormang3_manager must be running before you can run the Simple Demo nodes and before you can check the sensors because they are using **thormang3_manager**.

- Connect to the MPC via SSH client program. (**IP: 10.17.3.30**)
- To launch THORMANG3 Manager, simply type the following :

  $ sudo bash

  # roslaunch thormang3_manager thormang3_manager.launch

Running Basic Program (5)

Time Sync between MPC and PPC

3. time_sync

- It is necessary to synchronize each PC to get exact sensor data
- Connect to the MPC via SSH client program. (**IP: 10.17.3.30**)
- To synchronize time, simply type the following :

 # ./timesync

```
$ sudo service ntp stop
$ sudo date --set='-2 secs'
$ sudo ntpdate 10.17.3.35
$ sudo hwclock -w
```

Running Basic Program (6)

Running Simple Demo

1. **Walking Simple Demo**

   Walking Module provides two kinds of functions

   - One step forward/backward walking
   - Sensor Feedback Balance On/Off

2. **Manipulation Simple Demo**

   Manipulation Control Module allows for two kinds of control :

   - Joint Space Control
   - Task Space Control

Running Basic Program (7)

Running Simple Demo

1. **Walking Simple Demo**

   Walking Module provides two kinds of functions

   - One step forward/backward walking
   - Sensor Feedback Balance On/Off

   Connect to the PPC via SSH client program. (**IP: 10.17.3.35**)
   Execute the Walking Simple Demo by typing the following command :

   $ rosrun thormang3_walking_demo thormang3_walking_demo_node

2. **Manipulation Simple Demo**

Running Basic Program (8)

Running Simple Demo (Walking Simple Demo)

These commands should be executed in new terminal.

Initialization 1 : go to initial pose (from Base Module)

$ rostopic pub -1 /robotis/walking_demo/command std_msgs/String ini_pose

Initialization 2 : set Walking Control Module

$ rostopic pub -1 /robotis/walking_demo/command std_msgs/String set_mode

demo 1 : make balance algorithm enable/disable – Balance ON / OFF

$ rostopic pub -1 /robotis/walking_demo/command std_msgs/String balance_on
$ rostopic pub -1 /robotis/walking_demo/command std_msgs/String balance_off

demo 2 : walk forward – One step forward walking

$ rostopic pub -1 /robotis/walking_demo/command std_msgs/String forward

demo 3 : walk backward – One step backward walking

$ rostopic pub -1 /robotis/walking_demo/command std_msgs/String backward

Running Basic Program (9)

Running Simple Demo (Walking Simple Demo)

In terminal, you can confirm successful demo execution by comparing your screen output to the following:

```
robotis@mpc:~$ rosrun thormang3_walking_demo thormang3_walking_demo_node
[ INFO] [1455875066.217024182]: [Demo]  : receive [ini_pose] msg
[ INFO] [1455875066.217090233]: [Demo]  : go to initial pose
[ INFO] [1455875066.217208610]: [Demo]  : please wait 5 seconds
[ INFO] [1455875075.729093836]: [Demo]  : receive [set_mode] msg
[ INFO] [1455875075.729155435]: [Demo]  : set walking control mode
[ INFO] [1455875075.731306347]: [Robot] : Walking_Module_is_enabled
[ INFO] [1455875085.883878337]: [Demo]  : receive [balance_on] msg
[ INFO] [1455875085.883942699]: [Demo]  : balance enable
[ INFO] [1455875085.890794091]: [Demo]  : Succeed to set balance param
[ INFO] [1455875085.890860928]: [Robot] : Balance_Param_Setting_Started
[ INFO] [1455875086.885257816]: [Robot] : Balance_Param_Setting_Finished
[ INFO] [1455875097.554503231]: [Demo]  : receive [forward] msg
[ INFO] [1455875097.554568993]: [Demo]  : forward walking
[ INFO] [1455875097.562789075]: [Demo]  : Succeed to add step data array
[ INFO] [1455875097.563741307]: [Robot] : Walking_Started
[ INFO] [1455875104.660218824]: [Robot] : Walking_Finished
[ INFO] [1455875110.671069684]: [Demo]  : receive [backward] msg
[ INFO] [1455875110.671131309]: [Demo]  : backward walking
[ INFO] [1455875110.674200761]: [Demo]  : Succeed to add step data array
[ INFO] [1455875110.675751408]: [Robot] : Walking_Started
[ INFO] [1455875117.772244468]: [Robot] : Walking_Finished
```

Running Basic Program (10)

Running Simple Demo (Walking Simple Demo)

Running Basic Program (11)

Running Simple Demo (Manipulation Simple Demo)

1. **Walking Simple Demo**

2. **Manipulation Simple Demo**

   Manipulation Control Module allows for two kinds of control :

   - Joint Space Control
   - Task Space Control

   Connect to the PPC via SSH client program. (**IP: 10.17.3.35**)
   Execute the Manipulation Simple Demo by typing the following command :

   $ rosrun thormang3_manipulation_demo thormang3_manipulation_demo

Running Basic Program (12)

Running Simple Demo (Manipulation Simple Demo)

These commands should be executed in new terminal.

Initialization 1 : go to initial pose (from Base Module)

$ rostopic pub -1 /robotis/manipulation_demo/command std_msgs/String ini_pose

Initialization 2 : set Manipulation Control Module

$ rostopic pub -1 /robotis/manipulation_demo/command std_msgs/String set_mode

demo 3 : go to manipulation base pose **– Joint Space Control**

$ rostopic pub -1 /robotis/manipulation_demo/command std_msgs/String base_pose

demo 4 : move right arm **– Task Space Control**

$ rostopic pub -1 /robotis/manipulation_demo/command std_msgs/String right_arm

demo 5 : move left arm **– Task Space Control**

$ rostopic pub -1 /robotis/manipulation_demo/command std_msgs/String left_arm

Running Basic Program (13)

Running Simple Demo (Manipulation Simple Demo)

In terminal, you can confirm successful demo execution by comparing your screen output to the following:

```
thor@SIM-PC:~$ rosrun thormang3_manipulation_demo thormang3_manipulation_demo
[ INFO] [1456273852.518346294]: Robotis Thormang3 Manipulation Simple Demo
[ INFO] [1456273858.108976598, 1877.168000000]: demo 1: go to initial pose
[ INFO] [1456273929.708964751, 1946.535000000]: demo 2: set manipulation control mode
[ INFO] [1456273935.924793739, 1952.625000000]: demo 3: go to manipulation base pose
[ INFO] [1456273943.555563594, 1960.293000000]: demo 4: move right arm
[ INFO] [1456273949.728911625, 1966.374000000]: demo 5: move left arm
```

Running Basic Program (14)

Running Simple Demo (Manipulation Simple Demo)

Running Basic Program (15)

Check the Sensors

Run the MPC's Sensors (IMU, FT, Lidar)

Type the following commands to run the MPC's sensors & motors :

$ sudo bash

 # roslaunch thormang3_manager thormang3_manager.launch

How to Check the MPC's Sensors

IMU : Type the following command and check the output :

$ rostopic echo /robotis/sensor/imu/imu

```
robotis@mpc:~$ rostopic echo /robotis/sensor/imu/imu
header:
  seq: 218798
  stamp:
    secs: 1456227058
    nsecs: 657393447
  frame_id: imu
orientation:
  x: 0.551555931568
  y: 0.831686019897
  z: -0.0350182652473
  w: 0.0534624755383
orientation_covariance: [-1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
angular_velocity:
  x: 0.00529906712472
  y: 0.0256397109479
  z: 0.108881101012
angular_velocity_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
linear_acceleration:
  x: 1.40463167375
  y: 0.0995190255708
  z: 9.7412729802
linear_acceleration_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
```

Running Basic Program (16)

Check the Sensors

How to Check the MPC's Sensors

FT Sensor : Type the following commands and check the output :

$ rostopic echo /robotis/sensor/ft_right_foot/raw

$ rostopic echo /robotis/sensor/ft_left_foot/raw

```
robotis@mpc:~$ rostopic echo /robotis/sensor/ft_right_foot/raw
header:
  seq: 110484
  stamp:
    secs: 1456227469
    nsecs: 577939051
  frame_id: r_leg_foot_link
wrench:
  force:
    x: 5175.13464253
    y: 12796.9383106
    z: 9230.67538064
  torque:
    x: -134.298186423
    y: -234.033554692
    z: -267.399062596
```

```
robotis@mpc:~$ rostopic echo /robotis/sensor/ft_left_foot/raw
header:
  seq: 113669
  stamp:
    secs: 1456227506
    nsecs: 383800602
  frame_id: l_leg_foot_link
wrench:
  force:
    x: 9699.47838375
    y: 5305.99705173
    z: 10538.1527615
  torque:
    x: -243.133327243
    y: -325.658292323
    z: -202.514837491
```

Running Basic Program (17)

Check the Sensors

How to Check the MPC's Sensors

Lidar : Type the following command and check the output :

$ rostopic echo /robotis/sensor/scan --noarr

```
robotis@mpc:~$ rostopic echo /robotis/sensor/scan --noarr
header:
  seq: 403902
  stamp:
    secs: 1456227605
    nsecs: 320116247
  frame_id: lidar_link
angle_min: -2.35619449615
angle_max: 2.35619449615
angle_increment: 0.00436332309619
time_increment: 1.73611151695e-05
scan_time: 0.0250000003725
range_min: 0.0230000000447
range_max: 60.0
---
header:
  seq: 403903
  stamp:
    secs: 1456227605
    nsecs: 350245326
  frame_id: lidar_link
angle_min: -2.35619449615
angle_max: 2.35619449615
angle_increment: 0.00436332309619
time_increment: 1.73611151695e-05
scan_time: 0.0250000003725
range_min: 0.0230000000447
range_max: 60.0
```

Running Basic Program (18)

Check the Sensors

Run the PPC's Sensors (Web Camera (HD Camera), Depth Camera (RealSense))

Type the following command to run PPC's sensors :

$ roslaunch thormang3_sensors thormang3_sensors.launch

How to Check the PPC's Sensors

Web Camera : Type the following command and check the output.

$ rostopic echo /robotis/sensor/camera/image_raw --noarr

```
robotis@ppc:~$ rostopic echo /robotis/sensor/camera/image_raw --noarr
header:
  seq: 23
  stamp:
    secs: 1456228631
    nsecs: 803098057
  frame_id: head_p_link
height: 480
width: 640
encoding: rgb8
is_bigendian: 0
step: 1920
---
header:
  seq: 24
  stamp:
    secs: 1456228631
    nsecs: 903091604
  frame_id: head_p_link
height: 480
width: 640
encoding: rgb8
is_bigendian: 0
step: 1920
```

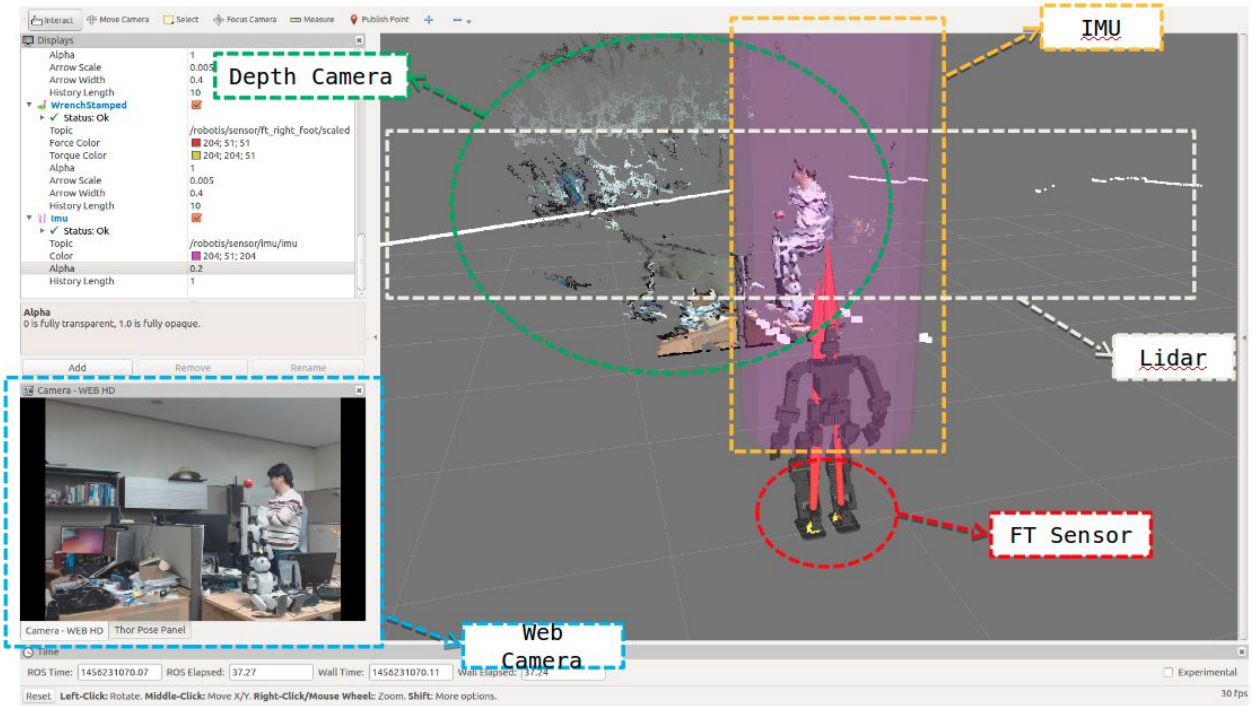
ROBOTIS

Running Basic Program (19)

Check the Sensors

How to Check the THORMANG's Sensors in the GUI

- Refer to Demo or Vision Presentations



Running Operating Program

How to execute OPC's GUI program

https://github.com/ROBOTIS-GIT/ROBOTIS-Documents/wiki/THORMANG3-Tutorials

PPC

1. *$ roscore*
3. *$ roslaunch thormang3_sensors thormang3_sensors.launch*

MPC

2. *# ./timesync*
3. *# roslaunch thormang3_manager thormang3_manager.launch*

OPC

2. *# ./timesync*
3. *$ roslaunch thormang3_description thormang3_opc.launch*
4. *$ roslaunch thormang3_demo thormang3_demo.launch*

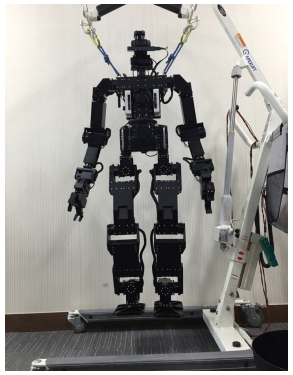
Powering Off THORMANG3

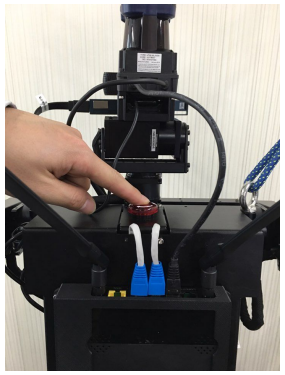
1. Hang THORMANG3 on lift.
   Lift THORMANG3's feet off the ground.

   First, shut down the MPC and the PPC
   by typing the following commands.
   MPC & PPC:

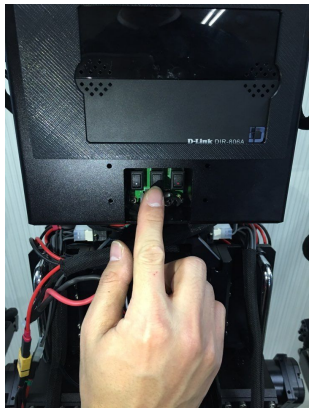
   $ sudo poweroff

   NOTE: If P/W is needed, P/W is 111111





2. Press the E-Stop Button to turn off the motor power.
3. After confirming that you have shut down the PCs,
   flip the three switches on the power board.



Q&A

Thank you for your attention !

Contact : Changhyun Sung, Ph.D.
sch@robits.com

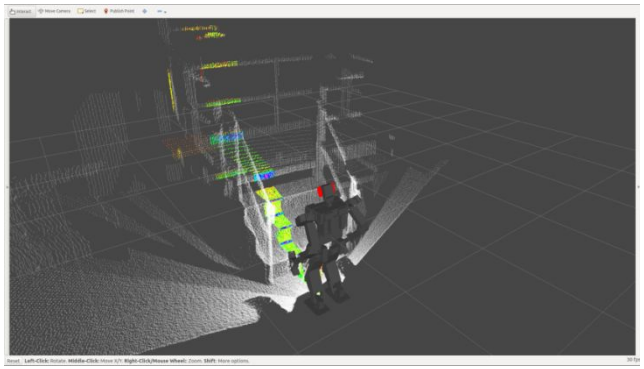
Agenda

- **THORMANG with the ROS**
- **ROBOTIS Framework**
  - Dynamixel
  - ROS Framework
- **THORMANG3 ROS Package**
  - Gazebo Simulation
  - Motion Modules for THORMANG3
- **ROS Framework Tutorial**
  - How to Create new Motion Module
  - How to Create new Sensor Module
  - How to Create new Robot Manager
- **Q&A**

THORMANG with the ROS

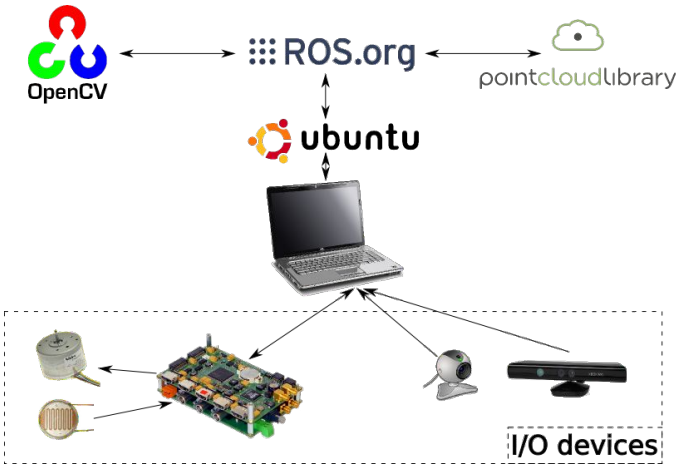
Flexible framework for writing robot software

- A Distributed, Modular Design
- A Vibrant Community
- Permissive Licensing
- A Collaborative Environment



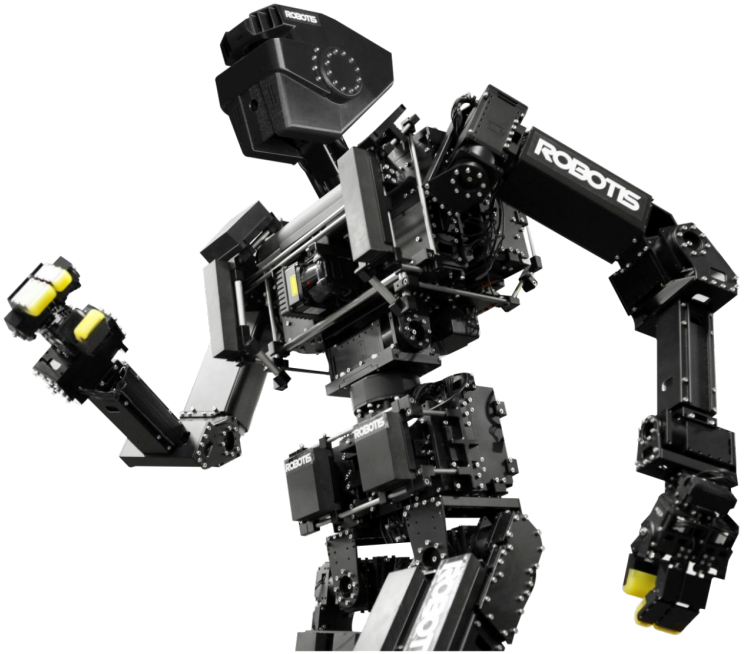
ROBOTIS

THORMANG with the ROS



Ecosystem based on the Vibrant Community

ROS Enabled



ROBOTIS with the ROS

Open Source Robotics Foundation

ROS Enabled

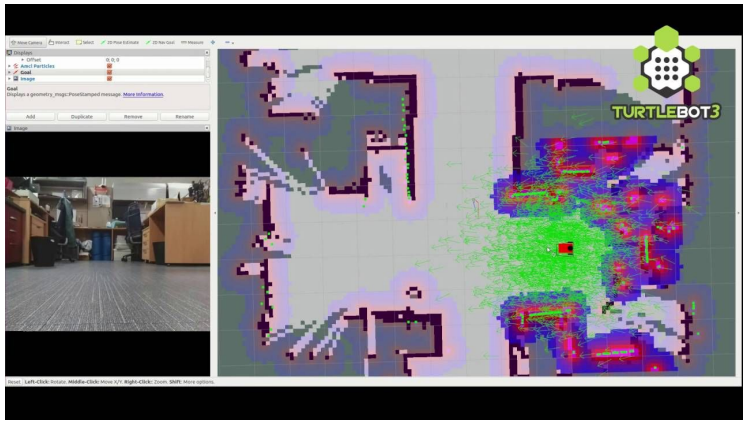
ROBOTIS

ROBOTIS GIT

ROBOTIS Official Github

#1505, Gasan Digital-1ro 145 (Ace High E... http://en.robotis.com/index/

Turtlebot3



Agenda

- THORMANG with the ROS
- **ROBOTIS Framework**
  - Dynamixel
  - ROS Framework
- THORMANG3 ROS Package
  - Gazebo Simulation
  - Motion Modules for THORMANG3
- ROS Framework Tutorial
  - How to Create new Motion Module
  - How to Create new Sensor Module
  - How to Create new Robot Manager
- Q&A

Dynamixel

Dynamixel

Dynamixel Protocol 1.0

(http://support.robotis.com/en/product/dynamixel/dxl_communication.htm)

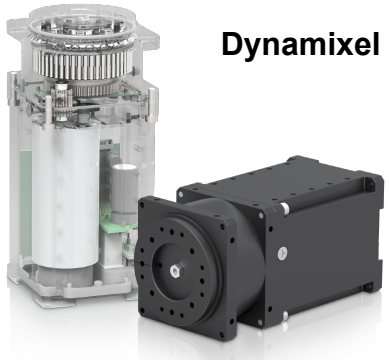
RS485/TTL

Dynamixel Protocol 2.0

(http://support.robotis.com/en/product/actuator/dynamixel_pro/communication.htm)

X-series

Dynamixel Pro



ROBOTIS

Dynamixel

Dynamixel SDK (https://github.com/ROBOTIS-GIT/DynamixelSDK)

The ROBOTIS Dynamixel SDK is a software development library
that provides Dynamixel control functions for packet communication.

(Example) Dynamixel Protocol 2.0

- **Instruction Packet**

0xFF 0xFF 0xFD	0x00	ID	LEN_L LEN_H	INST	Param1 … ParamN	CRC_L CRC_H
Header	Reserved	ID	Packet Length	Instruction	Parameter	16bit CRC

- **Instruction**
  - Ping
  - **Read / Write :** Data read / write command
  - Reg Write / Action
  - Factory Reset
  - Reboot
  - **Sync Write / Read :**
    Read / Write data from the same location and same size for multiple devices simultaneously
  - **Bulk Write / Read :**
    Read / Write data from the different locations and different sizes for multiple devices simultaneously

Dynamixel

Control Table

(http://support.robotis.com/en/product/dynamixel_pro/control_table.htm)

- Control Table consists of data regarding the current status and operation, which exists inside of Dynamixel.
- The user can control Dynamixel by changing data of Control Table via Instruction Packet.

EEPROM and RAM

Data in RAM area is reset to the initial value whenever the power is turned on while data in EEPROM area is kept once the value is set even if the power is turned off.

In EEPROM torque enable(562) can be written only if its value is 0.

Address

It represents the location of data.

To read or write data on Control Table, the user should assign the correct address in the Instruction Packet.

Access

Dynamixel has two kinds of data: Read-only data, which is mainly used for sensing, and Read-and-Write data, which is used for driving.

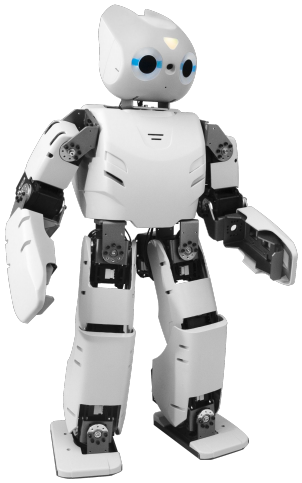
Size

Dynamixel PRO control table is 1-4 bytes.

ROBOTIS ROS Framework

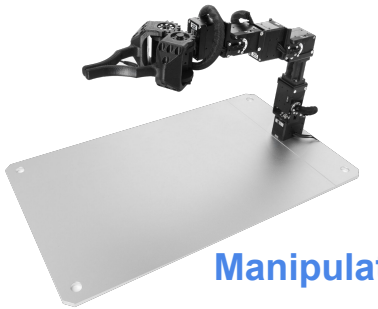
ROBOTIS Framework (https://github.com/ROBOTIS-GIT/ROBOTIS-Framework)

ROBOTIS-OP2

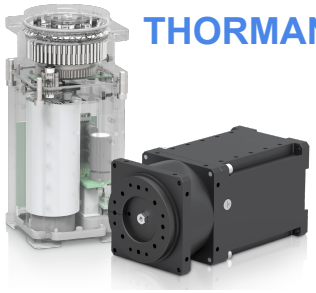


Turtlebot3

Manipulator-X



THORMANG3



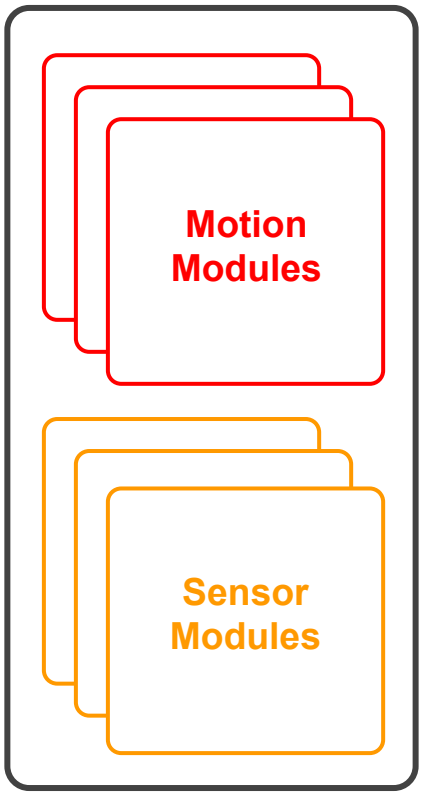
ROBOTIS ROS Framework

ROBOTIS-Framework

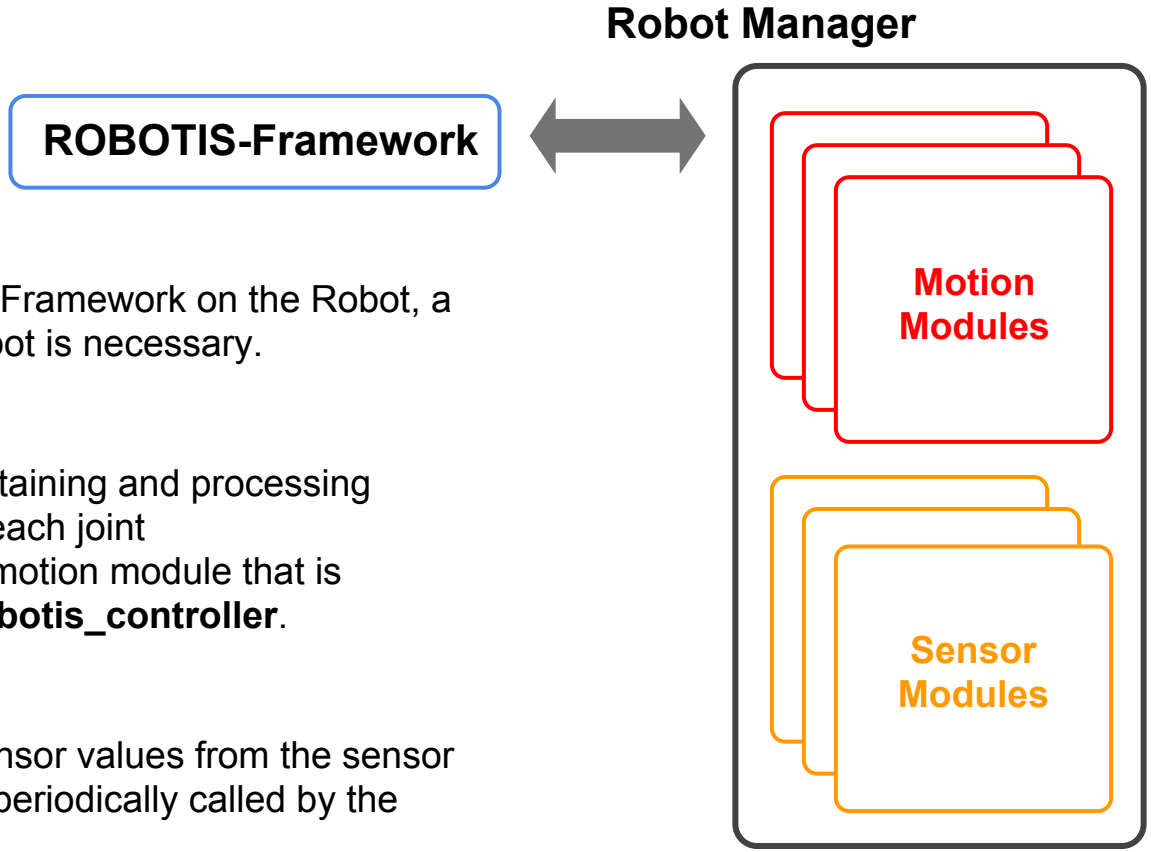
- **robotis_controller** :
  main package to control real robot & gazebo simulation
- **robotis_device** :
  package manages ROBOTIS motor device information
  (control table)
- **robotis_framework**
- **robotis_framework_common** :
  collection of headers commonly used in ROBOTIS framework

ROBOTIS-Framework-msgs

- **robotis_controller_msgs** :
  ROS messages used in the robotis_controller



ROBOTIS ROS Framework



Robot Manager

- In order to apply ROBOTIS Framework on the Robot, a specific manager for the robot is necessary.

Motion Modules

- The robot is operated by obtaining and processing calculated target values of each joint for the next move from the motion module that is periodically called by the **robotis_controller**.

Sensor Modules

- This package processes sensor values from the sensor module that is created and periodically called by the **robotis_controller**.

Agenda

- THORMANG with the ROS
- ROBOTIS Framework
  - Dynamixel
  - ROS Framework
- **THORMANG3 ROS Package**
  - Gazebo Simulation
  - Motion Modules for THORMANG3
- ROS Framework Tutorial
  - How to Create new Motion Module
  - How to Create new Sensor Module
  - How to Create new Robot Manager
- Q&A

THORMANG3 ROS Package

- **DynmixelSDK**
- **ROBOTIS-Framework**
- **ROBOTIS-Framework-msgs**
- **ROBOTIS-THORMANG-Common**
  URDF Model, Gazebo Simulation Environment
- **ROBOTIS-THORMANG-MPC**
  Robot Manager, Motion Modules, Sensor Modules
- **ROBOTIS-THORMANG-msgs**
- **ROBOTIS-THORMANG-OPC**

Gazebo Simulation (1)

THORMANG3 for Gazebo

$ roslaunch thormang3_gazebo robotis_world.launch

Robot manager for Gazebo

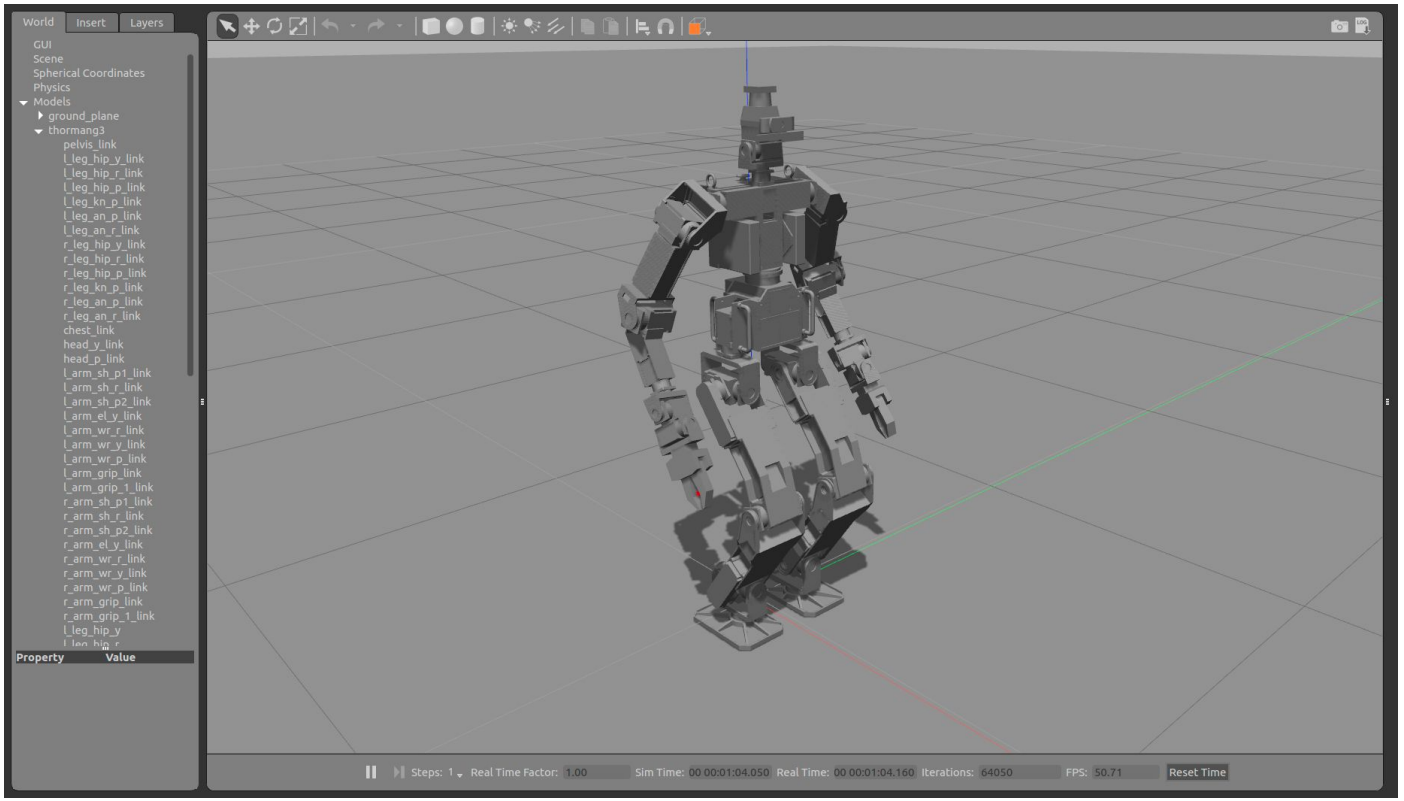
ROBOTIS-THORMANG-MPC/thormang3_manager/launch/thormang3_manager.launch
<param name="gazebo" value="true" type="bool"/>

$ roslaunch thormang3_manager thormang3_manager.launch
$ roslaunch thormang3_demo thormang3_demo.launch

Motion Modules for THORMANG3

- **ROBOTIS-THORMANG-MPC**
  - thormang3_base_module : Motion Module
  - thormang3_gripper_module : Motion Module
  - thormang3_head_control_module : Motion Module
  - thormang3_manipulation_module : Motion Module
  - thormang3_walking_module : Motion Module
  - thormang3_feet_ft_module : Sensor Module
  - thormang3_manager : Robot Manage
  - thormang3_kinematics_dynamics : Library
  - thormang3_balance_control : Library
  - ati_ft_sensor
  - thormang3_imu_3dm_gx4
- **ROBOTIS-THORMANG3-msgs**

Gazebo Simulation (2)



THORMANG3 Walking Module (1)

- **Representation of Pose**
  - The walking module uses global coordinate system
  - All pose is represented by *x, y, z, roll, pitch, yaw*
  - All unit is SI unit (meter and radian)

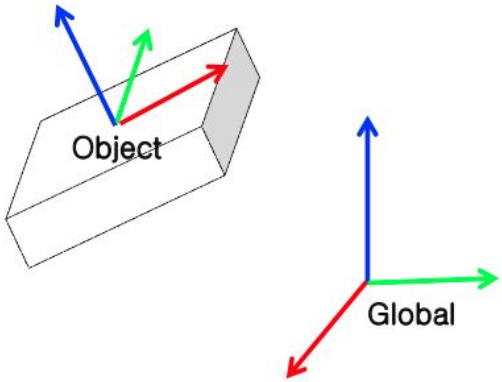
$$
\begin{aligned}
&{}^{G}T_{O} \\
&= {}^{G}t_{O}(x,y,z)\,{}^{G}R_{O}(\phi_{roll},\theta_{pitch},\psi_{yaw}) \\
&= t(x,y,z)R_{z}(\psi)R_{y}(\theta)R_{x}(\phi) \\
&= \begin{bmatrix} c\psi c\theta & c\psi s\theta s\phi - s\psi c\phi & c\psi s\theta c\phi + s\psi s\phi & x \\ s\psi c\theta & s\psi s\theta s\phi + c\psi c\phi & s\psi s\theta c\phi - c\psi s\phi & y \\ -s\theta & c\theta s\phi & c\theta c\phi & z \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}
$$

$\boldsymbol{G}$: Global Coordinate

$\boldsymbol{O}$: Object Cooridinate

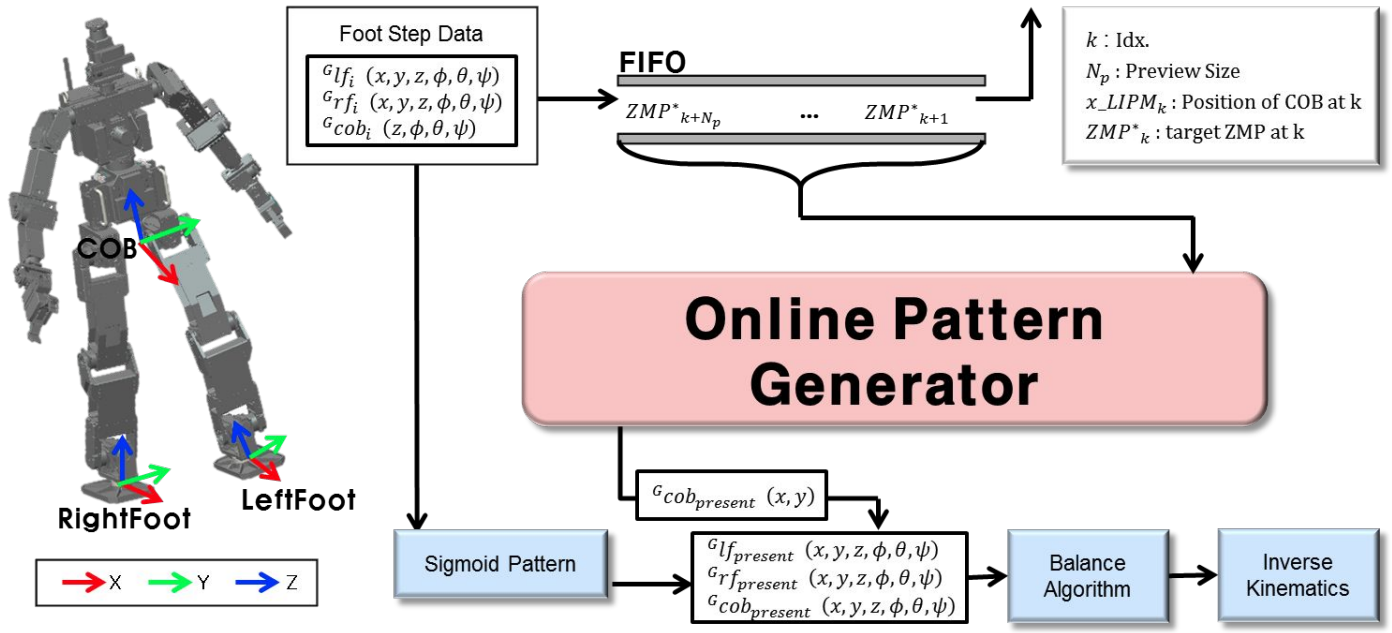
$c\phi$: $\cos\phi$

$s\phi$: $\sin\phi$



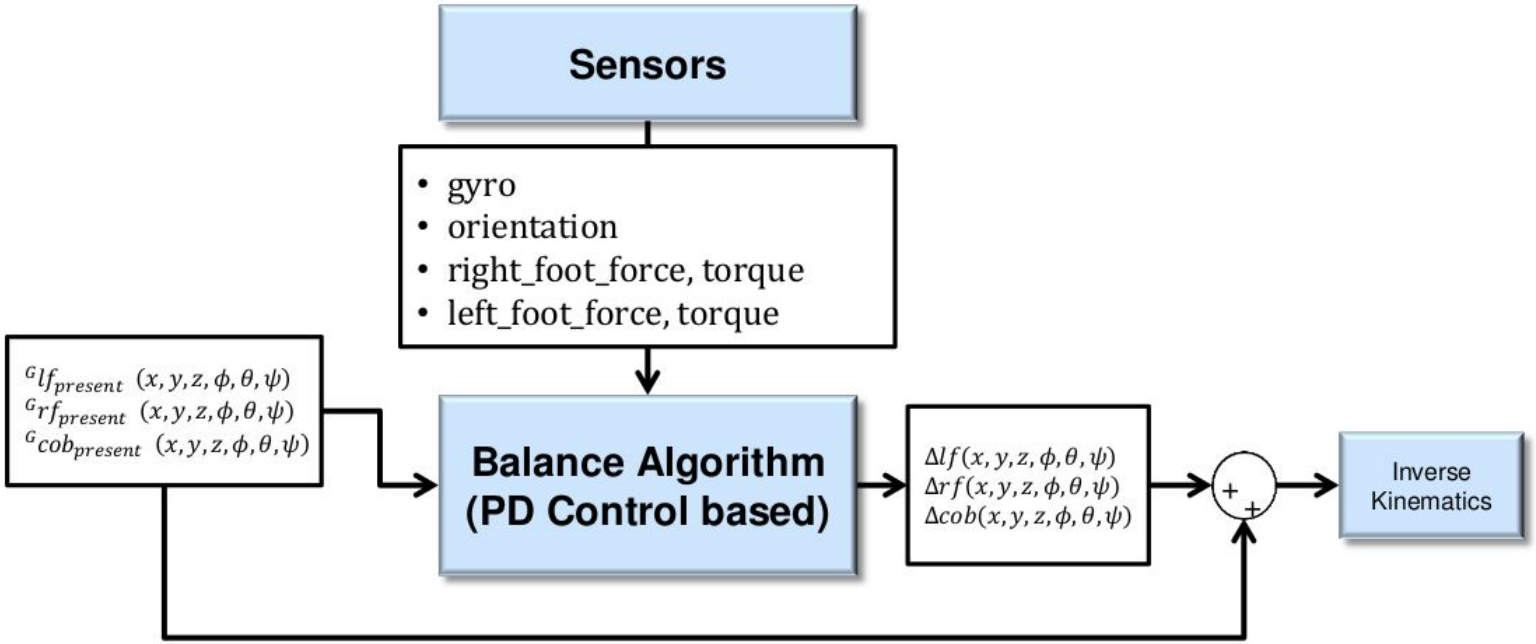
THORMANG3 Walking Module (2)

- **Pattern Generation**
  - COB(Center of Body) is at the middle of hip joints.
  - LF(Left Foot) and RF(Right Foot) is at the middle of each feet.



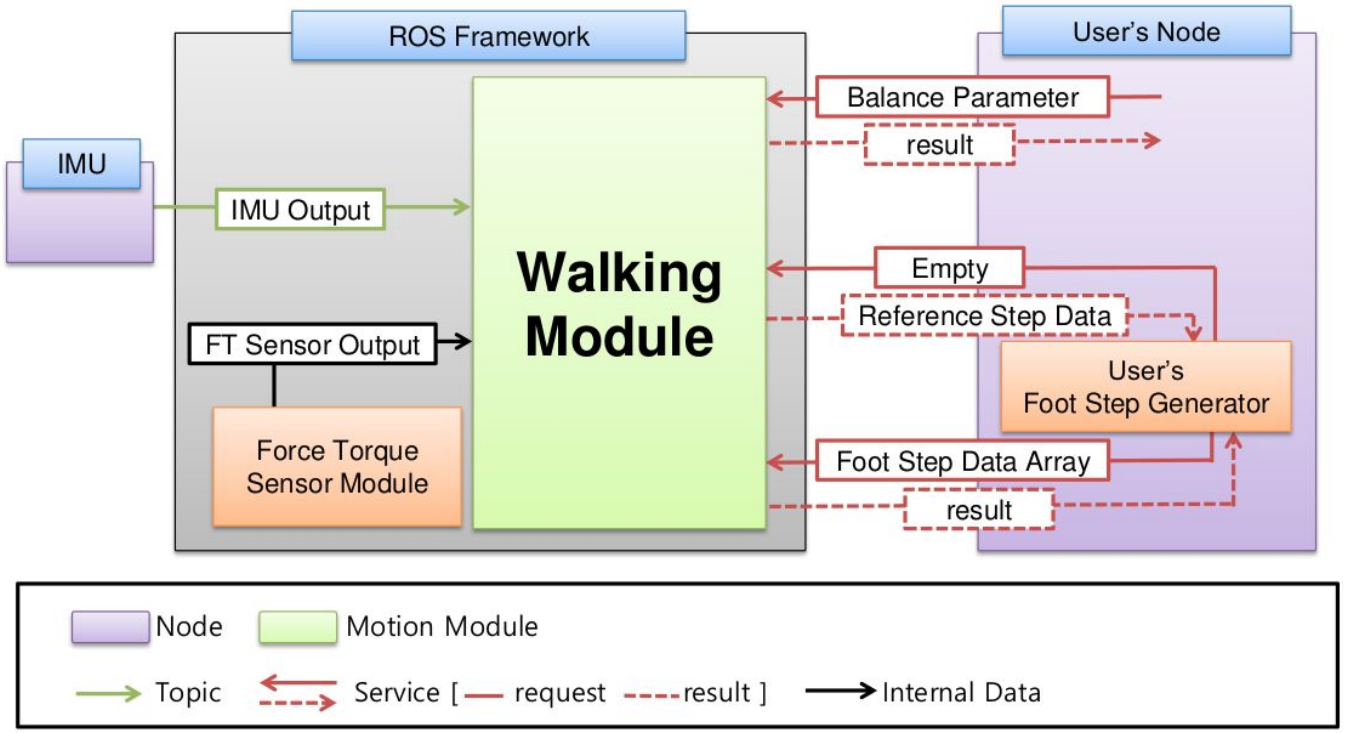
THORMANG3 Walking Module (3)

- **Balance Algorithm**



THORMANG3 Walking Module (4)

- **Overview**



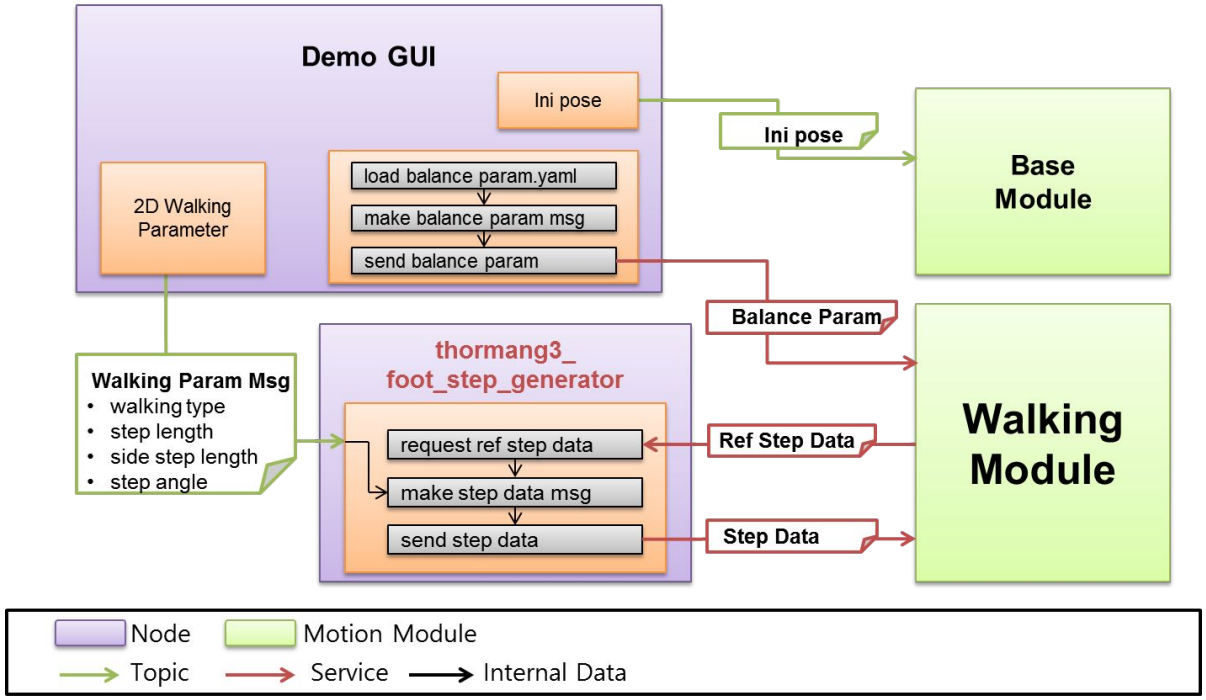
THORMANG3 Walking Module (5)

- **Topic and Service List**

	Name		Description
Topic (Publish)	/robotis/walking/status_message		The status message from walking module
Service (Server)	/robotis/walking/get_reference_step_data	req	Empty
		res	Reference Step Data
	/robotis/walking/add_step_data	req	"Auto Start" and "Step Data Array"
		res	Processing Result for Request
	/robotis/walking/walking_start	req	Empty
		res	Processing Result for Request
	/robotis/walking/remove_existing_step_data	req	Empty
		res	Processing Result for Request
	/robotis/walking/set_balance_param	req	All of Desired Balancing Parameter
		res	Processing Result for Request
	/robotis/walking/is_running	req	Empty
		res	Running or Not

THORMANG3 Walking Module (6)

- **Example**
  - Balance on/off
  - Add step data



THORMANG3 Walking Module (7)

- **Example**
  - Balance on/off

/robotis/walking/set_balance_param			
	Variable Type	Variable Name	Description
Request	`BalanceParam`	`balance_param`	All of parameter for balancing algorithm
Result	`int32`	`result`	The processing result of "balance_param" 0x00 : There is no error. 0x02 : The walking module is not enable. 0x20 : Previous request is not finished 0x40 : The time constant is zero or negative.

THORMANG3 Walking Module (8)

- **Example**
  - Add step data

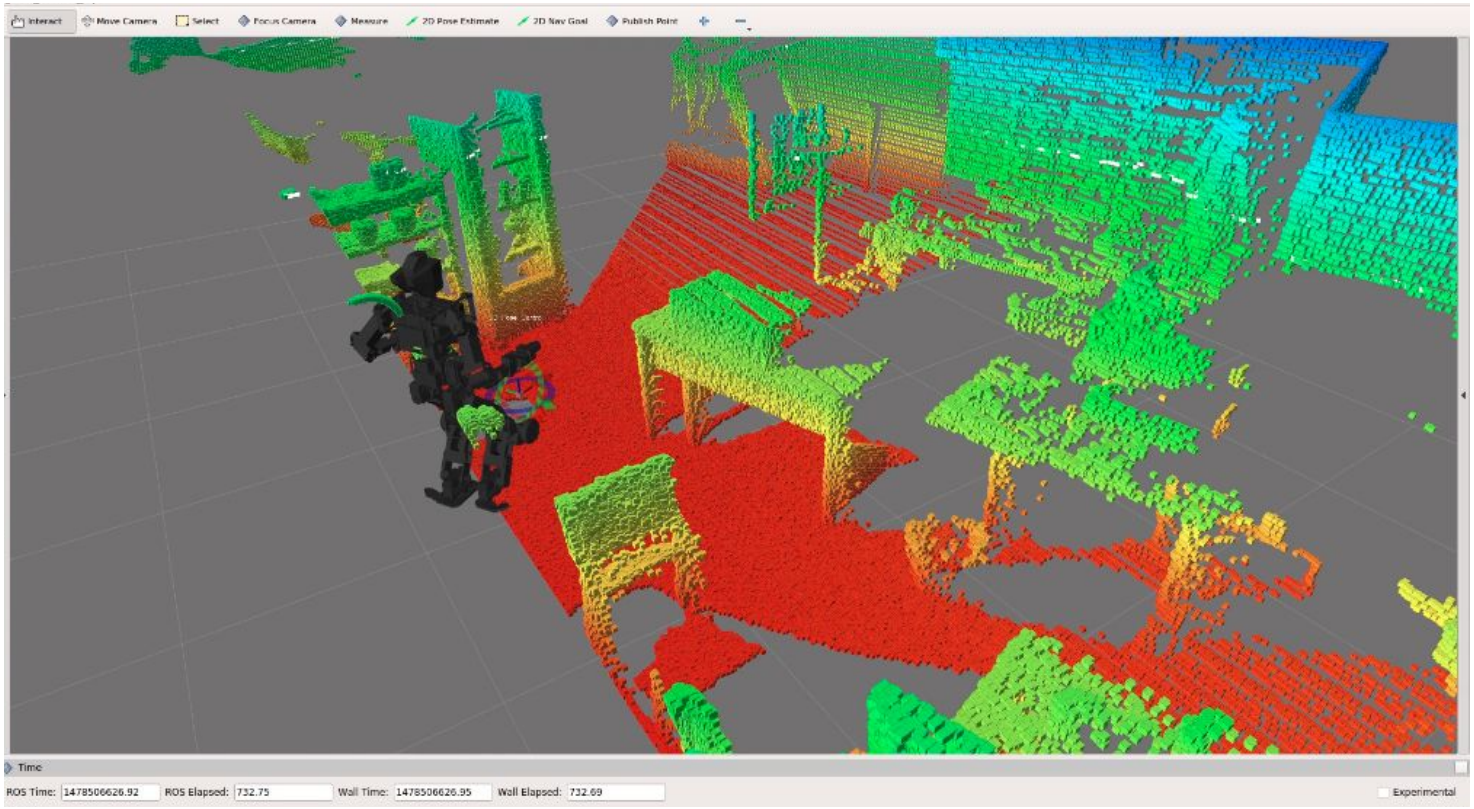
/robotis/walking/add_step_data			
	Variable Type	Variable Name	Description
Request	bool	auto_start	If "auto_start" is true, the robot will start walking automatically.
	bool	remove_existing_step_data	if "remove_existing_step_data" is true, all of the previous added step data are removed.
	StepData[]	step_data_array	step data array of user specified
Result	int32	result	The processing result of "add_step_data" 0x00 : There is no error. 0x02 : The walking module is not enable. 0x04 : There is some problem in step time data. 0x08 : There is some problem in step position data. 0x400 : The robot is walking now.

THORMANG3 Walking Module (9)

- **Example**

THORMANG3 Whole-body Module

THORMANG3 Localization Package



Agenda

- THORMANG with the ROS
- ROBOTIS Framework
  - Dynamixel
  - ROS Framework
- THORMANG3 ROS Package
  - Gazebo Simulation
  - Motion Modules for THORMANG3
- **ROS Framework Tutorial**
  - How to Create new Motion Module
  - How to Create new Sensor Module
  - How to Create new Robot Manager
- Q&A

ROS Framework Tutorial

https://github.com/ROBOTIS-GIT/ROBOTIS-Documents/wiki/Tutorials

- How to Create new Motion Module
- How to Create new Sensor Module
- How to Create new Robot Manager

How to Create new Motion Module

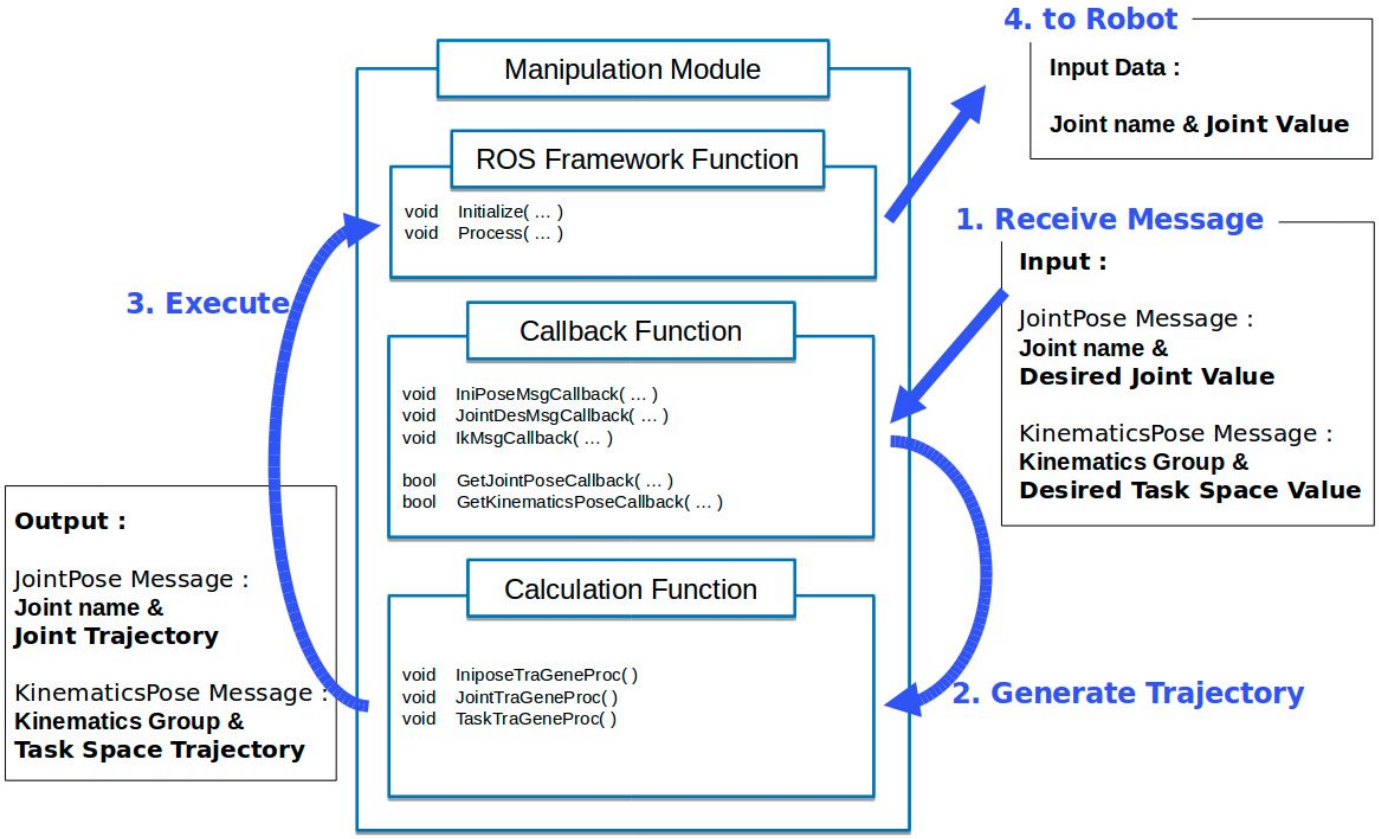
https://github.com/ROBOTIS-GIT/ROBOTIS-Documents/wiki/Creating-new-motion-module

- Motion Modules calculate **target position (velocity, torque)** of each joint.
- Motion Module is based on the Singleton Pattern.

- ***initialize()***
  - Function is called only once when the Robot Manager registers Motion Module to the ***RobotisController***.
- ***process()***
  - *process()* function that is periodically called by the ***RobotisController***.
- **stop() , isRunning()**
  - Function that checks the status of Motion Module is required along with the function that stops the Motion Module.

How to Create new Motion Module

Example Module



How to Create new Sensor Module

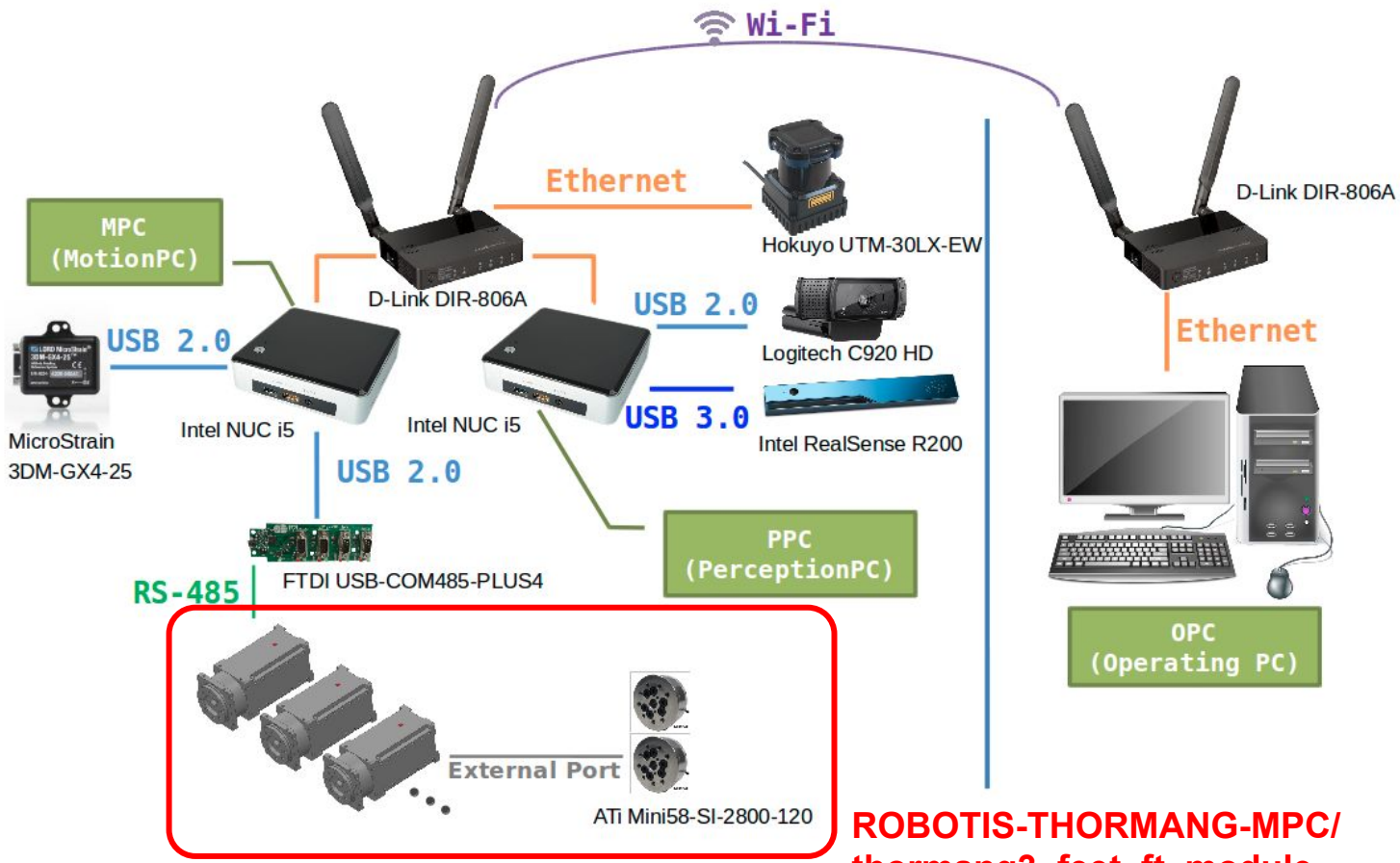
https://github.com/ROBOTIS-GIT/ROBOTIS-Documents/wiki/Creating-new-sensor-module

- Sensor Module processes obtained values from the sensor module.
- Once the ***RobotisController*** completes calculation for all registered Sensor Modules, it transfers the result to the Motion Module.
- Sensor Module is based on the Singleton Pattern.

- ***initialize()*** **:**
  - Function is called only once when the Robot Manager registers Motion Module to the ***RobotisController***.
- ***process()*** :
  - *process()* function that is periodically called by the ***RobotisController***.
  - After Bulk Reading current and target values from the argument ***dxls***, calculated values are saved to the result.

How to Create new Sensor Module

https://github.com/ROBOTIS-GIT/ROBOTIS-Documents/wiki/Creating-new-sensor-module



ROBOTIS-THORMANG-MPC/ thormang3_feet_ft_module

How to Create new Robot Manager

https://github.com/ROBOTIS-GIT/ROBOTIS-Documents/wiki/Creating-new-robot-manager

- The Robot Manager is a package to apply ROBOTIS Framework to a new robot.
- Configuration files that contain hardware information of the robot and setting values are necessary in order to use ROBOTIS Framework.
  - Robot information file (**.robot**)
  - Joint initialize file (**.yaml**)
  - Offset file (**.yaml**)

How to Create new Robot Manager

https://github.com/ROBOTIS-GIT/ROBOTIS-Documents/wiki/Creating-new-robot-manager

- Robot Information file (**.robot**)
  - port info
    - PORT NAME : File name of the port
    - BAUD RATE : Default baudrate for the port
    - DEFAULT JOINT :
      The default joint to obtain information such as Control table and Protocol Version when each port performs SyncWrite/BulkRead.
  - device info
    - TYPE : Type of the device (dynamixel, sensor)
    - PORT NAME : File name of the port connected to the device
    - ID : Allocated ID to the device
    - MODEL : Model name of the device
    - PROTOCOL : Protocol version to control the device
    - DEV NAME : Name of the device to be used for control
    - BULK READ ITEMS : The item list to be read in bulk from the device (',' is a field delimiter)

How to Create new Robot Manager

https://github.com/ROBOTIS-GIT/ROBOTIS-Documents/wiki/Creating-new-robot-manager

- Joint initialize file (**.yaml**)

```
JOINT_NAME1 :
        CTRL_TABLE_ITEM_NAME1 : VALUE
        CTRL_TABLE_ITEM_NAME2 : VALUE
```

Joint initialize file example

```
r_arm_sh_p1 :   # H54-100-S500-R
        return_delay_time     :  10    # item name : value
        operating_mode        :  3
        shutdown              :  58
        homing_offset         :  0
        torque_limit          :  310
        max_position_limit    :  250950
        min_position_limit    : -250950
        goal_torque           : 310
        goal_velocity         : 0
        goal_acceleration     : 0
        position_p_gain       : 32
        velocity_p_gain       : 180
        velocity_i_gain       : 452
```

How to Create new Robot Manager

https://github.com/ROBOTIS-GIT/ROBOTIS-Documents/wiki/Creating-new-robot-manager

- Offset file (**.yaml**)

  Joint initialize file example

```
offset :
        head_p : 0
        head_y : 0
init_pose_for_offset_tuner:
        head_p: 0
        head_y: 0
```

Q&A

Thank you for your attention !

Contact : Changhyun Sung, Ph.D.
sch@robits.com