Software version : V1.14
Document version : 1.14
Exported from Doosan Robotics Online Manual Center(en_US)

ROS manual(V1.14)

M0609 | M0617 | M1013 | M1509 | H2017 | H2515 A0509 | A0509S | A0912 | A0912S

© 2022 Doosan Robotics Inc.

Table of Contents

1 Preface

Doosan robotics ROS package is a metapackage for running Doosan cooperative robots on ROS URDF model is provided, simulation is possible through Rviz, Gazebo, and the real robot can be driven through moveIt or various examples.

The contents of this manual are current as of the date this manual was written, and product-related information may be modified without prior notification to the user.

For more information on the revised manual, please visit our Robot LAB website. (https://robotlab.doosanrobotics.com/)

1.1 Copyright

The copyright and intellectual property rights of the contents of this manual are held by Doosan Robotics. It is therefore prohibited to use, copy, or distribute the contents without written approval from Doosan Robotics. In the event of abuse or modification of the patent right, the user will be fully accountable for the consequences.

While the information in this manual is reliable, Doosan Robotics will not be held accountable for any damage that occurs due to errors or typos. The contents of this manual may be modified according to product improvement without prior notification.

© 2022 Doosan Robotics Inc., All rights reserved

2 Installation Doosan ROS package

2.1 Overview

2.1.1 Doosan robotics ROS package

Doosan robotics ROS package is a metapackage for running Doosan cooperative robots on ROS URDF model is provided, simulation is possible through Rviz, Gazebo, and the real robot can be driven through moveIt or various examples.

2.2 Prerequisitese

2.2.1 System

You must use an x86 system.

We recommend a workstation-class PC for the best simulation.

2.2.2 OS & Distro

Ubuntu 16.04(32/64bit) + ROS kinetic or melodic

2.3 Installation

2.3.1 Install the source from github

Download and build the source from Doosan robotics

Github : **https://github.com/doosan-robotics/doosan-robot**

2.3.2 How to Install

```
### We recoomand the /home/<user>/catkin_ws/src
$ mkdir -p /home/<user>/catkin_ws/src
$ cd /home/<user>/catkin_ws/src
$ catkin_init_workspace
$ git clone https://github.com/doosan-robotics/doosan-robot
$ rosdep install --from-paths doosan-robot --ignore-src --rosdistro
kinetic -r -y
$ catkin_make
$ source ./devel/setup.bash
# In this manual, workspace was used as '~/catkin_ws'
```

3 Operation mode

3.1 Virtual mode

3.1.1 Feature

- If you are driving without a real robot, use virtual mode.
- Selecting virtual mode sets the mode argument to virtual when running the dsr_aluncher launch file. If you omit the argument, it defaults to virsual.
  ex> roslaunch dsr_launcher single_robot_gazebo.launch **mode:=virtual**
- When ROS launches in virtual mode, the emulator(DRCF) runs automatically.
  - (DRCF) location: doosan-robot/common/bin/ DRCF
- One emulator is required for each robot.
- When controlling multiple robots, the emulator will automatically run as many as the number of robots and use different port.

3.2 Real mode

3.2.1 Feature

- Use real mode to drive a real robot.
- In real mode operation, communication must be established with the real robot controller.
- The default IP of the robot controller is 192.168.127.100 and the port is 12345.
- Selecting arguments(**mode:=real host:=192.168.127.100 port:=12345**) to real mode when running the dsr_launcher launch file.
  ex> roslaunch dsr_launcher single_robot_gazebo.launch **mode:=real host:=192.168.127.100 port:=12345**

3.2.2 Connect with Controller

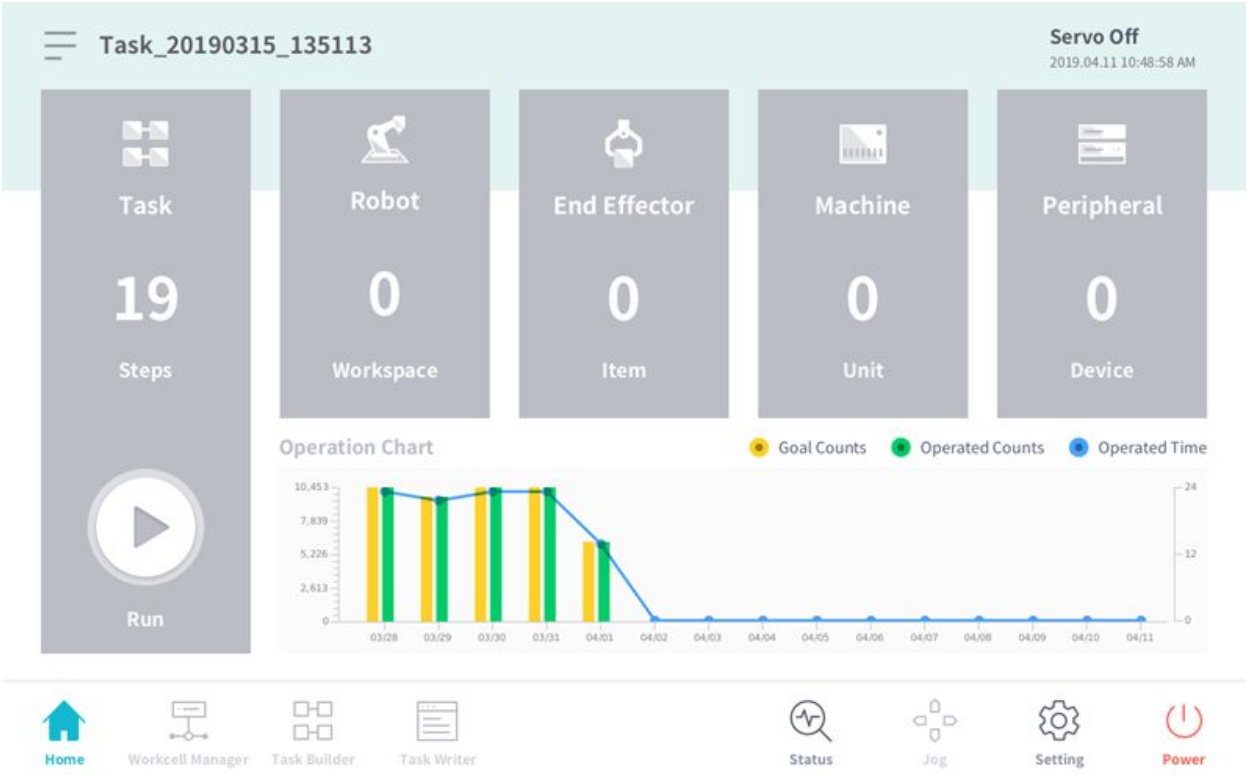


Figure 2.2 Teach Pandaunt Screen

- The user can set static IP in Setting -> Network of TP screen.

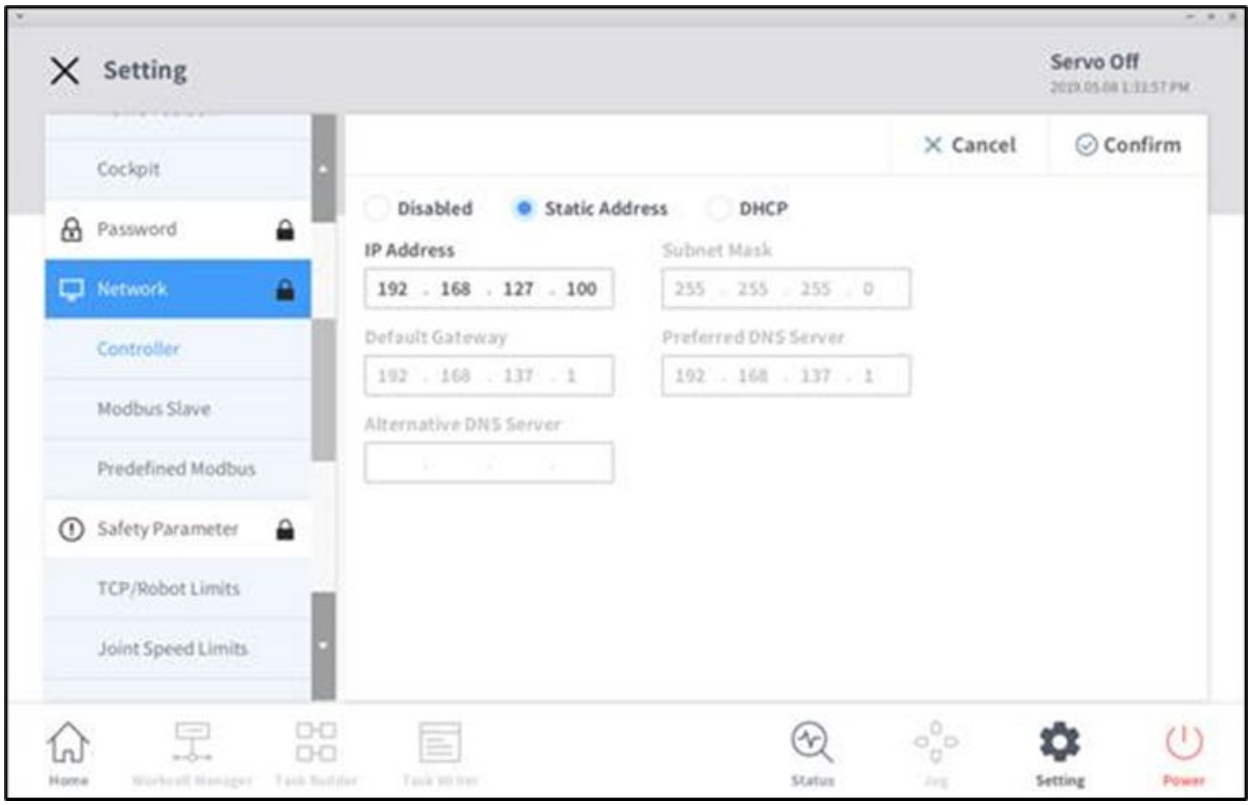


Figure 2.3 Check robot controller IP on TP

- Check the IP of the controller set in the Network tab, and set this IP in the ROS (host := ROBOT_IP)
- If the ROS control node is correctly executed, ROS has the robot control.
- If the TP transfer control, below pop-up message on the TP screen.

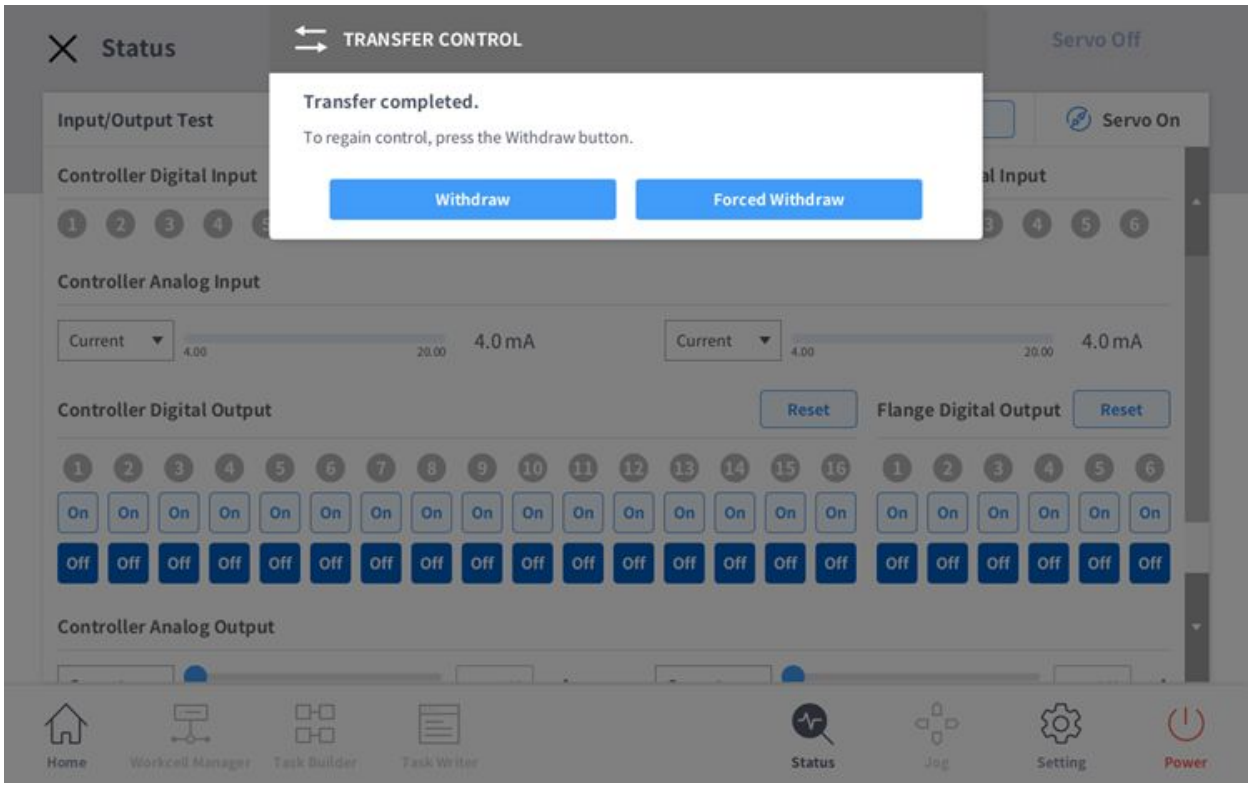


Figure 2.4 Transfer Control pop-up message

4 dsr_description

4.1 dsr_description <robot_model>.launch

4.1.1 Feature

- Launch the robot model on Rviz simulator and load Joint_state_publisher.
- The robot can be moved by using Joint_state_publisher.

4.1.2 Parameter

Parameter Name	Data Type	Default Value	Description
model	-	m1013	M-Series Robot model . m0609, m0617, m1013, m1509 A-Series Robot model . a0509
color	-	white	Robot Color . white or blue
gripper	-	none	using gripper or not . none : not use gripper . robotiq_2f : use robotiq 2finger gripper

4.1.3 Example

```
$ roslaunch dsr_description m0609.launch
$ roslaunch dsr_description m1013.launch color:=blue # Change Color
$ roslaunch dsr_description m1509.launch gripper:=robotiq_2f # insert
robotiq gripper
$ roslaunch dsr_description m0617.launch color:=blue gripper:=robotiq_2f
$ roslaunch dsr_description a0509.launch # A-Series
```

Robot and Joint_state_publisher are loaded on Rviz simulator (Figure 3.1). The robot can be moved by using Joint_state_publisher.

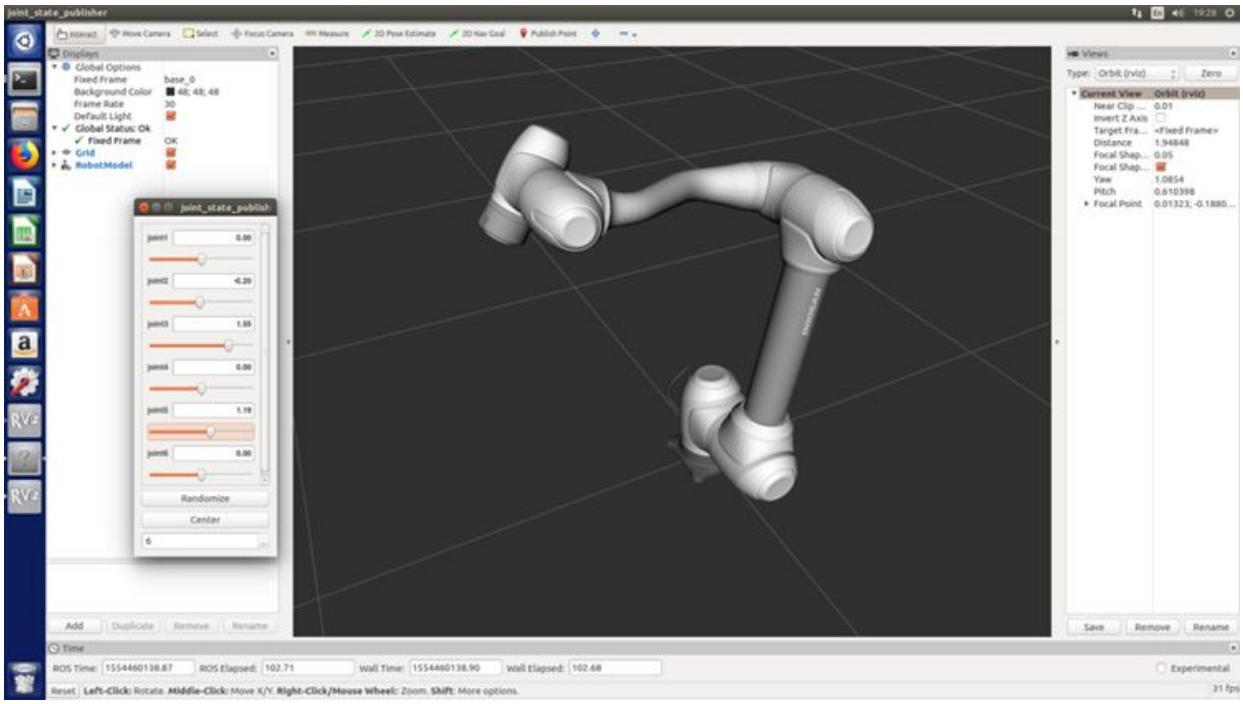


Figure 3.1 Robot on the Rviz simulator

5 dsr_moveit_config related

5.1 dsr_moveit_config

5.1.1 Feature

- Launch the robot model on Rviz simulator and operated by MoveIt.
- It only works in simulation mode.

5.1.2 Parameter

Parameter Name	Data Type	Default Value	Description
color	-	white	Robot Color . white or blue

5.1.3 Example

```
$ roslaunch moveit_config_m0609 m0609.launch
$ roslaunch moveit_config_m0617 m0617.launch
$ roslaunch moveit_config_m1013 m1013.launch color:=blue
$ roslaunch moveit_config_m1509 m1509.launch
$ roslaunch moveit_config_a0509 a0509.launch
```

Robot and Motion Planning Interface window are loaded on Rviz(Figure 4.1)

MotionPlanning allows the robot to run virtually.

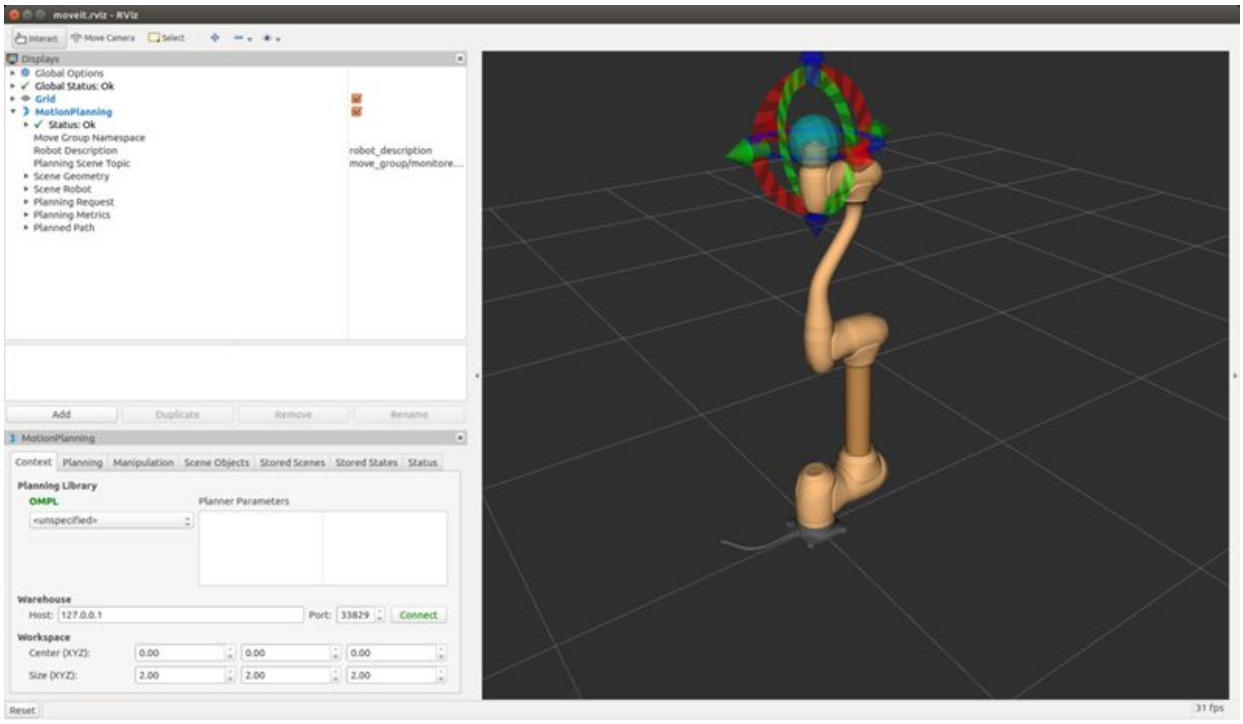


Figure 4.1 Rviz + MoveIt

5.2 dsr_control + moveit

5.2.1 Feature

- Launch the robot model on Rviz simulator and operated by MoveIt.
- Working by connecting with emulator mode or real robot.
- The emulator mode only works in virtual mode.

5.2.2 Parameter

Paramater Name	Paramater Name	Default Value	Description
host	-	127.0.0.1	Robot Controller IP . Emulator : 127.0.0.1 . Real robot controller : 192.168.127.100
Port	-	12345	port

Paramater Name	Paramater Name	Default Value	Description
mode	-	virtual	Robot operation mode - virtual : virtual mode - real : real mode
model	-	m1013	M-Series Robot model . m0609, m0617, m1013, m1509 A-Series Robot model . a0509
color	-	white	Robot color . white or blue
gripper	-	none	using gripper or not . none : not use gripper . robotiq_2f : use robotiq 2finger gripper

5.2.3 Example

```
<virtual mode>
$ roslaunch dsr_control dsr_moveit.launch model:=m0609 mode:=virtual
$ roslaunch dsr_control dsr_moveit.launch model:=m0617
$ roslaunch dsr_control dsr_moveit.launch model:=m1013 mode:=virtual
color:=blue

<real mode>
Robot controller IP defalut = 192.168.127.100, port = 12345
$ roslaunch dsr_control dsr_moveit.launch model:=m1509
host:=192.168.127.100 mode:=real color:=blue gripper:=robotiq_2f
$ roslaunch dsr_control dsr_moveit.launch model:=a0509
host:=192.168.127.100 mode:=real
```

The Robot and Motion Planning Interface window are loaded on Rviz(Figure 4.2).

The MotionPlanning allows the robot to run in real environment.

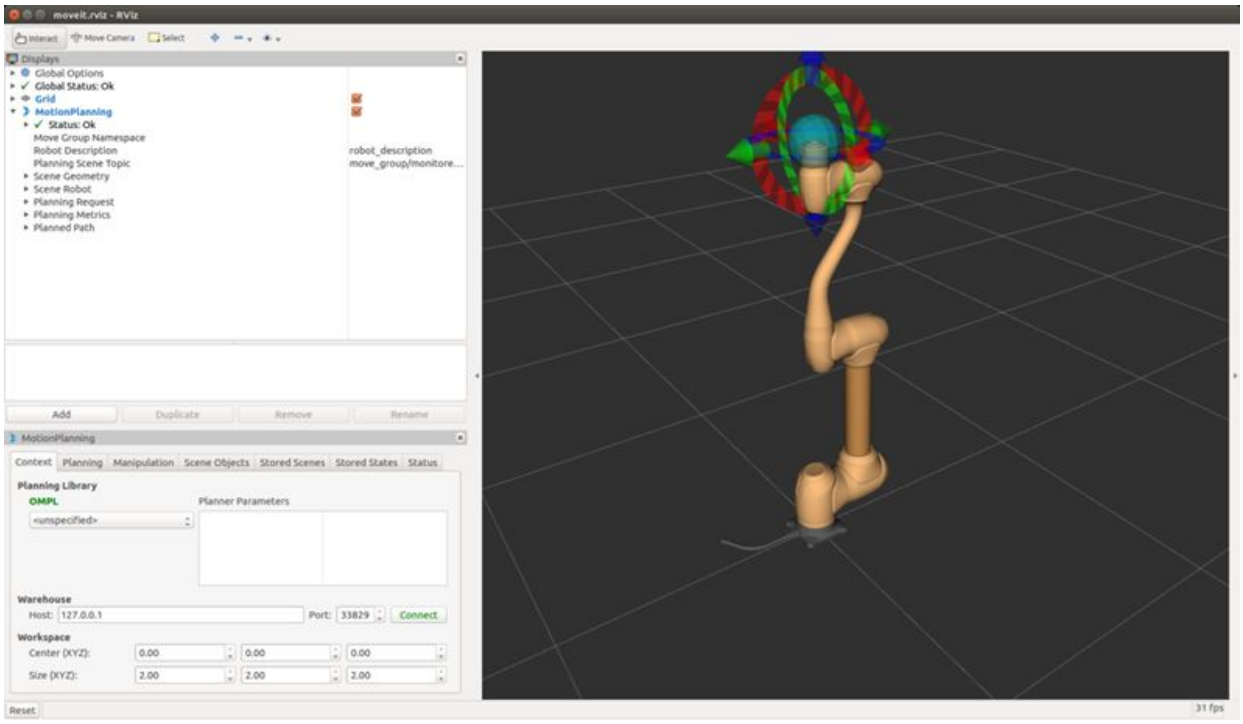


Figure 4.2 Rviz + dsr_control

5.3 MoveIt Commander

5.3.1 Feature

- Control the robot through Moveit Commander.

5.3.2 Install

- sudo apt-get insatll ros-kinetic-moveit-commander

5.3.3 Example

```
<virtual mode>
$ roslaunch dsr_control dsr_moveit.launch model:=m1013 mode:=virtual

$ ROS_NAMESPACE=/dsr01m1013 rosrun moveit_commander
moveit_commander_cmdline.py robot_description:=/dsr01m1013/
robot_description

> use arm
> goal0 = [0 0 0 0 0 0]      # save the home position to variable "goal0"
> goal1 = [0 0 1.57 0 1.57 0] # save the target position to varialbe
"goal1" / radian
```

```
> go goal1          # plan & excute (the robot is going to move target position)
> go goal0          # paln & excute (the robot is going to move home position)


<real mode>
Robot controller IP defalut = 192.168.127.100, port = 12345
$ roslaunch dsr_control dsr_moveit.launch model:=a0509 host:=192.168.127.100 mode:=real

$ ROS_NAMESPACE=/dsr01a0509 rosrun moveit_commander moveit_commander_cmdline.py robot_description:=/dsr01a0509/robot_description

> use arm
> goal0 = [0 0 0 0 0 0]      # save the home position to variable "goal0"
> goal1 = [0 0 1.57 0 1.57 0] # save the target position to varialbe "goal1" / radian
```

6 dsr_launcher

6.1 Feature

- Configure various robot environment through dsr_launcher.
- The user can build Single Robot, Multi Robot, Gripper, mobile environment according to Paramater.
- Multi Robot configuration is an extension of Single Robot configuration. This is an example of 2 robots of Multi Robot configuration basically provided in this package. Please modify the dsr_launcher / multi-robot * .launch files to the appropriate configuration for your environment. (ns, host, port, model, etc. should be corrected correctly.)
- After loading dsr_launcher, run dsr_example for each environment with rosrun.
  (For details, see Chapter 6, dsr_example.)

6.2 Parameter

Parameter Name	Data type	Default Value	Description
ns	-	dsr01	ROBOT name space . single robot : dsr01 . multi robot: dsr01, dsr02, dsr03, dsr04 …
host	-	127.0.0.1	Robot controller IP . Emulator : 127.0.0.1 . Real robot controller : 192.168.127.100
port	-	12345	port
mode	-	virtual	Robot operation mode - virtual : virtual mode - real : real mode
model	-	m1013	M-Series Robot model . m0609, m0617, m1013, m1509 A-Series Robot model . a0509
color	-	white	Robot color . white or blue

Parameter Name	Data type	Default Value	Description
gripper	-	none	using gripper or not . none : not use gripper . robotiq_2f : use robotiq 2finger gripper
mobile	-	none	using mobile robot or not . none : not use mobile robot . husky : use husky mobile robot

6.3 Examples

```
<single robot>
- rviz, virtual mode, m1013(white)
$ roslaunch dsr_launcher single_robot_rviz.launch mode:=virtual
model:=m1013
- gazebo, real mode, m1013(blue)
$ roslaunch dsr_launcher single_robot_gazebo.launch host:=192.168.127.100
port:= 12345 mode:=real color:=bule
- rviz + gazebo, real mode, m1013(white)
$ roslaunch dsr_launcher single_robot_rviz_gazebo.launch
host:=192.168.127.100 port:= 12345 mode:=real
- + gripper
$ roslaunch dsr_launcher single_robot_rviz_gazebo.launch
gripper:=robotiq_2f
- + mobile
$ roslaunch dsr_launcher single_robot_rviz_gazebo.launch
gripper:=robotiq_2f mobile:=husky

<multi robot>
- rviz, virtual mode, m1013 x 2
$ roslaunch dsr_launcher multi_robot_rviz.launch
- gazebo, virtual mode, m1013 x 2
$ roslaunch dsr_launcher multi_robot_gazebo.launch
- rviz + gazebo, virtual mode, m1013 x 2
$ roslaunch dsr_launcher multi_robot_rviz_gazebo.launch
```

7 dsr_example

Provides an example of robot operation according to robot environment configured through dsr_launcher. Please refer to Chapter 5, "dsr_launcher" for detailed robot environment configuration.

The example files was written by python.

- Directory of .py files: ~/catkin_ws/src/doosan-robot/dsr_example/py/scripts

7.1 Single Robot

7.1.1 Feature

- Provides an example of operating a Single Robot.
  (Please refer to Chapter 5, "dsr_launcher" for detailed robot environment configuration.)
- The example files was written in python
  - Directory of .py files: ~/catkin_ws/src/doosan-robot/dsr_example/py/scripts/simple

7.1.2 Paramaters of dsr_launcher

Parameter Name	Data type	Default Value	Description
ns	-	dsr01	ROBOT name space . single robot : dsr01 . multi robot: dsr01, dsr02, dsr03, dsr04 …
host	-	127.0.0.1	Robot controller IP . Emulator : 127.0.0.1 . Real robot controller : 192.168.127.100
port	-	12345	port
mode	-	virtual	Robot operation mode - virtual : virtual mode - real : real mode

Parameter Name	Data type	Default Value	Description
model	-	m1013	M-Series Robot model . m0609, m0617, m1013, m1509 A-Series Robot model . a0509
color	-	white	Robot color . white or blue
gripper	-	none	using gripper or not . none : not use gripper . robotiq_2f : use robotiq 2finger gripper
mobile	-	none	using mobile robot or not . none : not use mobile robot . husky : use husky mobile robot

7.1.3 Example

```
1. Robot controller default IP/Port
- IP : 192.168.127.100 , port = 12345

2. launch
Run dsr_launher for your desired configuration.
- single robot in rviz, virtual mode, m1013(white)
$ roslaunch dsr_launcher single_robot_rviz.launch mode:=virtual
model:=m1013 color:=white
- single robot in gazebo, real mode, m1013(blue)
$ roslaunch dsr_launcher single_robot_gazebo.launch mode:=real
host:=192.168.127.100 model:=m1013 color:=blue
- single robot in rviz + gazebo, virtual mode, m1013(white)
$ roslaunch dsr_launcher single_robot_rviz_gazebo.launch model:=m1013
color:=white

3. run application node
- Edit example files
   . Open the example file you want to run and modify the ROBOT_ID and
ROBOT_MODEL accordingly.
   .. ex>
ROBOT_ID = "dsr01"
ROBOT_MODEL = "m1013"

```

```
20  $ rosrun dsr_example_py single_robot_simple.py
```

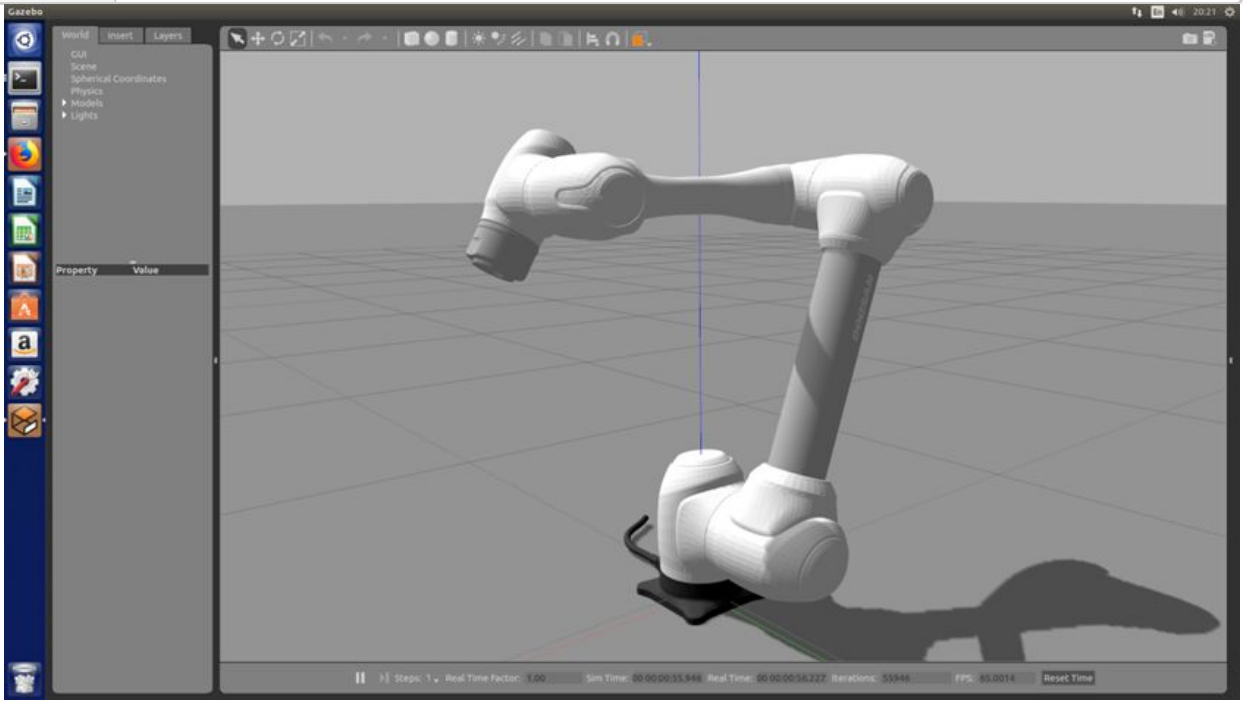


Figure 6.2 single robot

7.2 Multi Robot

7.2.1 Feature

- Provides an example of driving Multi Robot.
  (Please refer to Chapter 5, "dsr_launcher" for detailed robot environment configuration.)
- The example files was written in python- Directory of .py files: ~/catkin_ws/src/doosan-robot/dsr_example/py/scripts/simple

7.2.2 Paramaters of dsr_launcher

Parameter Name	Data type	Default Value	Description
ns	-	dsr01	ROBOT name space . single robot : dsr01 . multi robot: dsr01, dsr02, dsr03, dsr04 …

Parameter Name	Data type	Default Value	Description
host	-	127.0.0.1	Robot controller IP . Emulator : 127.0.0.1 . Real robot controller : 192.168.127.100
port	-	12345	port
mode	-	virtual	Robot operation mode - virtual : virtual mode - real : real mode
model	-	m1013	M-Series Robot model . m0609, m0617, m1013, m1509 A-Series Robot model . a0509
color	-	white	Robot color . white or blue
gripper	-	none	using gripper or not . none : not use gripper . robotiq_2f : use robotiq 2finger gripper
mobile	-	none	using mobile robot or not . none : not use mobile robot . husky : use husky mobile robot

7.2.3 Example

```
1. Robot controller default IP/Port
- IP : 192.168.127.100 , port = 12345
- For multi robot, set the IP of each robot controller differently

1.  launch
- edit launch file
 . $ cd ~/catkin_ws/src/doosan-robot/dsr_launcher/launch
. Modify the multi_robot_*.launch file for each situation.
.. edit argument ns host port, model…
- multi robot in rviz
```

```
$ roslaunch dsr_launcher multi_robot_rviz.launch model:=m1013
- multi robot in gazebo
$ roslaunch dsr_launcher multi_robot_gazebo.launch color:=bule
- multi robot in rviz + gazebo
$ roslaunch dsr_launcher multi_robot_rviz_gazebo.launch

2.  run application node
- Edit example files
   . Open the example file you want to run and modify the robot_id and
robot_model accordingly.
   .. ex>
robot_id1 = "dsr01"; robot_model1 = "m1013"
robot_id2 = "dsr02"; robot_model2 = "m1013"

$ rosrun dsr_example_py multi_robot_simple.py
```

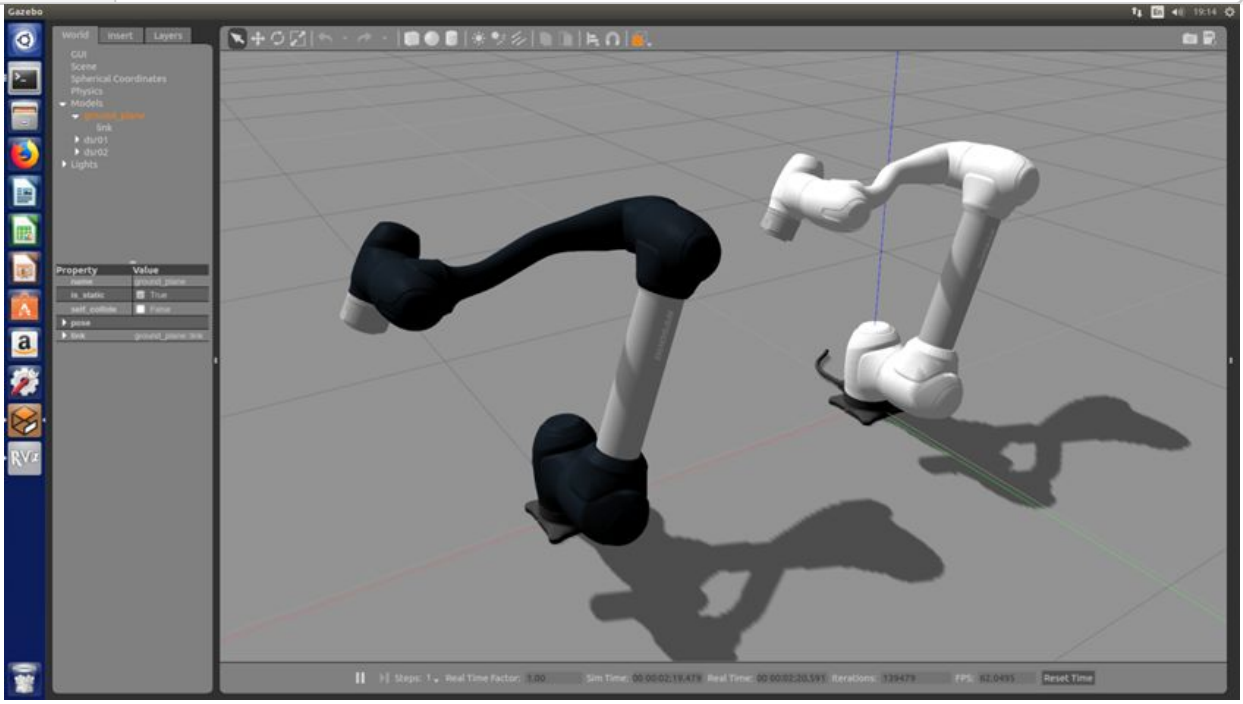


Figure 6.3 multi robot

7.3 Gripper

7.3.1 Feature

- Provides an example of using gripper (robotiq_2f)
  (Please refer to Chapter 5, "dsr_launcher" for detailed robot environment configuration.)
- When running dsr_launcher, give argument (**gripper: = robotiq_2f)**.
- The example files was written in python
  - Directory of .py files: ~/catkin_ws/src/doosan-robot/dsr_example/py/scripts/gripper

7.3.2 Praramaters of dsr_launcher

Parameter Name	Data type	Default Value	Description
ns	-	dsr01	ROBOT name space . single robot : dsr01 . multi robot: dsr01, dsr02, dsr03, dsr04 …
host	-	127.0.0.1	Robot controller IP . Emulator : 127.0.0.1 . Real robot controller : 192.168.127.100
port	-	12345	port
mode	-	virtual	Robot operation mode - virtual : virtual mode - real : real mode
model	-	m1013	M-Series Robot model . m0609, m0617, m1013, m1509 A-Series Robot model . a0509
color	-	white	Robot color . white or blue
gripper	-	none	using gripper or not . none : not use gripper . robotiq_2f : use robotiq 2finger gripper
mobile	-	none	using mobile robot or not . none : not use mobile robot . husky : use husky mobile robot

7.3.3 Example

```
1. Robot controller default IP/Port
```

```
- IP : 192.168.127.100 , port = 12345

2. launch : single robot + gripper
- single robot in rviz
$ roslaunch dsr_launcher single_robot_rviz.launch model:=m1013
gripper:=robotiq_2f
- single robot in gazebo
$ roslaunch dsr_launcher single_robot_gazebo.launch model:=m1013
gripper:=robotiq_2f
- single robot in rviz + gazebo
$ roslaunch dsr_launcher single_robot_rviz_gazebo.launch model:=m1013
gripper:=robotiq_2f

3. run application node
- Edit example files
   . Open the example file you want to run and modify the ROBOT_ID and
ROBOT_MODEL accordingly.
   .. ex>
ROBOT_ID = "dsr01"
ROBOT_MODEL = "m1013"

$ rosrun dsr_example_py pick_and_place.py
```

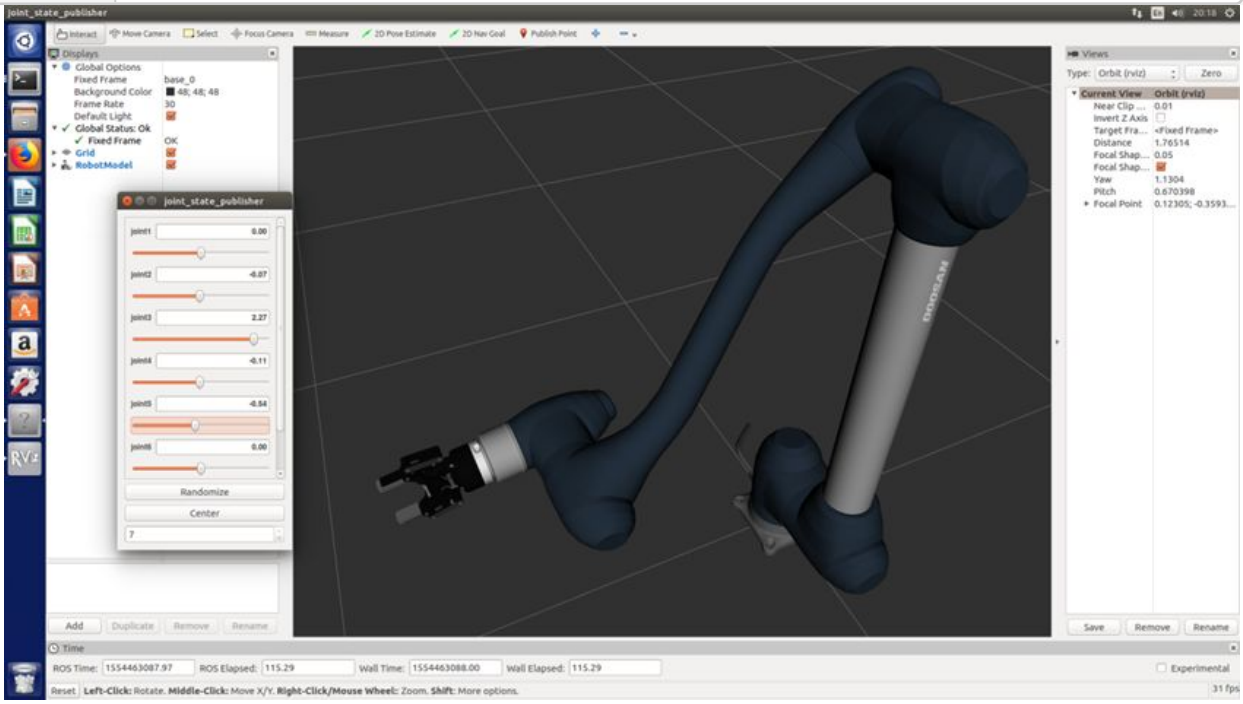


Figure 6.4 robot + gripper

7.4 Mobile robot

7.4.1 Feature

- Provides mobile robot(huskey) examples.
  (Please refer to Chapter 5, "dsr_launcher" for detailed robot environment configuration.)
- When running dsr_launcher, give argument (**mobile:=husky**).
- The example files was written in python
  - Directory of .py files: ~/catkin_ws/src/doosan-robot/dsr_example/py/scripts/mobile

7.4.2 Paramaters of dsr_launcher

Parameter Name	Data type	Default Value	Description
ns	-	dsr01	ROBOT name space . single robot : dsr01 . multi robot: dsr01, dsr02, dsr03, dsr04 …
host	-	127.0.0.1	Robot controller IP . Emulator : 127.0.0.1 . Real robot controller : 192.168.127.100
port	-	12345	port
mode	-	virtual	Robot operation mode - virtual : virtual mode - real : real mode
model	-	m1013	M-Series Robot model . m0609, m0617, m1013, m1509 A-Series Robot model . a0509
color	-	white	Robot color . white or blue

Parameter Name	Data type	Default Value	Description
gripper	-	none	using gripper or not . none : not use gripper . robotiq_2f : use robotiq 2finger gripper
mobile	-	none	using mobile robot or not . none : not use mobile robot . husky : use husky mobile robot

7.4.3 Example

```
1. Robot controller default IP/Port
- IP : 192.168.127.100 , port = 12345

2. launch : single robot + mobile
- single robot in rviz
$ roslaunch dsr_launcher single_robot_rviz.launch model:=m1013
mobile:=husky
- single robot in gazebo
$ roslaunch dsr_launcher single_robot_gazebo.launch model:=m1013
mobile:=husky
- single robot in rviz + gazebo
$ roslaunch dsr_launcher single_robot_rviz_gazebo.launch model:=m1013
mobile:=husky

3. run application node
- Edit example files
   . Open the example file you want to run and modify the ROBOT_ID and
ROBOT_MODEL accordingly.
   .. ex>
ROBOT_ID = "dsr01"
ROBOT_MODEL = "m1013"


$ rosrun dsr_example_py single_robot_moblie.py
```

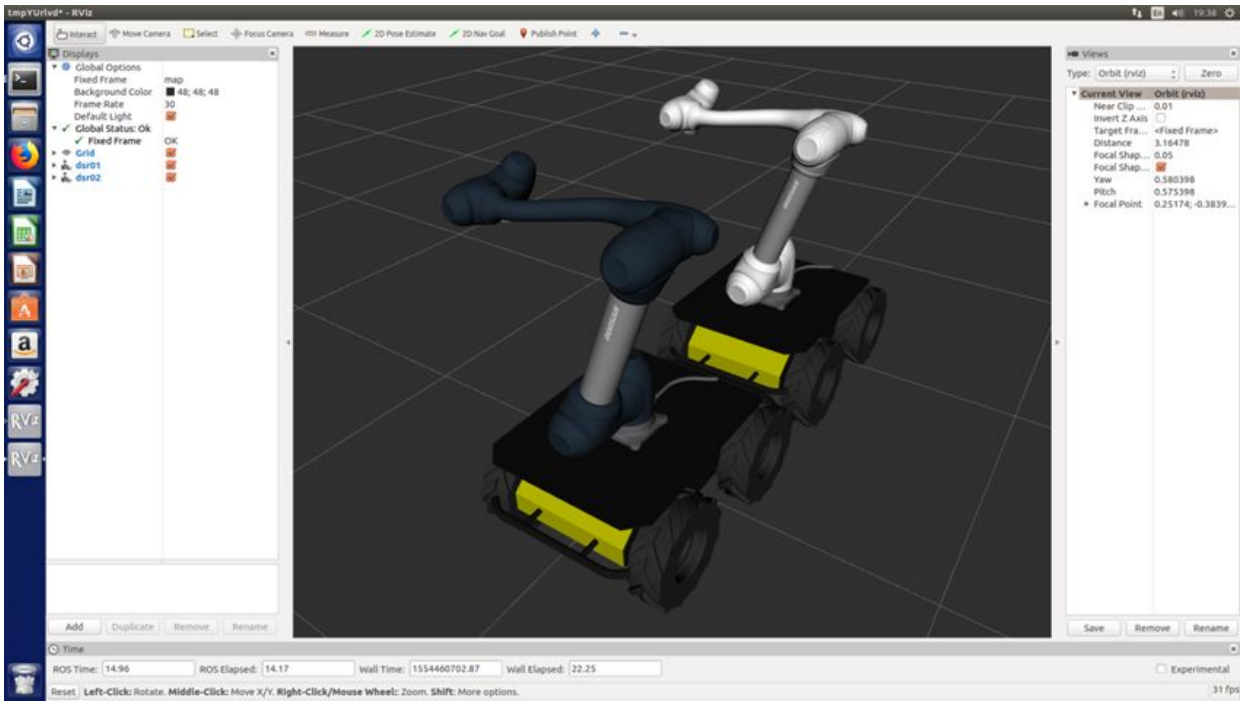


Figure 6.5 robot on mobile

8 dsr_msgs

8.1 Topic

8.1.1 RobotState.msg

Features

Topic message of robot state.

Parameters

Parameter Name	Data Type	Default Value	Description
robot_state	int32		Robot State : enum.ROBOT_STATE
robot_state_str	string		Output robot status as string
actual_mode	int8		Robot Control Mode: enum.CONTROL_MODE
actual_space	int8		Robot Control Space: enum.ROBOT_SPACE
current_posj	float64[6]		6 current joint angle
joint_abs	float64[6]		6 current joint angle(Absolute Encorder)
current_velj	float64[6]		6 current joint velocity
joint_err	float64[6]		6 current joint error
target_posj	float64[6]		6 target joint angle
target_velj	float64[6]		6 target joint velocity
current_posx	float64[6]		6 current task position
current_tool_posx	float64[6]		6 current flange position
current_velx	float64[6]		6 current task velocity

Parameter Name	Data Type	Default Value	Description
task_err	float64[6]		6 current task error
target_posx	float64[6]		6 target task position
target_velx	float64[6]		6 target task velocity
solution_space	int8		solution space(0~7)
rotation_matrix	std_msgs/Float64MultiArray[]		Rotation matrix: float[3][3]
dynamic_tor	float64[6]		6 dynamic torque
actual_jts	float64[6]		6 joint torque sensor data
actual_ejt	float64[6]		6 current joint axis external force
actual_ett	float64[6]		6 current Task-based external forces
actual_w2b	float64[6]		Position deviation between 6 World and Base coordinate systems **This argument is only available in M2.50 and higher versions.**
current_posw	std_msgs/Float64MultiArray[]		6 current task position based on world coordinate system **This argument is only available in M2.50 and higher versions.**
current_velw	float64[6]		6 current task velocity based on world coordinate system **This argument is only available in M2.50 and higher versions.**
world_ett	float64[6]		6 external force based on world coordinate system **This argument is only available in M2.50 and higher versions.**

Parameter Name	Data Type	Default Value	Description
target_posw	float64[6]		6 target task position based on world coordinate system **This argument is only available in M2.50 and higher versions.**
target_velw	float64[6]		6 target task velocity based on world coordinate system **This argument is only available in M2.50 and higher versions.**
rotation_matrix_world	std_msgs/Float64MultiArray[]		Rotation matrix based on world coordinate system: float[3][3] **This argument is only available in M2.50 and higher versions.**
actual_UCN	int8		User coordinate ID information(101 ~ 200) **This argument is only available in M2.50 and higher versions.**
parent	int8		Parent coordinate of current user coordinate system **This argument is only available in M2.50 and higher versions.**
current_posu	float64[6]		6 current task position based on user coordinate system **This argument is only available in M2.50 and higher versions.**
current_velu	float64[6]		6 current task velocity based on user coordinate system **This argument is only available in M2.50 and higher versions.**
user_ett	float64[6]		6 external force based on user coordinate system **This argument is only available in M2.50 and higher versions.**
target_posu	float64[6]		6 target task position based on user coordinate system **This argument is only available in M2.50 and higher versions.**

Parameter Name	Data Type	Default Value	Description
target_velu	float64[6]		6 target task velocity based on user coordinate system **This argument is only available in M2.50 and higher versions.**
rotation_matrix_user	std_msgs/Float64MultiArray[]		Rotation matrix based on user coordinate system: float[3][3] **This argument is only available in M2.50 and higher versions.**
sync_time	int8		Internal clock count
flange_digital_output	int8[6]		6 flange digital output
flange_digital_input	int8[6]		6 Flange Digital Input
actual_bk	int8[6]		6 break state
actual_bt	int8[5]		5 robot button information
actual_mc	float64[6]		Current consumption of 6 motors
actual_mt	float64[6]		6 inverter temperature information
ctrlbox_digital_input	int8[6]		16 control boxes digital intput information
actual_ai	int8[2]		2 Analog Input numeric information
actual_sw	int8[3]		3 switch state
actual_si	int8[2]		2 safety input state
actual_at	int8[2]		2 analog input type information **This argument is only available in M2.50 and higher versions.**
ctrlbox_digital_output	int8[16]		16 control box digital output information
target_ao	float64[2]		2 Analog Output numeric information

Parameter Name	Data Type	Default Value	Description
target_at	int8[2]		2 analog output type information **This argument is only available in M2.50 and higher versions.**
actual_es	int8[2]		2 Encorder state information **This argument is only available in M2.50 and higher versions.**
actual_ed	int8[2]		2 Encorder Raw data numerical information
actual_er	int8[2]		2 Encorder Reset status information **This argument is only available in M2.50 and higher versions.**
modbus_state	ModbusState[]		Modbus State : See ModbusState.msg
access_control	int32		Control state: See enum.ACCESS_CONTROL
homming_completed	bool		Whether the robot is homing
tp_initialized	bool		TP initialization state
mastering_need	int8		Whether robot mastering is required
drl_stopped	bool		DRL (Doosan Robot Language) standstill
disconnected	bool		communication connection status

enum.ROBOT_STATE

Num	Name	Description
0	STATE_INITIALIZING	It is an initialization state for setting various parameters by automatically entering by TP application.
1	STATE_STANDBY	Operable Standby state.

Num	Name	Description
2	STATE_MOVING	It is a command operation state that is automatically switched when the command is received after receiving the command in the command wait state. When the operation is completed, it is switched to the auto command waiting state.
3	STATE_SAFE_OFF	Robot stop mode due to function and operation error, Servo off state (motor and brake power are cut off after control stop)
4	STATE_TEACHING	Direct teaching state
5	STATE_SAFE_STOP	Robot stop mode due to function and operation error. Safe stop status (status in which only the control is stopped, program suspended status in the automatic mode)
6	STATE_EMERGENCY_STOP	Emergency stop state
7	STATE_HOMMING	Homing mode state (state in which robot is aligned in hardware).
8	STATE_RECOVERY	When the robot is stopped due to an error such as exceeding the robot driving range, it is in the recovery mode state to move within the driving range.
9	eSTATE_SAFE_STOP2	Same as eSTATE_SAFE_STOP but a state that must be switched to recovery mode because the robot is out of range
10	STATE_SAFE_OFF2	Same as eSTATE_SAFE_OFF state, but must be in recovery mode due to exceeding the robot drive range
11	STATE_RESERVED1	reserved
12	STATE_RESERVED2	reserved

enum.CONTROL_MODE

Num	Name	Description
0	CONTROL_MODE_POSITION	Position control mode
1	CONTROL_MODE_TORQUE	Torque control mode

enum.ROBOT_STATE

Num	Name	Description
0	ROBOT_SPACE_JOINT	Joint space
1	ROBOT_SPACE_TASK	Task Space

enum.ACCESS_CONTROL

Num	Name	Description
0	MANAGE_ACCESS_CONTROL_FORCE_REQUEEST	Control forced release message transmission
1	MANAGE_ACCESS_CONTROL_REQUEST	Send control request message
2	MANAGE_ACCESS_CONTROL_RESPONSE_YES	Send control authorization request acknowledge message
3	MANAGE_ACCESS_CONTROL_RESPONSE_NO	Send control permission request decline message

8.1.2 RobotStop.msg

Features

Topic message of robot stop.

Parameters

Parameter Name	Data Type	Default Value	Description
stop_mode	int32		robot stop mode : Refer to enum.STOP _MODE.

enum.STOP_MODE

Num	Name	Description
0	DR_QSTOP_STO	reserved
1	DR_QSTOP	quick stop (motion trajectory maintenance)
2	DR_SSTOP	slow stop (motion trajectory maintenance)
3	DR_HOLD	Emergency stop

8.1.3 RobotError.msg

Features

Topic message of robot error.

Parameters

Parameter Name	Data Type	Default Value	Description
Level	int32	-	Log level : enum.LOG _LEVEL
Group	int32	-	Log group : enum.LOG _GROUP
Code	int32	-	error code
msg1	string	-	error msg 1
msg2	string	-	error msg 2
msg3	string	-	error msg 3

enum.LOG_LEVEL

Num	Name	Description
0	LOG_LEVEL_RESERVED	reserved

Num	Name	Description
1	LOG_LEVEL_SYSINFO	Informational messages about simple functions and operational errors
2	LOG_LEVEL_SYSWARN	Robot is stopped due to simple function and operation error.
3	LOG_LEVEL_SYSERROR	Robot is stopped due to safety issue or device error.

enum.LOG_GROUP

Num	Name	Description
0	LOG_GROUP_RESERVED	reserved
1	LOG_GROUP_SYSTEMFMK	Robot Controller (framework)
2	eLOG_GROUP_MOTIONLIB,	Robot Controller (motion)
3	LOG_GROUP_SMARTTP	TP application (GUI)
4	LOG_GROUP_INVERTER	Inverter board
5	LOG_GROUP_SAFETYCONTROLLER	Safety board (Safety Controller)

8.1.4 LogAlarm.msg

Features

Topic message of log alarm

Parameters

Parameter Name	Data Type	Default Value	Description
level	int32	-	Log level : enum.LOG _LEVEL
group	int32	-	Log group : enum.LOG _GROUP
index	int32	-	error code

Parameter Name	Data Type	Default Value	Description
param	string	-	messages[3]

enum.LOG_LEVEL

Num	Name	Description
0	LOG_LEVEL_RESERVED	reserved
1	LOG_LEVEL_SYSINFO	Informational messages about simple functions and operational errors
2	LOG_LEVEL_SYSWARN	Robot is stopped due to simple function and operation error.
3	LOG_LEVEL_SYSERROR	Robot is stopped due to safety issue or device error.

enum.LOG_GROUP

Num	Name	Description
0	LOG_GROUP_RESERVED	reserved
1	LOG_GROUP_SYSTEMFMK	Robot Controller (framework)
2	eLOG_GROUP_MOTIONLIB,	Robot Controller (motion)
3	LOG_GROUP_SMARTTP	TP application (GUI)
4	LOG_GROUP_INVERTER	Inverter board
5	LOG_GROUP_SAFETYCONTROLLER	Safety board (Safety Controller)

8.1.5 ModbusState.msg

Features

Topic message of Modbus State

Parameters

Parameter Name	Data Type	Default Value	Description
level	string	-	Modbus Signal Name.
group	int32	-	Modbus Register Value (Unsigned : 0 ~ 65535)

8.1.6 JogMultiAxis.msg

Features

Topic message for multi-axis jog control

Multi-axis jog speed = (250mm/s)/ x [Unit vector] x speed[%]

> **Caution**
>
> This message is available with robot controller software version 2.50 or later.

Parameters

Parameter Name	Data Type	Default Value	Description
jog_axis	float64[6]	-	Unit vector orientation of task space [Tx, Ty, Tz, Rx, Ry, Rz] (-1.0 ~ 1.0)
move_reference	int8	-	0: MOVE_REFERENCE_BASE 1: MOVE_REFERENCE_TOOL: 2: MOVE_REFERENCE_WORLD
speed	float64		jog speed [%] (1~100)

8.2 Service/motion

8.2.1 Trans.srv

Features

- Input parameter(pos) based on the ref coordinate is translated/rotated as delta based on the same coordinate and this function returns the result that is converted to the value based on the ref_out coordinate.
- In case that the ref coordinate is the tool coordinate, this function retuns the value based on input parameter(pos)'s coordinate without ref_out coordinate.

Parameters

Parameter Name	Data Type	Default Value	Description
pos	float64[6]	-	position list
delta	float64[6]	-	position list
ref	int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120
ref_out	int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120

Note

- The ref argument DR_WORLD is only available in M2.40 or later versions.
- The ref_out argument is only available in M2.40 or later versions.

Return

Parameter Name	Data Type	Default Value	Description
trans_pos	float64[6]	-	Task space point

Parameter Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.2 Fkin.srv

Features

This service receives the input data of joint angles or equivalent forms (float[6]) in the joint space and returns the TCP (objects in the task space) based on the ref coordinate.

Parameter

Parameter Name	Data Type	Default Value	Description
pos	float64[6]	-	Joint Space position list
ref	int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2

> **Note**
>
> The ref argument is only available in M2.40 or later versions.

Return

Parameter Name	Data Type	Default Value	Description
conv_posx	float64[6]	-	Task space point
success	bool	-	True or False

8.2.3 Ikin.srv

Features

This service returns the joint position corresponding to sol_space, which is equivalent to the robot pose in the operating space, among 8 joint shapes.

Parameters

Parameter Name	Data Type	Default Value	Description
pos	float64[6]	-	position list
sol_space	int	-	solution space
ref	int	0	ž MOVE_REFERENCE_BASE =0 ž MOVE_REFERENCE_WORLD=2

Note

The ref argument is only available in M2.40 or later versions.

Robot configuration vs. solution space

Solution space	Binary	Shoulder	Elbow	Wrist
0	000	Lefty	Below	No Flip
1	001	Lefty	Below	Flip
2	010	Lefty	Above	No Flip
3	011	Lefty	Above	Flip
4	100	Righty	Below	No Flip
5	101	Righty	Below	Flip
6	110	Righty	Above	No Flip

Return

Parameter Name	Data Type	Default Value	Description
conv_posj	float64[6]	-	joint angle list

Parameter Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.4 SetRefCoord.srv

Features

This service sets the reference coordinate system.

Parameters

Parameter Name	Data Type	Default Value	Description
coord	int	-	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120

> **Note**
>
> The MOVE_REFERENCE_WORLDargument of ref is only available in M2.40 or later versions

Return

Parameter Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.5 MoveJoint.srv

Features

The robot moves to the target joint position (pos) from the current joint position.

Parameters

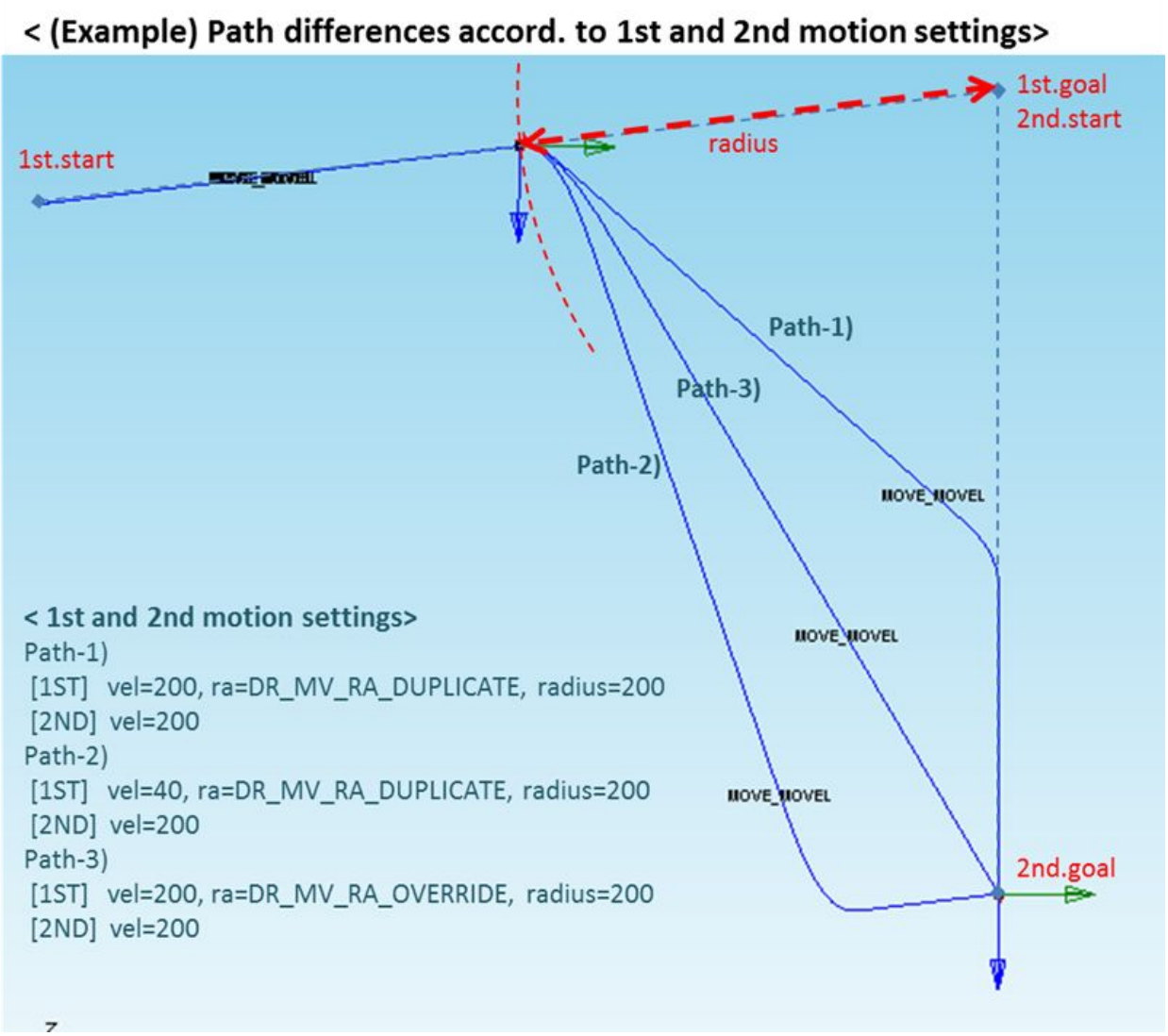
Parameter Name	Data Type	Default Value	Description
pos	float64[6]	-	joint angle list
vel	float64	-	velocity
acc	float64	-	acceleration
time	float64	0.0	Reach time [sec]
radius	float64	0.0	Radius for blending
mode	int8	0	MOVE_MODE_ABSOLUTE =0 MOVE_MODE_RELATIVE =1
blendType	int8	0	BLENDING_SPEED_TYPE_DUPLICATE =0 BLENDING_SPEED_TYPE_OVERRIDE =1
syncType	int8	0	SYNC = 0 ASYNC = 1

Note

If the time is specified, values are processed based on time, ignoring vel and acc.

Caution

If the following motion is blended with the conditions of blendType =BLENDING_SPEED_TYPE_BUPLICATE and radius>0, the preceding motion can be terminated when the following motion is terminated while the remaining motion time determined by the remaining distance, velocity, and acceleration of the preceding motion is greater than the motion time of the following motion. Refer to the following image for more information.



Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.6 MoveLine.srv

Features

The robot moves along the straight line to the target position (pos) within the task space.

Parameters

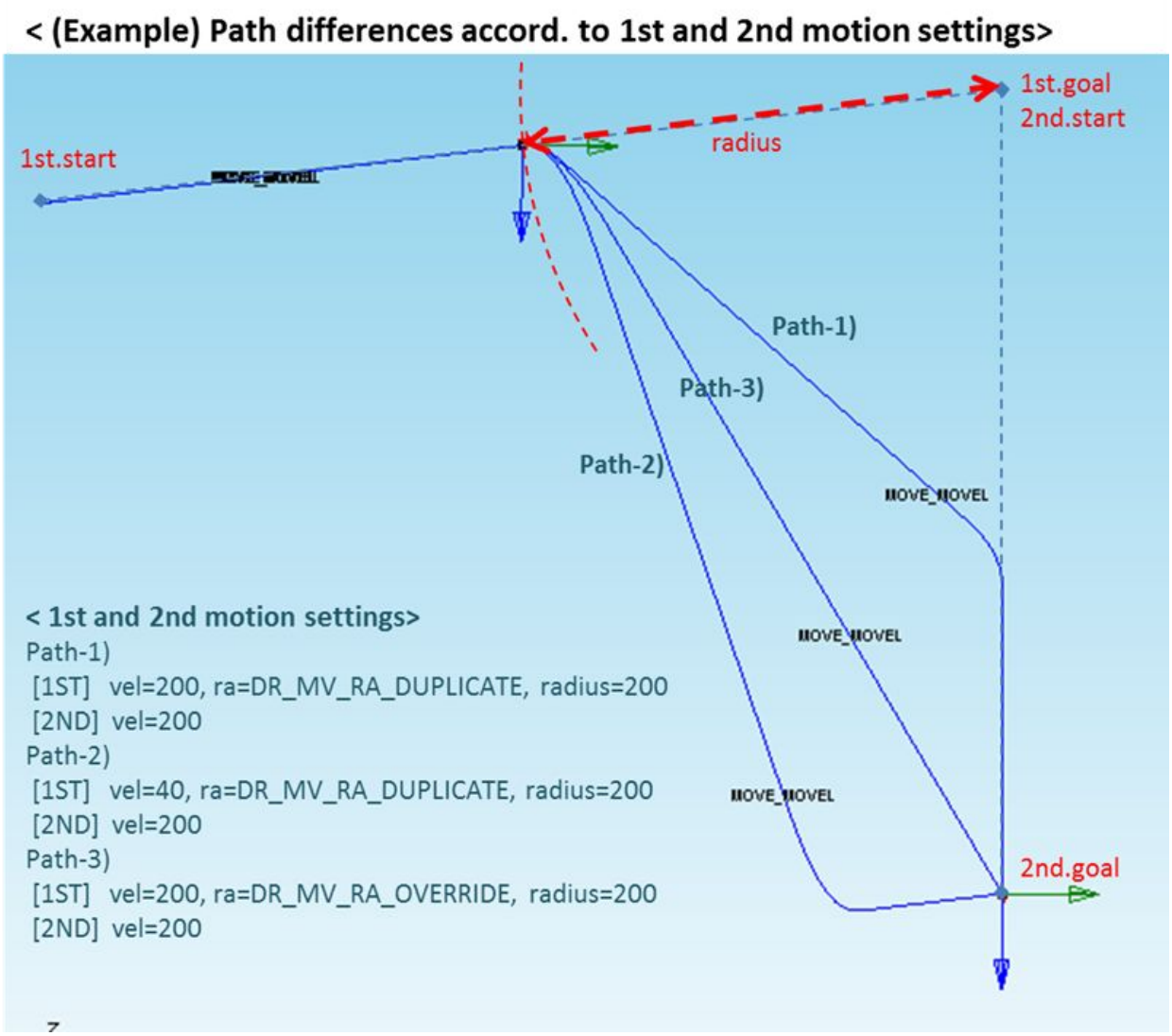
Parameter Name	Data Type	Default Value	Description
pos	float64[6]	-	position list
vel	float64[2]	-	linear velocity, angular velocity
acc	float64[2]	-	linear acceleration, angular acceleration
time	float64	0.0	Reach time [sec] * If the time is specified, values are processed based on time, ignoring vel and acc.
radius	float64	0.0	Radius for blending
ref	int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1
mode	int8	0	MOVE_MODE_ABSOLUTE =0 MOVE_MODE_RELATIVE =1 MOVE_REFERENCE_WORLD=2
blendType	int8	0	BLENDING_SPEED_TYPE_DUPLICATE =0 BLENDING_SPEED_TYPE_OVERRIDE =1
syncType	int8	0	SYNC = 0 ASYNC = 1

Note

- If an argument is inputted to vel (e.g., vel=30), the input argument corresponds to the linear velocity of the motion while the angular velocity is determined proportionally to the linear velocity.
- If an argument is inputted to acc (e.g., acc=60), the input argument corresponds to the linear acceleration of the motion while the angular acceleration is determined proportionally to the linear acceleration.
- If the time is specified, values are processed based on time, ignoring vel and acc
- **The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions.**

Caution

If the following motion is blended with the conditions of blendType = BLENDING_SPEED_TYPE_DUPLICATEE and radius>0, the preceding motion can be terminated when the following motion is terminated while the remaining motion time determined by the remaining distance, velocity, and acceleration of the preceding motion is greater than the motion time of the following motion. Refer to the following image for more information.



Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.7 MoveJointx.srv

Features

The robot moves to the target position (pos) within the joint space.

Since the target position is inputted as a posx form in the task space, it moves in the same way as movel. However, since this robot motion is performed in the joint space, it does not guarantee a linear path to the target position. In addition, one of 8 types of joint combination (robot configurations) corresponding to the task space coordinate system (posx) must be specified in sol (solution space).

Parameters

Parameter Name	Data Type	Default Value	Description
pos	float64[6]	-	position list
vel	float64	-	velocity
acc	float64	-	acceleration
time	float64	0.0	Reach time [sec]
radius	float64	0.0	Radius for blending
ref	int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2
mode	int8	0	MOVE_MODE_ABSOLUTE =0 MOVE_MODE_RELATIVE =1
blendType	int8	0	BLENDING_SPEED_TYPE_DUPLICATE =0 BLENDING_SPEED_TYPE_OVERRIDE =1
sol	int8	0	Solution space
syncType	int8	0	SYNC = 0 ASYNC = 1

Note

- If the time is specified, values are processed based on time, ignoring vel and acc.
- Using the blending in the preceding motion generates an error in the case of input with relative motion (eMoveMode = MOVE_MODE_ RELATIVE), and it is recommended to blend using MoveJoint or MoveLine
- **The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions.**
- Refer to the description of MoveJoint.srv and MoveLine.srv for blending according to option ra and vel/acc.

Robot configuration (Shape vs. solution space)

Solution space	Binary	Shoulder	Elbow	Wrist
0	000	Lefty	Below	No Flip
1	001	Lefty	Below	Flip
2	010	Lefty	Above	No Flip
3	011	Lefty	Above	Flip
4	100	Righty	Below	No Flip
5	101	Righty	Below	Flip
6	110	Righty	Above	No Flip
7	111	Righty	Above	Flip

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.8 MoveCircle.srv

Features

The robot moves along an arc to the target pos (pos2) via a waypoint (pos1) or to a specified angle from the current position in the task space.

Parameters

Parameter Name	Data Type	Default Value	Default Value
pos	std_msgs/Float64MultiArray[]	-	target[2][6] • position list
vel	float64[2]	-	linear velocity, angular velocity
acc	float64[2]	-	linear acceleration, angular acceleration
time	float64	0.0	Reach time [sec]
radius	float64	0.0	Radius for blending
ref	int8	0	reference coordinate • MOVE_REFERENCE_BASE =0 • MOVE_REFERENCE_TOOL=1 • MOVE_REFERENCE_WORLD=2
mode	int8	0	Movement basis • MOVE_MODE_ABSOLUTE =0 • MOVE_MODE_RELATIVE =1
angle1	float64	0.0	angle1
angle2	float64	0.0	angle2

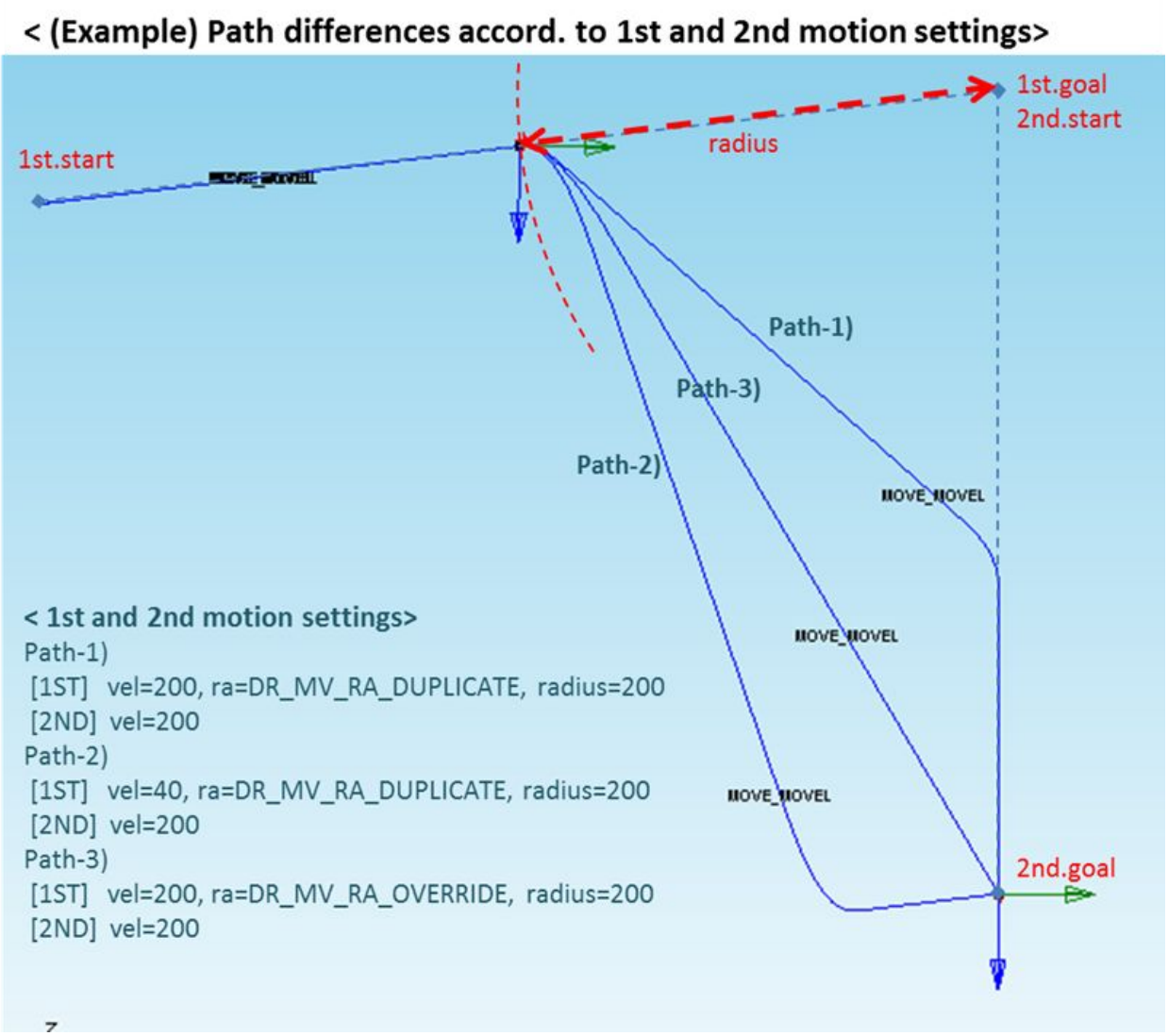
Parameter Name	Data Type	Default Value	Default Value
blendType	int8	0	Reactive motion mode • BLENDING_SPEED_TYPE_DUPLICATE =0 • BLENDING_SPEED_TYPE_OVERRIDE =1
syncType	int8	0	SYNC = 0 ASYNC = 1

Note

- If an argument is inputted to vel (e.g., vel=[30, 0]), the input argument corresponds to the linear velocity of the motion while the angular velocity is determined proportionally to the linear velocity.
- If an argument is inputted to acc (e.g., acc=[60, 0]), the input argument corresponds to the linear acceleration of the motion while the angular acceleration is determined proportionally to the linear acceleration.
- If the time is specified, values are processed based on time, ignoring vel and acc.
- **The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions.**
- If the mod is MOVE_MODE_RELATIVE L, pos[0] and pos[1] are defined in the relative coordinate system of the previous pos. (pos[0] is the relative coordinate from the starting point while pos[1] is the relative coordinate from pos[0].)
- If only one angle is inputted, the total rotated angle on the circular path is applied to the angle.
- If two angle values are inputted, angle1 refers to the total rotating angle moving at a constant velocity on the circular path while angle2 refers to the rotating angle in the rotating section for acceleration and deceleration. In that case, the total moving angle angle1 + 2 X angle2 moves along the circular path.

Caution

If the following motion is blended with the conditions of blendType= BLENDING_SPEED_TYPE_DUPLICATE and radius>0, the preceding motion can be terminated when the following motion is terminated while the remaining motion time determined by the remaining distance, velocity, and acceleration of the preceding motion is greater than the motion time of the following motion. Refer to the following image for more information.



Return

Parameter Name	Data Type	Default Value	Default Value
success	bool	-	True or False

8.2.9 MoveSplineJoint.srv

Features

The robot moves along a spline curve path that connects the current position to the target position (the last waypoint in position list) via the waypoints of the joint space input in position list.

The input velocity/acceleration means the maximum velocity/acceleration in the path, and the acceleration and deceleration during the motion are determined according to the position of the waypoint.

Parameters

Parameter Name	Data Type	Default Value	Description
pos	std_msgs/Float64MultiArray[]	-	target pos [100][6] max = 100
posCnt	int8	-	Count of target pos
vel	float64	-	velocity
acc	float64	-	acceleration
time	float64	0.0	Reach time [sec]
mode	int8	0	Movement basis • MOVE_MODE_ABSOLUTE =0 • MOVE_MODE_RELATIVE =1
syncType	int8	0	SYNC = 0 ASYNC = 1

Note

- If the time is specified, values are processed based on time, ignoring vel and acc.
- If the mod is MOVE_MODE_RELATIVE, each pos in the pos_list is defined in the relative coordinate of the previous pos. (If pos_list=[q1, q2, ...,q(n-1), q(n)], q1 is the relative angle of the starting point while q(n) is the relative coordinate of q(n-1).)
- This service does not support online blending of previous and subsequent motions.

Return

Return Name	Data Type	Default Value	Default Value
success	bool	-	True or False

8.2.10 MoveSplineTask.srv

Features

The robot moves along a spline curve path that connects the current position to the target position (the last waypoint in position list) via the waypoints of the task space input in pos_list.

The input velocity/acceleration means the maximum velocity/acceleration in the path and the constant velocity motion is performed with the input velocity according to the condition if the option for the constant speed motion is selected.

Parameters

Parameter Name	Data Type	Default Value	Description
pos	std_msgs/Float64MultiArray[]	-	target pos [100][6] max = 100
posCnt	int8	-	Count of target pos
vel	float64[2]	-	linear velocity, angular velocity
acc	float64[2]	-	linear acceleration, angular acceleration
time	float64	0.0	Reach time [sec]
ref	int8	0	• reference coordinate MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2
mode	int8	0	• Movement basis MOVE_MODE_ABSOLUTE =0 MOVE_MODE_RELATIVE =1
opt	int8	0	• Velocity option SPLINE_VELOCITY_OPTION_DEFAULT=0 SPLINE_VELOCITY_OPTION_CONST=1

Parameter Name	Data Type	Default Value	Description
syncType	int8	0	SYNC = 0 ASYNC = 1

Note

- If an argument is inputted to vel (e.g., vel=[30, 0]), the input argument corresponds to the linear velocity of the motion while the angular velocity is determined proportionally to the linear velocity.
- If an argument is inputted to acc (e.g., acc=[60, 0]), the input argument corresponds to the linear acceleration of the motion while the angular acceleration is determined proportionally to the linear acceleration.
- If the time is specified, values are processed based on time, ignoring vel and acc.
- **The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions.**
- If the mod is MOVE_MODE_RELATIVE, each pos in the pos_list is defined in the relative coordinate of the previous pos. (If positiolist=[p1, p2, ...,p(n-1), p(n)], p1 is the relative angle of the starting point while p(n) is the relative coordinate of p(n-1).)
- This service does not support online blending of previous and subsequent motions.

Caution

The constant velocity motion according to the distance and velocity between the waypoints cannot be used if the “opt= SPLINE_VELOCITY_OPTION_CONST” option (constant velocity option) is selected, and the motion is automatically switched to the variable velocity motion (opt= SPLINE_VELOCITY_OPTION_DEFAULT) in that case.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.11 MoveBlending.srv

Features

This function takes a list that has one or more path segments (line or circle) as arguments and moves at a constant velocity by blending each segment into the specified radius. Here, the radius can be set through posb.

Parameters

Parameter Name	Data Type	Default Value	Description
pos	std_msgs/Float64MultiArray[]	-	posb list (pos1[6]:pos2[6]:type[1]:radius[1]) x 50(max)
posCnt	int8		Count of target pos
vel	float64[2]	-	linear velocity, angular velocity
acc	float64[2]	-	linear acceleration, angular acceleration
time	float64	0.0	Reach time [sec]
ref	int8	0	• Reference coordinate MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2
mode	int8	0	• Movement basis MOVE_MODE_ABSOLUTE =0 MOVE_MODE_RELATIVE =1
syncType	int8	0	SYNC = 0 ASYNC = 1

Note

- If an argument is inputted to vel (e.g., vel=[30, 0]), the input argument corresponds to the linear velocity of the motion while the angular velocity is determined proportionally to the linear velocity.
- If an argument is inputted to acc (e.g., acc=[60, 0]), the input argument corresponds to the linear acceleration of the motion while the angular acceleration is determined proportionally to the linear acceleration.
- If the time is specified, values are processed based on time, ignoring vel and acc.
- **The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions.**
- If the mod is MOVE_MODE_RELATIVE, each pos in the posb_list is defined in the relative coordinate of the previous pos.

Caution

- A user input error is generated if the blending radius in posb is 0.
- A user input error is generated due to the duplicated input of Line if contiguous Line-Line segments have the same direction.
- A user input error is generated to prevent a sudden acceleration if the blending condition causes a rapid change in direction.
- This service does not support online blending of previous and subsequent motions

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.12 MoveSpiral.srv

Features

The radius increases in a radial direction and the robot moves in parallel with the rotating spiral motion in an axial direction. It moves the robot along the spiral trajectory on the surface that is perpendicular to the axis on the coordinate specified as ref and the linear trajectory in the axis direction.

Parameters

Parameter Name	Data Type	Default Value	Description
revolution	float64	-	Total number of revolutions [revolution]

Parameter Name	Data Type	Default Value	Description
maxRadius	float64		Final spiral radius [mm]
maxLength	float64		Distance moved in the axis direction [mm]
vel	float64[2]	-	linear velocity, angular velocity
acc	float64[2]	-	linear acceleration, angular acceleration
time	float64	0.0	Total execution time [sec]
taskAxis	int8	0	• axis TASK_AXIS_X = 0 TASK_AXIS_Y = 1 TASK_AXIS_Z = 2
ref	int8	0	• reference coordinate MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2
syncType	int8	0	SYNC = 0 ASYNC = 1

Note

- Revolution refers to the maximum radius of the spiral motion.
- Rmax refers to the maximum radius of the spiral motion.
- Lmax refers to the parallel distance in the axis direction during the motion. A negative value means the parallel distance in the –axis direction.
- Vel refers to the moving velocity of the spiral motion
- If the time is specified, values are processed based on time, ignoring vel and acc.
- **The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions.**
- The axis defines the axis that is perpendicular to the surface defined by the spiral motion.
- Ref refers to the reference coordinate system defined by the spiral motion.
- This service does not support online blending of previous and subsequent motions.

Caution

- An error can be generated to ensure safe motion if the rotating acceleration calculated by the spiral path is too great. In this case, reduce the vel, acc, or time value.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.13 MovePeriodic.srv

Features

This function performs the cyclic motion based on the sine function of each axis (parallel and rotation) of the reference coordinate (ref) input as a relative motion that begins at the current position. The attributes of the motion on each axis are determined by the amplitude and period, and the acceleration/deceleration time and the total motion time are set by the interval and repetition count.

Parameters

Parameter Name	Data Type	Default Value	Description
amp	float64[6]	-	Amplitude (motion between -amp and +amp) [mm] or [deg]
periodic	float64[6]	-	Period (time for 1 cycle) [sec]
acc	float64	-	Acceleration
time	float64	-	Acc-, dec- time [sec]
repeat	int8	-	Repetition count
ref	int8	0	• reference coordinate MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2

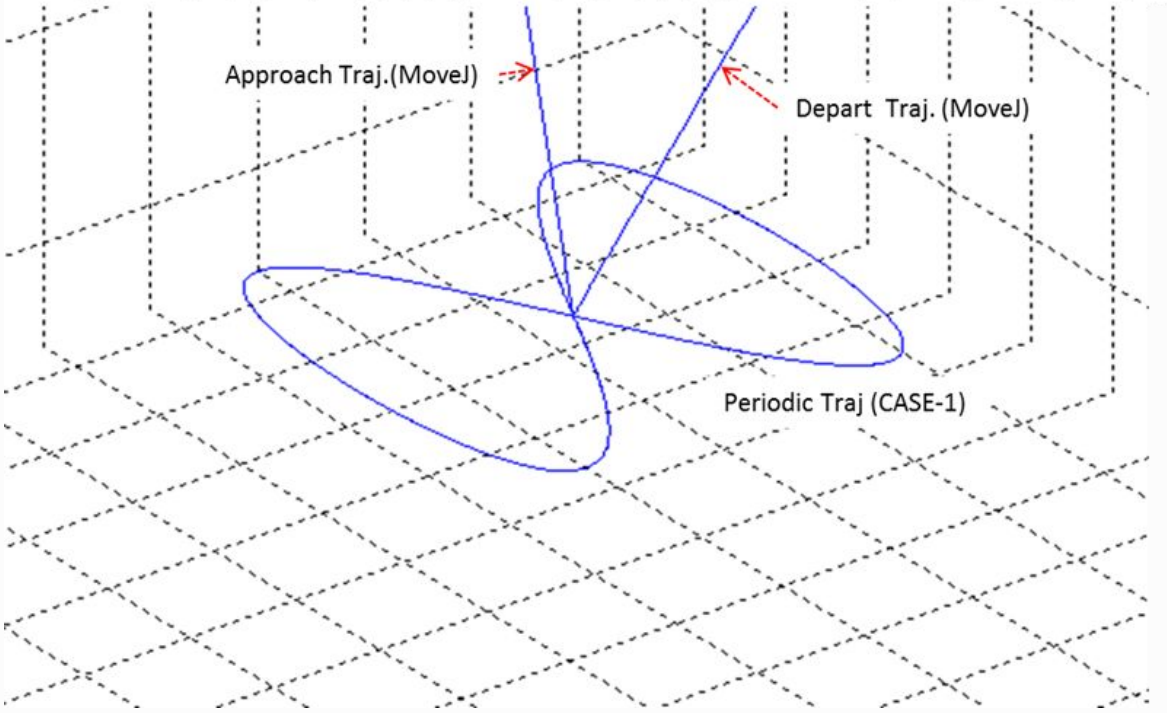
Parameter Name	Data Type	Default Value	Description
syncType	int8	0	SYNC = 0 ASYNC = 1

Note

- Amp refers to the amplitude. The input is a list of 6 elements which are the amp values for the axes (x, y, z, rx, ry, and rz). The amp input on the axis that does not have a motion must be 0.
- Period refers to the time needed to complete a motion in the direction, the amplitude. The input is a list of 6 elements which are the periods for the axes (x, y, z, rx, ry, and rz).
- Atime refers to the acceleration and deceleration time at the beginning and end of the periodic motion. The largest of the inputted acceleration/deceleration times and maximum period*1/4 is applied. An error is generated when the inputted acceleration/deceleration time exceeds 1/2 of the total motion time.
- Repeat refers to the number of repetitions of the axis (reference axis) that has the largest period value and determines the total motion time. The number of repetitions for each of the remaining axes is determined automatically according to the motion time.
- If the motion terminates normally, the motions for the remaining axes can be terminated before the reference axis's motion terminates so that the end position matches the starting position. The deceleration section will deviate from the previous path if the motions of all axes are not terminated at the same time. Refer to the following image for more information.

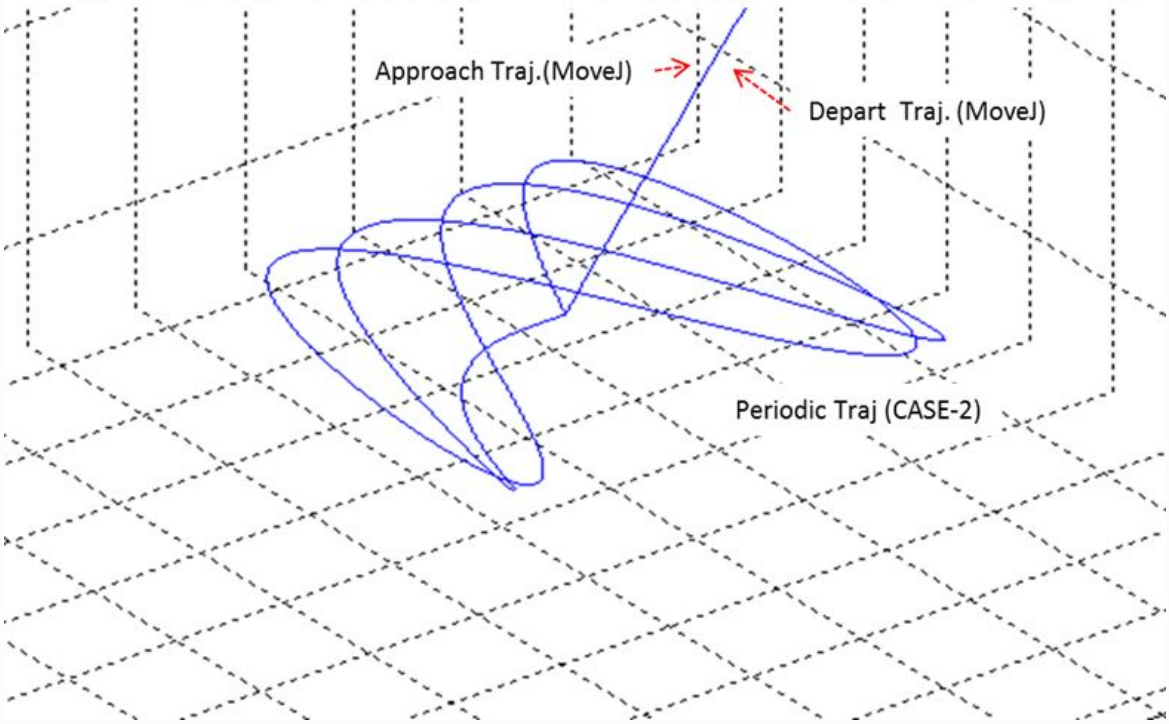
CASE-1) All-axis motions end at the same time

move_periodic(amp=[100,100,0,0,0,0], period=[3.2,1.6,0,0,0,0], atime=3.1, repeat=2, ref=DR_BASE)



CASE-2) Diff-axis motions end individually

move_periodic(amp=[100,100,0,0,0,0], period=[3.2,1.5,0,0,0,0], atime=0, repeat=2, ref=DR_BASE)



- ref refers to the reference coordinate system of the repeated motion.
- **The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions.**

- If a maximum velocity error is generated during a motion, adjust the amplification and period using the following formula.
  velocity = Amplification(amp)*2*pi(3.14)/Period(period) (i.e., Max. velocity=62.83mm/sec if amp=10mm and period=1 sec)
- This function does not support online blending of previous and subsequent motions.

Return

Parameter Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.14 MoveWait.srv

Features

This service sets the waiting time between the previous motion command and the motion command in the next line.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.15 MovePause.srv

Features

It is a service to decelerate and pause the motion of the current robot.

If no robot motion is in progress, it is ignored.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.16 MoveResume.srv

Features

Service to resume the motion of a suspended robot.

If no robot path motion is in progress, it is ignored.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.17 MoveStop.srv

Features

Service to stop robot motion.

It stops differently depending on the input parameters.

Parameters

Parameter Name	Data Type	Default Value	Description
Stop_mode	int32	-	0 : reserved 1 : quick stop (keep motion trajectory) 2 : slow stop (keep motion trajectory)

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.18 Jog.srv

Features

This service is for performing jog motion control for each axis of the robot.

shortened jog speed = (250mm/s) x speed [%]

Parameters

Parameter Name	Data Type	Default Value	Description
Jog_axis	int8	-	0 ~ 5 : JOINT 1 ~ 6 6 ~ 11: TASK 1 ~ 6 (X,Y,Z,rx,ry,rz)
move_reference	int8	-	0 : MOVE_REFERENCE_BASE 1 : MOVE_REFERENCE_TOOL
speed	float64	-	jog speed [%] : + forward , 0=stop, - backward

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.19 JogMulti.srv

Features

This is a service to perform jog control on multiple axes of robots in the robot controller.

Multi-axis jog speed = (250mm / s) / √3 x [unit vector] x speed [%]

This service is available with robot controller software version 2.50 or later.

Parameters

Parameter Name	Data Type	Default Value	Description
jog_axis	float64[6]	-	Unit vector direction of task space [Tx, Ty, Tz, Rx, Ry, Rz] (-1.0 ~ 1.0)
move_reference	int8	-	0: MOVE_REFERENCE_BASE 1: MOVE_REFERENCE_TOOL: 2: MOVE_REFERENCE_WORLD
speed	float64		jog speed [%] (1~100)

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.20 CheckMotion.srv

Features

This service checks the status of the currently active motion..

Return

Return Name	Data Type	Default Value	Description
status	int8	-	0 : No motion in action 1 : nituib being calculated 2 : motion is operation
success	bool	-	True or False

8.2.21 ChangeOperationSpeed.srv

Features

This service adjusts the operation velocity. The argument is the relative velocity in a percentage of the currently set velocity and has a value from 1 to 100. Therefore, a value of 50 means that the velocity is reduced to 50% of the currently set velocity.

Parameters

Parameter Name	Data Type	Default Value	Description
speed	int8	-	operation speed(1~100)

Return

Return	Data Type	Default Value	Description
success	bool	-	True or False

8.2.22 EnableAlterMotion.srv

Features

This service is only available for M2.40 or later versions.

This function sets the configurations for altering function and allows the input quantity of alter_motion() to be applied to motion trajectory. The unit cycle time of generating alter motion is 100msec. Cycle time(n*100msec) can be changed through input parameter n. This function provide 2 modes(Accumulation mode, Increment mode). Input quantity of alter_motion() can be applied to motion trajectory in two ways as accumulated value or increment value. In accumulation mode, the input quantity means absolute altering amount(dX,dY,dZ,dRX,dRY,dRZ) from current motion trajectory. On the contrary in increment mode, the quantity means increment value from the previous absolute altering amount. The reference coordinate can be changed through input parameter ref. Limitations of accumulation amout and increment amount can be set through input paramet limit_dPOS (accumulated limit) and limit_dPOS_per(increment input limit during 1 cycle). The actual alter amount is limited to these limits.

Parameters

Parameter Name	Data Type	Default Value	Description
n	int32	-	Cycle time number
mode	Int8	-	PATH_MODE_DPOS = 0 PATH_MODE_DVEL = 1
ref	int8	-	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120
limit_dPOS	float64[2]	-	First value : limitation of position[mm] Second value : limitation of orientation[deg]
limit_dPOS_per	float64[2]	-	First value : limitation of position[mm] Second value : limitation of orientation[deg]

Note

- alter_motion() can be executed only in user thread.
- When ref is None, _g_coord is applied (_g_coord initial value is DR_BASE, and can be set by the set_ref_coord command)
- Accumulation amount or increment amout isn't be limited if limit_dPOS or limit_dPOS_per is None.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.23 AlterMotion.srv

Features

This service is only available for M2.40 or later versions.

This service applies altering amount of motion trajectory when the alter function is activated. The meaning of the input values is defined from enable_alter_motion().

Caution

alter_motion() can be executed only in user thread.

Note

- alter_motion() can be executed only in user thread.
- Alter motion can be adjusted through setting value limit_dPOS or limit_dPOS_per in enable_alter_motion function.
- Orientation of Input pose follows fixed XYZ notation.

Parameters

Parameter Name	Data Type	Default Value	Description
pos	float64[6]	-	position list

Return

Parameter Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.24 DisableAlterMotion.srv

Features

This service is only available for M2.40 or later versions.

This service deactivates alter motion.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.2.25 SetSingularityHandling.srv

Features

This is a service to allow the user to select a response policy when path deviation occurs due to the influence of a singularity in task motion. The following settings are possible for mode setting.

- Auto Avoidance Mode(Default) : SINGULARITY_AVOIDANCE_AVOID
- Path First : SINGULARITY_AVOIDANCE_STOP
- Velocity Variable : SINGULARITY_AVOIDANCE_VEL

The default setting is **Auto Avoidance Mode**, which reduces instability due to singularities but reduces path tracking accuracy. In the case of **Path First** setting, if there is a possibility that instability may occur due to the influence of singularity, a warning message is output after deceleration and the task is terminated. For **Velocity Variable** setting, it increases path tracking accuracy while reducing instability due to singularities. However, TCP speed change occurs in the singularity section.

Parameters

Parameter Name	Data Type	Default Value	Description
mode	int8	0	SINGULARITY_AVOIDANCE_AVOID = 0 SINGULARITY_AVOIDANCE_STOP = 1 SINGULARITY_AVOIDANCE_VEL = 2

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.3 Service/system

8.3.1 GetRobotMode.srv

Features

It is a service input for checking the current operation mode of the robot controller.

The auto mode is a mode for automatically performing a series of operations (programs), and the manual mode is for performing a single operation such as jogging.

Return

Return Name	Data Type	Default Value	Description
robot_mode	int8	-	refer to enum.ROBOT_MODE.
success	bool	-	True or False

enum.ROBOT_MODE

Num	Name	Description
0	ROBOT_MODE_MANUAL	Manual mode
1	ROBOT_MODE_AUTONOMOUS	Auto mode
2	ROBOT_MODE_MEASURE	Measure mode (Not currently supported)

8.3.2 SetRobotMode.srv

Features

This service is for setting the current operation mode of the robot controller.

Parameters

Parameter Name	Data Type	Default Value	Description
robot_mode	int8	-	refer to enum.ROBOT_MODE

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

enum.ROBOT_MODE

Num	Name	Description
0	ROBOT_MODE_MANUAL	Manual mode
1	ROBOT_MODE_AUTONOMOUS	Auto mode
2	ROBOT_MODE_MEASURE	Measure mode (Not currently supported)

8.3.3 GetRobotSystem

Features

It is a service input for confirming the current operation mode (virtual robot, actual robot) of the robot controller.

Return

Return Name	Data Type	Default Value	Description
robot_system	int8	-	refer to enum.ROBOT_SYSTEM
success	bool	-	True or False

enum.ROBOT_SYSTEM

Num	Name	Description
0	ROBOT_SYSTEM_REAL	Actual robot system
1	ROBOT_SYSTEM_VIRTUAL	virtual robot system

8.3.4 SetRobotSystem.srv

Features

This is a service for setting up the current robot system of the robot controller.

Parameters

Parameter Name	Data Type	Default Value	Description
robot_system	int8	-	refer to enum.ROBOT_SYSTEM

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

enum.ROBOT_SYSTEM

Num	Name	Description
0	ROBOT_SYSTEM_REAL	Actual robot system
1	ROBOT_SYSTEM_VIRTUAL	virtual robot system

8.3.5 GetRobotSpeedMode.srv

Features

This service is used to check the current speed mode (normal mode, deceleration mode) from the robot controller.

Return

Return Name	Data Type	Default Value	Description
Speed_mode	int8	-	refer to enum.SPEED_MODE
success	bool	-	True or False

enum.SPEED_MODE

Num	Name	Description
0	SPEED_NORMAL_MODE	Normal speed mode
1	SPEED_REDUCED_MODE	deceleration speed mode

8.3.6 SetRobotSpeedMode.srv

Features

This service is used to set and change the currently operating speed mode of the robot controller.

Parameters

Parameter Name	Data Type	Default Value	Description
speed_mode	int8	-	refer to enum.SPEED_MODE

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

enum.SPEED_MODE

Num	Name	Description
0	SPEED_NORMAL_MODE	Normal speed mode
1	SPEED_REDUCED_MODE	deceleration speed mode

8.3.7 SetSafeStopResetType.srv

Features

This service is used to define a series of actions to be executed automatically after the state transition using the SetRobotMode service when the operation status information of the robot controller is SAFE_STOP.

If the robot operation mode is automatic, you can define and set whether to re-execute the program. In manual mode, this setting is ignored.

Parameters

Parameter Name	Data Type	Default Value	Description
reset_type	int8	-	refer to enum.SAFE_STOP_RESET_TYPE

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

enum.SAFE_STOP_RESET_TYPE

Num	Name	Description
0	SAFE_STOP_RESET_TYPE_DEFAULT	Simple state release (manual mode)
	SAFE_STOP_RESET_TYPE_PROGRAM_STOP	Stop program (autoc mode)
1	SAFE_STOP_RESET_TYPE_PROGRAM_RESUME	Restart the program (automatic mode)

8.3.8 GetCurrentPose.srv

Features

This service is used to check the current position information of each axis of the robot according to the coordinate system (joint space or task space) in the robot controller.

Parameters

Parameter Name	Data Type	Default Value	Description
space_type	int8	-	refer to enum.ROBOT_SPACE.

Return

Return Name	Data Type	Default Value	Description
pos	float64[6]	-	Robot position information
success	bool	-	True or False

enum.ROBOT_SPACE

Num	Name	Description
0	ROBOT_SPACE_JOINT	Joint space

Num	Name	Description
1	ROBOT_SPACE_TASK	task space

8.3.9 GetLastAlarm.srv

Features

This service is used to check the most recent log and alarm codes generated by the robot controller.

Return

Return Name	Data Type	Default Value	Description
log_alarm	LogAlarm.msg	-	refet to LogAlam.msg
success	bool	-	True or False

LogAlarm.msg

Parameter Name	Data Type	Default Value	Description
level	int32	-	refet to enum.LOG _LEVEL
group	int32	-	refet to enum.LOG _GROUP
index	int32	-	error code
param	string[3]	-	param[3]

enum.LOG_LEVEL

Num	Name	Description
0	LOG_LEVEL_RESERVED	reserved
1	LOG_LEVEL_SYSINFO	Informational messages about basic functions and operational errors

Num	Name	Description
2	LOG_LEVEL_SYSWARN	Robot is stopped due to basic function and operation error
3	LOG_LEVEL_SYSERROR	Robot is stopped due to safety issue or device error

enum.LOG_GROUP

Num	Name	Description
0	LOG_GROUP_RESERVED	reserved
1	LOG_GROUP_SYSTEMFMK	framework
2	eLOG_GROUP_MOTIONLIB,	Motion algorithm
3	LOG_GROUP_SMARTTP	TP program (GUI)
4	LOG_GROUP_INVERTER	Robot Inverter Board
5	LOG_GROUP_SAFETYCONTROLLER	Safety Controller

The log and alarm messages are passed through the predefined contents through the number, and the relevant parameters are sent together if necessary.

Please refer to log and alarm definition section for details.

8.4 Service/aux_control

8.4.1 GetControlMode.srv

Features

This service returns the current control mode.

Return

Parameter Name	Data Type	Default Value	Description
control_mode	int8	-	Control mode 3 : Position Control Mode 4 : Torque Control mode
success	bool	-	True or False

8.4.2 GetControlSpace.srv

Features

This service returns the current control space.

Return

Return Name	Data Type	Default Value	Description
space	int8	-	Control mode 1 : Joint space control 2 : Task space control
success	bool	-	True or False

8.4.3 GetCurrentPosj.srv

Feature

This service returns the current joint angle.

Return

Return Name	Data Type	Data Type	Description
pos	float64[6]	-	Joint angle
success	bool	-	True or False

8.4.4 GetCurrentVelj.srv

Features

This service returns the current joint velocity.

Return

Return Name	Data Type	Default Value	Description
joint_speed	float64[6]	-	Joint Speed
success	bool	-	True or False

8.4.5 GetDesiredPosj.srv

Features

This service returns the current target joint angle. It cannot be used in the movel, movec, movesx, moveb, move_spiral, or move_periodic command.

Return

Return Name	Data Type	Default Value	Description
pos	float64[6]	-	Joint angle
success	bool	-	True or False

8.4.6 GetDesiredVelj.srv

Features

This service returns the current target joint velocity. It cannot be used in the movel, movec, movesx, moveb, move_spiral, or move_periodic command.

Return

Return Name	Data Type	Default Value	Description
joint_vel	float64[6]	-	target joint velocity
success	bool	-	True or False

8.4.7 GetCurrentPosx.srv

Features

This service returns the pose and solution space of the current coordinate system. The pose is based on the ref coordinate.

Parameters

Parameter Name	Data Type	Default Value	Description
ref	Int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120

Note

- MOVE_REFERENCE_BASE is applied when ref is omitted.
- The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions.

Return

Return Name	Data Type	Default Value	Description
task_pos_info	std_msgs/Float64MultiArray[]	-	posx list : task_pos_info[0][0:5] solution space : task_pos_info[0][6]
success	bool	-	True or False

8.4.8 GetCurruntToolFlangePosx.srv

Features

This service returns the pose of the current tool flange based on the ref coordinate. In other words, it means the return to tcp=(0,0,0,0,0,0).

Parameters

Parameter Name	Data Type	Default Value	Description
ref	Int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2

> **Note**
>
> The ref argument is only available in M2.40 or later versions.

Return

Return Name	Data Type	Default Value	Description
pos	float64[6]	-	Pose of tool flange
success	bool	-	True or False

8.4.9 GetCurrentVelx.srv

Features

This service returns the current tool velocity based on the ref coordinate.

Parameters

Parameter Name	Data Type	Default Value	Description
ref	Int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2

> **Note**
>
> The ref argument is only available in M2.40 or later versions.

Return

Return Name	Data Type	Default Value	Description
vel	float64[6]	-	tool velocity
success	bool	-	True or False

8.4.10 GetDesiredPosx.srv

Features

This service returns the target pose of the current tool. The pose is based on the ref coordinate.

Parameters

Parameter Name	Data Type	Default Value	Description
ref	Int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120

> **Note**
>
> The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions

Return

Return Name	Data Type	Default Value	Description
pos	float64[6]	-	target tool position
success	bool	-	True or False

8.4.11 GetDesiredVelx.srv

Features

This service returns the target velocity of the current tool based on the ref coordinate. It cannot be used in the movej, movejx, or movesj command.

Parameters

Parameter Name	Data Type	Default Value	Description
ref	Int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2

> **Note**
>
> The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions.

Return

Return Name	Data Type	Default Value	Description
vel	float64[6]	-	tool velocity
success	bool	-	True or False

8.4.12 GetCurrentSolutionSpace.srv

Features

It is a service to get the the current solution space value.

Return

Return Name	Data Type	Default Value	Description
solution_space	int8	-	solution space (0~7)
success	bool	-	True or False

Robot configuration (shape vs. solution space)

Solution space	Binary	Shoulder	Elbow	Wrist
0	000	Lefty	Below	No Flip
1	001	Lefty	Below	Flip
2	010	Lefty	Above	No Flip
3	011	Lefty	Above	Flip
4	100	Righty	Below	No Flip
5	101	Righty	Below	Flip
6	110	Righty	Above	No Flip
7	111	Righty	Above	Flip

8.4.13 GetCurrentRotm.srv

Features

This serivce returns the direction and matrix of the current tool based on the ref coordinate.

Parameters

Parameter Name	Data Type	Default Value	Description
ref	Int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2

> **Note**
>
> The ref argument is only available in M2.40 or later versions.

Return

Return Name	Return	Default Value	Description
rot_matrix	std_msgs/Float64MultiArray[]	-	Rotation Matrix
success	bool	-	True or False

8.4.14 GetJointTorque.srv

Features

This service returns the sensor torque value of the current joint.

Return

Return Name	Default Value	Default Value	Description
joint_torque	float64[6]	-	JTS torque value
success	bool	-	True or False

8.4.15 GetExternalTorque.srv

Features

This service returns the torque value generated by the external force on each current joint.

Return

Return Name	Data Type	Default Value	Description
ext_torque	float64[6]	-	Torque value generated by and external force
success	bool	-	True or False

8.4.16 GetToolForce.srv

Features

This service returns the external force applied to the current tool based on the ref coordinate.

Parameters

Parameter Name	Data Type	Default Value	Description
ref	Int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2

> **Note**
>
> The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions

Return

Return Name	Data Type	Default Value	Description
tool_force	float64[6]	-	External force applied to the tool
success	bool	-	True or False

8.4.17 GetSolutionSpace.srv

Features

This service obtains the solution space value.

Parameters

Parameter Name	Data Type	Default Value	Description
pos	float64[6]	-	position list

Return

Return Name	Data Type	Default Value	Description
sol_space	int8	-	solution space : 0 ~ 7
success	bool	-	True or False

8.4.18 GetOrientationError.srv

Features

This service returns the orientation error value between the arbitrary poses xd and xc of the axis.

Parameters

Parameter Name	Data Type	Default Value	Description
xd	float64[6]	-	position list
xc	float64[6]	-	Position list
axis	int8	-	axis • TASK_AXIS_X : 0 • TASK_AXIS_Y : 1 • TASK_AXIS_Z : 2

Return

Return Name	Data Type	Default Value	Description
ori_error	float32	-	Orientaion error value
success	bool	-	True or False

8.5 Service/tcp

8.5.1 ConfigCreateTcp.srv

Features

This is a service to register and use robot TCP information in advance for safety. TCP information registered using this service is stored in memory, so it must be set again after rebooting. if it is registered in the T/P application, it can be reused as it is added during the initialization process.

Parameters

Parameter Name	Data Type	Default Value	Description
name	string	-	Name of the TCP
pos	float64[6]	-	TCP infomation

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.5.2 ConfigDeleteTcp.srv

Features

This service calls deleting the registered TCP information.

Parameters

Parameter Name	Data Type	Default Value	Description
name	string	-	Name of the TCP

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.5.3 GetCurrentTcp.srv

Features

It is a service that fetches the currently set TCP information from the robot controller.

Return

Return Name	Data Type	Default Value	Description
info	string	-	Name of the TCP
success	bool	-	True or False

8.5.4 SetCurrentTcp.srv

Features

This is a service to set the currently installed TCP information among the TCP information registered in advance in the robot controller. If there is no currently installed TCP, passing an empty string will initialize the currently set information.

Parameters

Parameter Name	Data Type	Default Value	Description
name	string	-	Name of the TCP

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.6 Service/tool

8.6.1 ConfigCreateTool.srv

Features

This is a service to register and use the tool information to be installed at the end of the robot in advance for safety. Tool information registered using this service is stored in the memory, so it must be set again after rebooting, but if registered in the T/P application, the initialization process It can be reused as it is added from.

Parameters

Parameter Name	Data Type	Default Value	Description
name	string	-	Name of the Tool
weight	float		Weight of tool
cog	float64[3]		center of mass
inertia	float64[6]		inertia information

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.6.2 ConfigDeleteTool.srv

Features

This service calls deleting the registered TOOL information.

Parameters

Parameter Name	Data Type	Default Value	Description
name	string	-	Name of the Tool

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.6.3 GetCurrentTool.srv

Features

It is a service that fetches the currently set TOOL information from the robot controller. If there is no tool information set, an empty string is returned.

Return

Return Name	Data Type	Default Value	Description
info	string	-	Name of the TOOL
success	bool	-	True or False

8.6.4 SetCurrentTool.srv

Features

This is a service to set the information about the currently installed tool among the tool information registered in advance in the robot controller. If there is no currently installed tool, passing an empty string will initialize the currently set information.

Parameters

Parameter Name	Data Type	Default Value	Description
name	string	-	Name of the TOOL

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7 Service/force

8.7.1 ParallelAxis1.srv

Features

This service matches the normal vector of the plane consists of points(x1, x2, x3) based on the ref coordinate(refer to get_normal(x1, x2, x3)) and the designated axis of the tool frame. The current position is maintained as the TCP position of the robot.

Parameters

Parameter Name	Data Type	Default Value	Description
x1	float64[6]	-	position list
x2	float64[6]	-	position list

Parameter Name	Data Type	Default Value	Description
x3	float64[6]	-	position list
axis	int8	-	TASK_AXIS_X = 0 TASK_AXIS_Y = 1 TASK_AXIS_Z = 2
ref	int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120

Note

The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7.2 ParallelAxis2.srv

Features

This service matches the given vect direction based on the ref coordinate and the designated axis of the tool frame. The current position is maintained as the TCP position of the robot.

Parameters

Parameter Name	Data Type	Default Value	Description
vect	float64[3]	-	vector
axis	int8	-	TASK_AXIS_X = 0 TASK_AXIS_Y = 1 TASK_AXIS_Z = 2

Parameter Name	Data Type	Default Value	Description
ref	int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120

> **Note**
>
> The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7.3 AlignAxis1.srv

Features

This service matches the normal vector of the plane consists of points(x1, x2, x3) based on the ref coordinate(refer to get_normal(x1, x2, x3)) and the designated axis of the tool frame. The robot TCP moves to the pos position.

Parameters

Parameter Name	Data Type	Default Value	Description
x1	float64[6]	-	position list
x2	float64[6]	-	position list
x3	float64[6]	-	position list
pos	float64[6]	-	position list
axis	int8	-	TASK_AXIS_X = 0 TASK_AXIS_Y = 1 TASK_AXIS_Z = 2

Parameter Name	Data Type	Default Value	Description
ref	int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120

> **Note**
>
> The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7.4 AlignAxis2.srv

Features

This serivce matches the given vect direction based on the ref coordinate and the designated axis of the tool frame. The robot TCP moves to the pos position.

Parameters

Parameter Name	Data Type	Default Value	Description
vect	float64[3]	-	vector
pos	float64[6]	-	position list
axis	int	-	TASK_AXIS_X = 0 TASK_AXIS_Y = 1 TASK_AXIS_Z = 2
ref	int	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120

> **Note**
>
> The MOVE_REFERENCE_WORLD argument of ref is only available in M2.40 or later versions.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7.5 IsDoneBoltTightening.srv

Features

This service monitors the tightening torque of the tool and returns True if the set torque (m) is reached within the given time and False if the given time has passed.

Parameters

Parameter Name	Data Type	Default Value	Description
m	float64	0	Target torque
timeout	float64	0	Monitoring duration [sec]
axis	int8	-	TASK_AXIS_X = 0 TASK_AXIS_Y = 1 TASK_AXIS_Z = 2

리턴

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7.6 ReleaseComplianceCtrl.srv

Features

This service terminates compliance control and begins position control at the current position.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7.7 TaskComplianceCtrl.srv

Features

This service begins task compliance control based on the preset reference coordinate system.

Parameters(Stiffness TBD)

Parameter Name	Data Type	Default Value	Description
stx	float64[6]	[3000, 3000, 3000, 200, 200, 200]	Three translational stiffnesses Three rotational stiffnesses
ref	int8	1	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120
time	float	0	Stiffness varying time [sec] Range: 0 - 1.0 * Linear transition during the specified time

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7.8 SetStiffnessx.srv

Features

This service sets the stiffness value based on the global coordinate(refer to set_ref_coord()). The linear transition from the current or default stiffness is performed during the time given as STX. The user-defined ranges of the translational stiffness and rotational stiffness are 0-20000N/m and 0-400Nm/rad, respectively.

Parameters

Parameter Name	Data Type	Default Value	Description
stx	float64[6]	[500, 500, 500, 100, 100, 100]	Three translational stiffnesses Three rotational stiffnesses
ref	int8	1	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120
time	float	0	Stiffness varying time [sec] Range: 0 - 1.0 * Linear transition during the specified time

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7.9 CalcCoord.srv

Features

This service is only available for M2.50 or later versions.

This service returns a new user cartesian coordinate system by using up to 4 input poses ([x1]~[x4]), input mode [mod] and the reference coordinate system [ref]. The input mode is only valid when the number of input robot poses is 2.

In the case that the number of input poses is 1, the coordinate system is calculated using the position and orientation of x1.

In the case that the number of input poses is 2 and the input mode is 0, X-axis is defined by the direction from x1 to x2, and Z-axis is defined by the projection of the current Tool-Z direction onto the plane orthogonal to the x-axis. The origin is the position of x1.

In the case that the number of input poses is 2 and the input mode is 1, X-axis is defined by the direction from x1 to x2, and Z-axis is defined by the projection of the z direction of x1 onto the plane orthogonal to the X-axis. The origin is the position of x1.

In the case that the number of input poses is 3, X-axis is defined by the direction from x1 to x2. If a vector v is the direction from x1 to x3, Z-axis is defined by the cross product of X-axis and v (X-axis cross v). The origin is the position of x1.

In the case that the number of input poses is 4, the definition of axes is identical to the case that the number of input poses is 3, but the origin is the position of x4.

Parameters

Parameter Name	Data Type	Default Value	Description
input_pos_cnt	int8		Number of input positions
x1	float64[6]	-	position list
x2	float64[6]	-	position list
x3	float64[6]	-	position list
x4	float64[6]	-	position list
ref	int8	-	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2

Parameter Name	Data Type	Default Value	Description
mod	int8	-	input mode (only valid when the number of input poses is 2) 0: defining z-axis based on the current Tool-z direction 1: defining z-axis based on the z direction of x1

Return

Return Name	Data Type	Default Value	Description
conv_posx	float64[6]	-	position list
success	bool	-	True or False

8.7.10 SetUserCartCoord1.srv

Features

This service set a new user cartesian coordinate system using input pose [pos] and reference coordinate system[ref]. Up to 20 user coordinate systems can be set including the coordinate systems set within Workcell Item. Since the coordinate system set by this function is removed when the program is terminated, setting new coordinate systems within Workcell Item is recommended for maintaining the coordinate information.

Parameters

Parameter Name	Data Type	Default Value	Description
pos	float64[6]	-	coordinate information (position and orientation)
ref	int8	-	reference coordinate DR_BASE: base coordinate DR_WORLD: world coordinate

Note

The ref argument is only available in M2.40 or later versions.

Return

Return Name	Data Type	Default Value	Description
id	int8	-	user coordinate ID(101~120) or fail(-1)
success	bool	-	True or False

8.7.11 SetUserCartCoord2.srv

Features

The user can set the new rectangular coordinate system using x1, x2, and x3 based on the ref coordinate. Creates a rectangular coordinate system with ux, uy, and uz as the vector for each axis and origin point is the pos based on the ref coordinate assuming that 1)ux is the unit vector of x1x2 and uy is the unit vector of the vector that represents the shortest distance between x1x2 and x3. A maximum of 20 can be used, and the most recent 20 values are used if there are more than 20.

1)Before M2.0.2 software version, ux is the unit vector of x2x1

Parameters

Parameter Name	Data Type	Default Value	Description
x1	float64[6]	-	position list
x2	float64[6]	-	position list
x3	float64[6]	-	position list
pos	float64[6]	-	position list
ref	int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2

Note

The ref argument is only available in M2.40 or later versions..

Return

Return Name	Data Type	Default Value	Description
id	int8	-	user coordinate ID(101~120) or fail(-1)
success	bool	-	True or False

8.7.12 SetUserCartCoord3.srv

Features

The user can set the new rectangular coordinate system using u1 and v1 based on the ref coordinate. The origin point of the rectangular coordinate system is pos based on the ref coordinate while the x-axis and y-axis bases are given in the vectors u1 and v1, respectively. Other directions are determined by u1 x v1. If u1 and v1 are not orthogonal, v1', that is perpendicular to u1 on the surface spanned by u1 and v1, is set as the vector in the y-axis direction.

Parameters

Parameter Name	Data Type	Default Value	Description
u1	float64[3]	-	X-axis unit vector
v1	float64[3]	-	Y-axis unit vector
pos	float64[6]	-	position list
ref	int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2

Note

The ref argument is only available in M2.40 or later versions.

Return

Return Name	Data Type	Default Value	Description
id	int8	-	user coordinate ID(101~120) or fail(-1)
success	bool	-	True or False

8.7.13 OverwriteUserCartCoord.srv

Features

This service is only available for M2.50 or later versions.

This service changes the pose and reference coordinate system of the requested user coordinate system [id] with the [pos] and [ref], respectively.

Parameters

Parameter Name	Data Type	Default Value	Description
id	int8	-	coordinate ID
pos	float64[6]	-	position list
ref	int8	0	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_WORLD=2

Return

Return Name	Data Type	Default Value	Description
id	int8	-	user coordinate ID(101~120) or fail(-1)
success	bool	-	True or False

8.7.14 GetUserCartCoord.srv

Features

This service is only available for M2.50 or later versions.

This service returns the pose and reference coordinate system of the requested user coordinate system [id].

Parameters

Parameter Name	Data Type	Default Value	Description
id	int8	-	user coordinate ID

Return

Return Name	Data Type	Default Value	Description
conv_posx	float64[6]	-	position list
ref	int8	-	User coordination info.
success	bool	-	True or False

8.7.15 SetDesiredForce.srv

Features

This service defines the target force, direction, translation time, and mode for force control based on the global coordinate.

Parameters

Parameter Name	Data Type	Default Value	Description
Fd	float64[6]	[0, 0, 0, 0, 0, 0]	Three translational target forces Three rotational target moments

Parameter Name	Data Type	Default Value	Description
dir	int8[6]	[0, 0, 0, 0, 0, 0]	Force control in the corresponding direction if 1 Compliance control in the corresponding direction if 0
ref	int8	1	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120
time	float64	0.0	Transition time of target force to take effect [sec] Range: 0 - 1.0
mod	int8	0	FORCE_MODE_ABSOLUTE(0): Force control with absolute value FORCE_MODE_RELATIVE(1): force control with relative value to initial state (the instance when this function is called)

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7.16 ReleaseForce.srv

Features

This service reduces the force control target value to 0 through the time value and returns the task space to adaptive control.

Parameters

Parameter Name	Data Type	Default Value	Description
time	float64	0.0	Time needed to reduce the force Range: 0 - 1.0

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7.17 CheckPositionCondition.srv

Features

This service checks the status of the given position. This condition can be repeated with the while or if statement. Axis and pos of input paramets are based on the ref coordinate.

Parameters

Parameter Name	Data Type	Default Value	Description
axis	int8	-	FORCE_AXIS_X = 0 FORCE_AXIS_Y = 1 FORCE_AXIS_Z = 2
min	float64	0	Minimum value
max	float64	0	Maximum value
ref	int8	1	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120

Parameter Name	Data Type	Default Value	Description
mod	int8	-	MOVE_MODE_ABSOLUTE = 0 MOVE_MODE_RELATIVE = 1
pos	float64[6]	-	position list

Note

- The absolution position is used if the mod is DR_MV_MOD_ABS.
- The pos position is used if the mod is DR_MV_MOD_REL.
- Pos is meaningful only if the mod is DR_MV_MOD_REL.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7.18 CheckForceCondition.srv

Features

This service checks the status of the given force. It disregards the force direction and only compares the sizes. This condition can be repeated with the while or if statement. Measuring the force, axis is based on the ref coordinate and measuring the moment, axis is based on the tool coordinate.

Parameters

Parameter Name	Data Type	Default Value	Description
axis	int8	-	FORCE_AXIS_X = 0 FORCE_AXIS_Y = 1 FORCE_AXIS_Z = 2 FORCE_AXIS_A = 10 FORCE_AXIS_B = 11 FORCE_AXIS_C = 12

Parameter Name	Data Type	Default Value	Description
min	float64	-	Minimum value (min ≥ 0)
max	float64	-	Maximum value (max ≥ 0)
ref	int8	None	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120

Return

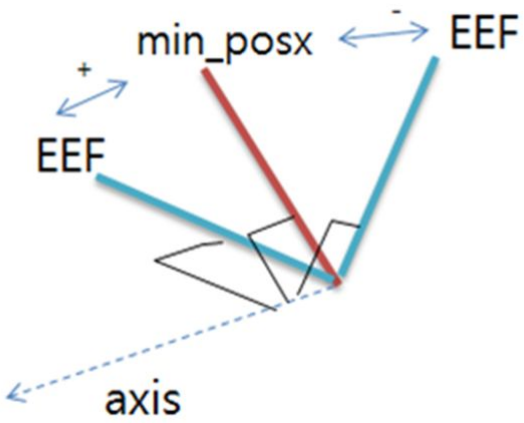
Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7.19 CheckOrientationCondition1.srv

Features

This service checks the difference between the current pose and the specified pose of the robot end effector. It returns the difference between the current pose and the specified pose in rad with the algorithm that transforms it to a rotation matrix using the “AngleAxis” technique. It returns True if the difference is positive (+) and False if the difference is negative (-). It is used to check if the difference between the current pose and the rotating angle range is + or -. For example, the function can use the direct teaching position to check if the difference from the current position is + or - and then create the condition for the orientation limit. This condition can be repeated with the while or if statement.

- Setting Min only: True if the difference is + and False if -
- Setting Min and Max: True if the difference from min is - while the difference from max is + and False otherwise
- Setting Max only: True if the maximum difference is + and False otherwise



Parameters

Parameter Name	Data Type	Default Value	Description
axis	int8	-	FORCE_AXIS_A = 10 FORCE_AXIS_B = 11 FORCE_AXIS_C = 12
min	float64[6]	-	position list
max	float64[6]	-	position list
ref	int8	1	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120
mod	int8	0	MOVE_MODE_ABSOLUTE = 0 MOVE_MODE_RELATIVE = 1

Return

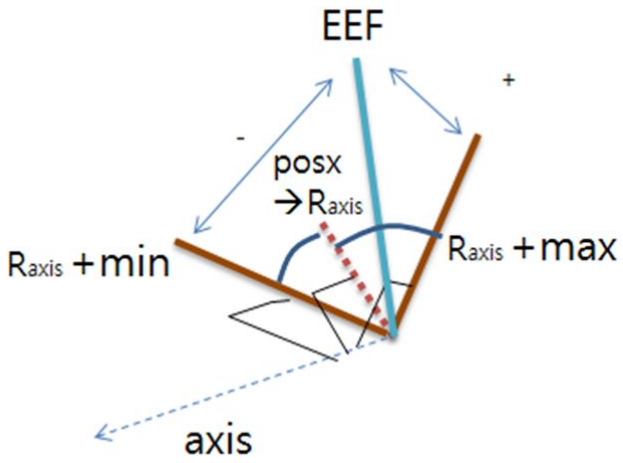
Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7.20 CheckOrientationCondition2.srv

Features

This service checks the difference between the current pose and the rotating angle range of the robot end effector. It returns the difference (in rad) between the current pose and the rotating angle range with the algorithm that transforms it to a rotation matrix using the “AngleAxis” technique. It returns True if the difference is positive (+) and False if the difference is negative (-). It is used to check if the difference between the current pose and the rotating angle range is + or -. For example, the function can be used to set the rotating angle range to min and max at any reference position, and then determine the orientation limit by checking if the difference from the current position is + or -. This condition can be repeated with the while or if statement.

- Setting Min only: True if the difference is + and False if -
- Setting Min and Max: True if the difference from min is - while the difference from max is + and False if the opposite.
- Setting Max only: True if the maximum difference is + and False otherwise



Note

Range of rotating angle: This means the relative angle range (min, max) based on the specified axis from a given position based on the ref coordinate.

Parameters

Parameter Name	Data Type	Default Value	Description
axis	int8	-	FORCE_AXIS_X = 0 FORCE_AXIS_Y = 1 FORCE_AXIS_Z = 2
min	float64	-	Minimum value
max	float64	-	Maximum value
ref	int8	1	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120
mod	int8	0	MOVE_MODE_ABSOLUTE = 0 MOVE_MODE_RELATIVE = 1
pos	float64[6]	-	position list

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.7.21 CoordTransform.srv

Features

This service transforms given task position expressed in reference coordinate, ‘ref_in’ to task position expressed in reference coordinate, ‘ref_out’. It returns transformed task position. It supports calculation of coordinate transformation for the following cases.

- (ref_in) world reference coordinate → (ref_out) world reference coordinate
- (ref_in) world reference coordinate → (ref_out) base reference coordinate
- (ref_in) world reference coordinate → (ref_out) tool reference coordinate

- (ref_in) world reference coordinate → (ref_out) user reference coordinate
- (ref_in) base reference coordinate → (ref_out) base reference coordinate
- (ref_in) base reference coordinate → (ref_out) tool reference coordinate
- (ref_in) base reference coordinate → (ref_out) user reference coordinate
- (ref_in) tool reference coordinate → (ref_out) world reference coordinate
- (ref_in) tool reference coordinate → (ref_out) base reference coordinate
- (ref_in) tool reference coordinate → (ref_out) tool reference coordinate
- (ref_in) tool reference coordinate → (ref_out) user reference coordinate
- (ref_in) user reference coordinate → (ref_out) world reference coordinate
- (ref_in) user reference coordinate → (ref_out) base reference coordinate
- (ref_in) user reference coordinate → (ref_out) tool reference coordinate
- (ref_in) user reference coordinate → (ref_out) user reference coordinate

Parameters

Parameter Name	Data Type	Default Value	Description
pose_in	float64[6]	-	position list
ref_in	int8	DR_COND_NONE	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120
ref_out	int8	DR_COND_NONE	MOVE_REFERENCE_BASE =0 MOVE_REFERENCE_TOOL=1 MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120

Return

Return Name	Data Type	Default Value	Description
conv_posx	float64[6]	-	position list
success	bool	-	True or False

8.7.22 GetWorkpieceWeight.srv

Features

This service measures and returns the weight of the workpiece.

Return

Return Name	Data Type	Default Value	Description
weight	float64	-	positive value : measured weight negative value : error
success	bool	-	True or False

8.7.23 ResetWorkpieceWeight.srv

Features

This is a service to initialize the weight information of the material to initialize the algorithm before measuring the weight of the material.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.8 Service/IO

8.8.1 SetCtlBoxDigitalOutput.srv

Features

This service sends a signal at the digital contact point of the controller..

Parameters

Parameter Name	Data Type	Default Value	Description
index	int8	-	I/O contact number mounted on the controller Val argument existing: A number between 1 and 16
value	int8		I/O value ON: 1 OFF: 0

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.8.2 GetCtlBoxDigitalOutput.srv

Features

This is a service to check the current output status of the digital contact mounted on the control box in the robot controller.

Parameters

Parameter Name	Data Type	Default Value	Description
index	int8	-	A number 1 - 16 which means the contact number of I/O mounted on the controller.

Return

Return Name	Data Type	Default Value	Description
value	int8	-	current stataus (OFF =0, ON =1)

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.8.3 GetCtlBoxDigitalInput.srv

Features

This service reads the signals from digital contact points of the controller.

Parameters

Parameter Name	Data Type	Default Value	Description
index	int8	-	A number 1 - 16 which means the contact number of I/O mounted on the controller.

Return

Return Name	Data Type	Default Value	Description
value	int8	-	OFF =0, ON =1
success	bool	-	True or False

8.8.4 SetToolDigitalOutput.srv

Features

This service sends the signal of the robot tool from the digital contact point.

Parameters

Parameter Name	Data Type	Default Value	Description
index	int8	-	I/O contact number mounted on the robot arm Val argument existing: A number between 1 and 6

Parameter Name	Data Type	Default Value	Description
value	int8		I/O value ON: 1 OFF: 0

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.8.5 GetToolDigitalOutput.srv

Features

This is a service to check the current signal status of the digital contact mounted on the end of the robot from the robot controller.

Parameters

Parameter Name	Data Type	Default Value	Description
index	int8	-	I/O contact number (1-6) mounted on the robot tool

Return

Return Name	Data Type	Default Value	Description
value	int8	-	current status (OFF =0, ON =1)
success	bool	-	True or False

8.8.6 GetToolDigitalIntput.srv

Features

This service reads the signal of the robot tool from the digital contact point.

Parameters

Parameter Name	Data Type	Default Value	Description
index	int8	-	I/O contact number (1-6) mounted on the robot tool

Return

Return Name	Data Type	Default Value	Description
value	int8	-	OFF =0, ON =1
success	bool	-	True or False

8.8.7 SetCtlBoxAnalogOutputType.srv

Features

This service sets the channel mode of the controller analog output.

Parameters

Parameter Name	Data Type	Default Value	Description
channel	int8	-	1 : channel 1 2 : channel 2
mode	int8	-	analog io mode current =0 voltage =1

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.8.8 SetCtlBoxAnalogInputType.srv

Features

This service sets the channel mode of the controller analog input.

Parameters

Parameter Name	Data Type	Default Value	Description
channel	int8	-	1 : channel 1 2 : channel 2
mode	int8	-	analog io mode current =0 voltage =1

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.8.9 SetCtlBoxAnalogOutput.srv

Features

This service outputs the channel value corresponding to the controller analog output.

Parameters

Parameter Name	Data Type	Default Value	Description
channel	int8	-	1 : channel 1 2 : channel 2
value	float64	-	analog output value Current mode: 4.0~20.0 [mA] Voltage mode: 0~10.0 [V]

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.8.10 GetCtlBoxAnalogInput.srv

Features

This service reads the channel value corresponding to the controller analog input.

Parameters

Parameter Name	Data Type	Default Value	Description
channel	int8	-	1 : channel 1 2 : channel 2

리턴

Return Name	Data Type	Default Value	Description
value	float	-	The analog input value of the specified channel Current mode: 4.0~20.0 [mA] Voltage mode: 0~10.0 [V]
success	bool	-	True or False

8.9 Service/modbus

8.9.1 ConfigCreateModbus.srv

Features

This service registers the Modbus signal. The Modbus I/O must be set in the Teach Pendant I/O set-up menu. Use this command only for testing if it is difficult to use the Teach Pendant. The Modbus menu is disabled in the Teach Pendant if it is set using this command.

Parameters

Parameter Name	Data Type	Default Value	Description
name	string	-	Modbus signal name
ip	string	-	IP address of the Modbus module
port	int8	-	Port number of the Modbus module
reg_type	int8	-	Modbus signal type MODBUS_REGISTER_TYPE_DISCRETE_INPUTS MODBUS_REGISTER_TYPE_COILS MODBUS_REGISTER_TYPE_INPUT_REGISTER MODBUS_REGISTER_TYPE_HOLDING_REGISTER
index	int8	-	Modbus signal index

Parameter Name	Data Type	Default Value	Description
value	int8	-	Output when the type is MODBUS_REGISTER_TYPE_COILS or MODBUS_REGISTER_TYPE_HOLDING_REGISTER (ignored otherwise)
slaveid	int	255	Slave ID of the ModbusTCP module (0 or 1-247 or 255) 0 : Broadcase address 255 : Default value for ModbusTCP

Note

The slaveid argument is only available for versions M2.40 and higher.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.9.2 ConfigDeleteModbus.srv

Features

This service deletes the registered Modbus signal. The Modbus I/O must be set in the Teach Pendant I/O set-up menu. Use this command only for testing if it is difficult to use the Teach Pendant. The Modbus menu is disabled in the Teach Pendant if it is set using this command.

Parameters

Parameter Name	Data Type	Default Value	Description
name	string	-	Name of the registered Modbus signal

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.9.3 SetModbusOutput.srv

Features

This service sends the signal to an external Modbus system.

Parameters

Parameter Name	Data Type	Default Value	Description
name	string	-	modbus name
value	int32	-	Modbus digital I/O • ON : 1 • OFF : 0 Value for Modbus analog I/O : Data

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.9.4 GetModbuInput.srv

Features

This service reads the signal from the Modbus system.

Parameters

Parameter Name	Data Type	Default Value	Description
name	string	-	modbus name

Return

Return Name	Data Type	Default Value	Description
value	int32	-	ON or Off in the case of the Modbus digital I/O The register value in the case of the Modbus analog module
success	bool	-	True or False

8.10 Service/DRL

8.10.1 DrlStart.srv

Features

This service is used to execute DRL script. (DRL is a Doosan robot language).

> **Caution**
>
> The drl service is only available in real mode.

Parameters

Parameter Name	Data Type	Default Value	Description
robotSystem	int8	-	
Code	string	-	DRL code string

Return

Parameter Name	Data Type	Default Value	Description
success	bool	-	True or False

Note

- Robot operation status should be in STANDBY state (STATE_STANDBY) and should be used when robot mode is in auto mode.
- DRL programming should be done by referring to the Programming Manual.

8.10.2 DrlStop.srv

Features

This service is used to stop the DRL program (task) currently running on the robot controller. Stops differently according to the eStopType received as an argument, and stops the motion of the current section

Caution

The drl service is only available in real mode

Parameters

Return Name	Data Type	Default Value	Description
stop_mode	int8	-	drl stop mode STOP_TYPE_QUICK_STO = 0 STOP_TYPE_QUICK = 1 STOP_TYPE_SLOW = 2 STOP_TYPE_HOLD = STOP_TYPE_EMERGENCY = 3

Return

Return Name	Data Type	Default Value	Description
Success	bool	-	True or False

8.10.3 DrlPause.srv

Features

This is a service to temporarily stop the DRL program (task) currently running on the robot controller.

> **Caution**
>
> The drl service is only available in real mode.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.10.4 DrlResume.srv

Features

This is a service to resume the currently paused DRL program (task) from the robot controller.

> **Caution**
>
> The drl service is only available in real mode.

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.10.5 GetDrlState.srv

Features

This service reads the status of the DRL program

> **Caution**
>
> The drl service is only available in real mode.

Return

Return Name	Data Type	Default Value	Description
state	int8	-	state of DRL program . 0 : DRL_PROGRAM_STATE_PLAY . 1 : DRL_PROGRAM_STATE_STOP . 2 : DRL_PROGRAM_STATE_HOLD
success	bool	-	True or False

8.11 Service/gripper

8.11.1 SerialSendData.srv

Features

It controls the gripper through actual serial communication.

serial_node_example

Parameters

Parameter Name	Data Type	Default Value	Description
data	string	-	String to send

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

8.11.2 RobotiqMove.srv

Features

It is a service to control robotiq's gripper in simulator environment.

Parameters

Parameter Name	Data Type	Default Value	Description
width	float	-	Width of finger gripper : 0.0(open)~0.8(close)

Return

Return Name	Data Type	Default Value	Description
success	bool	-	True or False

9 DRL Command

Robot mode is a ROS-only DRL command. For the latest DRL commands other than robot mode, refer to Programming manual[1].

9.1 Robot Mode

9.1.1 set_robot_mode(robot_mode)

Features

This function is to set the operation mode (manual / automatic) of the robot controller. Motion command is available in both modes, but manual handguiding is available in manual mode and operates in deceleration mode for safety.

In automatic mode, manual handguiding is not possible and operates in normal speed mode.

Manual mode: robot LED lights up blue

Auto mode: robot LED lights up in white

Parameters

Parameter Name	Data Type	Default Value	Description
Robot_mode	int	-	• ROBOT_MODE_MANUAL : 0 • ROBOT_MODE_AUTONOMOUS : 1

Return

Value	Description
0	Success
Negative value	Failed

Example

```
1 #...
```

1 https://manual.doosanrobotics.com/display/Programming

```
set_robot_mode(ROBOT_MODE_MANUAL)
#...
#...
#...
set_robot_mode(ROBOT_MODE_AUTONOMOUS)
#...
```

9.1.2 get_robot_mode()

Features

This function reads the current operating mode of the robot controller.

Return

Value	Description
ROBOT_MODE_MANUAL (0)	Robot mode is manual.
ROBOT_MODE_AUTONOMOUS (1)	Robot mode is auto.

Example

```
#...
if get_robot_mode() != ROBOT_MODE_AUTONOMOUS:
set_robot_mode(ROBOT_MODE_AUTONOMOUS)
#...
```

9.1.3 get_last_alarm()

Feature

This function reads the latest alarm log of the robot controller.

Return

Value	Description
level	Log message level 0 : Info 1 : Warning 2 : Alarm

Value	Description
group	Log message group
index	Alarm log number
param	Alarm log parameters

Example

```
#...
index = get_last_alarm().index
print(str(index))
#...
```

9.1.4 set_safe_stop_reset_type(reset_type)

Feature

When the operation status information of the robot controller is SAFE_STOP, it is a function to define a series of actions that are automatically executed after the change of status using the set_robot_control function. If the robot operation mode is automatic, the program can be defined and set again. In the manual mode, this setting is ignored.

Parameters

Parameter Name	Data Type	Default Value	Description
reset_type	int8	0	Reset Type STOP : 1 RESUME : 2

Return

Value	Description
0	Success
1	Failed

Example

```
#...
robot_state = get_robot_mode()
if robot_state == ROBOT_MODE_AUTONOMOUS:
  set_safe_reset_type(SAFE_STOP_PROGRAM_RESUME);
#...
```

9.1.5 set_robot_speed_mode(speed_mode)

Feature

This is a function for setting and changing the speed mode currently being operated by the controller.

Parameters

Parameter Name	Data Type	Default Value	Description
speed_mode	int8	-	Robot Speed mode 0 : normal mode 1 : reduced mode

Return

Value	Description
0	Failed
1	Success

Example

```
if get_robot_speed_mode() == SPEED_REDUCED_MODE:
    #When in deceleration mode, change to constant speed mode
set_robot_speed_mode(SPEED_NORMAL_MODE)
```

9.1.6 get_robot_speed_mode()

Feature

This is a function to check the current speed mode (normal mode, deceleration mode) information from the robot controller.

Return

Value	Description
robot_speed	robot speed mode 0 : normal mode 1 : reduced mode

Example

```
if get_robot_speed_mode() == SPEED_REDUCED_MODE:
   # When in deceleration mode, change to constant speed mode
set_robot_speed_mode(SPEED_NORMAL_MODE)
```

9.1.7 set_robot_system(robot_system)

Feature

This is a function for setting and changing the current operating robot system in the robot controller.

Parameters

Parameter Name	Data Type	Default Value	Description
robot_system	int8	-	robot system info 0 : Real 1 : Virtual

Return

Value	Description
0	Failed
1	Success

Example

```
if(get_robot_system() != ROBOT_SYSTEM_REAL):
    #Automatic mode switching
    set_robot_system(ROBOT_SYSTEM_REAL)
else :
  #do somting …
```

9.1.8 get_robot_system()

Feature

This function reads the current operating robot system from the robot controller.

Return

Value	Description
0	Real
1	Virtual

Example

```
if get_robot_system() != ROBOT_SYSTEM_REAL:
    # Automatic mode switching
    set_robot_system(ROBOT_SYSTEM_REAL)
else:
  #do somting …
```

9.1.9 get_robot_state()

Feature

This is for checking the current operation status information of the robot controller.

Return

Value	Description
robot_state	Robot status 0 : Initializing 1 : Standby 1 : Moving 2 : SAFE OFF 3 : Teaching 4 : SAFE STOP 5 : Emergency Stop 6 : Homming 7 : Recovery

Example

```
if get_robot_state() == STATE_STANDBY:
    if get_robot_mode() == ROBOT_MODE_AUTONOMOUS:
        # Manual Mode
        # do something
```