ROS

Doosan Robot

M0609 | M0617 | M1013 | M1509

ROS Programming Manual

DOOSAN

1. Doosan ROS package 설치

1.1 Overview

- **Doosan robotics ROS package**

Doosan robotics ROS 패키지는 ROS 상에서 두산 협동로봇을 구동하기 위한 메타 패키지로 URDF 모델을 제공하고 Rviz, Gazebo 를 통하여 시뮬레이션이 가능하며, moveIt 이나 다양한 예제를 통하여 실제 로봇을 구동 시킬 수 있습니다.

1.2 제약 사항

- **System**

사용자 PC는 X86 시스템이어야 합니다.

원할한 시뮬레이션을 위해서는 워크스테이션급 PC를 권장합니다.

- **OS 및 ROS version**

Ubuntu 16.04(32/64bit) + ROS kinetic

1.3 설치

- **소스 설치**

Doosan robotics Github 에서 소스를 다운 받아서 빌드 합니다

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

- **설치 방법**

```
### 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
```

2. 동작 모드

2.1 Virtual mode

▪ 기능

- 실제 로봇이 없이 구동하는 경우에는 virtual mode를 사용합니다.
- dsr_launcher 파일 실행 시, mode 인자를 virtual 로 설정합니다

  인자를 생략 시, default 로 virtual 로 설정됩니다.

  ex> roslaunch dsr_launcher single_robot_gazebo.launch **mode:=virtual**
- virtual mode 로 ROS launch 시, 자동으로 애뮬레이터(DRCF)가 실행됩니다.
  - 애뮬레이터(DRCF) 위치 : doosan-robot/common/bin/DRCF
- 로봇 한대 당 하나의 애뮬레이터가 필요합니다.
- 다중로봇 제어 시, 로봇 개수만큼 애뮬레이터가 자동 실행되며 각각 다른 포트를 사용하게 됩니다.

2.2 Real mode

▪ 기능

- 실제 로봇을 구동할 경우에는 real mode를 사용합니다.
- real mode 동작 시에는 실제 로봇 제어기와 통신 연결을 해야만 합니다.
- 로봇 제어기의 디폴트 IP는 192.168.127.100, port는 12345 입니다.
- dsr_launcher 파일 실행 시, mode:=real host:=192.168.127.100 port:=12345 로 설정합니다

  ex> roslaunch dsr_launcher single_robot_gazebo.launch **mode:=real host:=192.168.127.100 port:=12345**

▪ 실제 제어기 연결

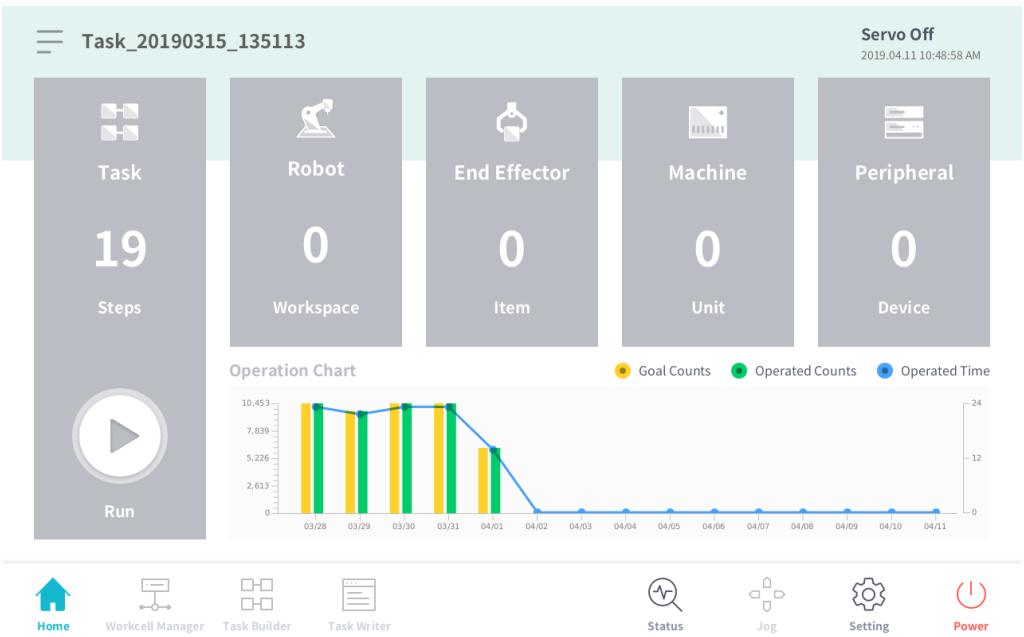


그림 2.2 TP 화면

- 사용자는 TP 화면의 Setting -> Network에서 고정 IP를 설정할 수 있습니다.

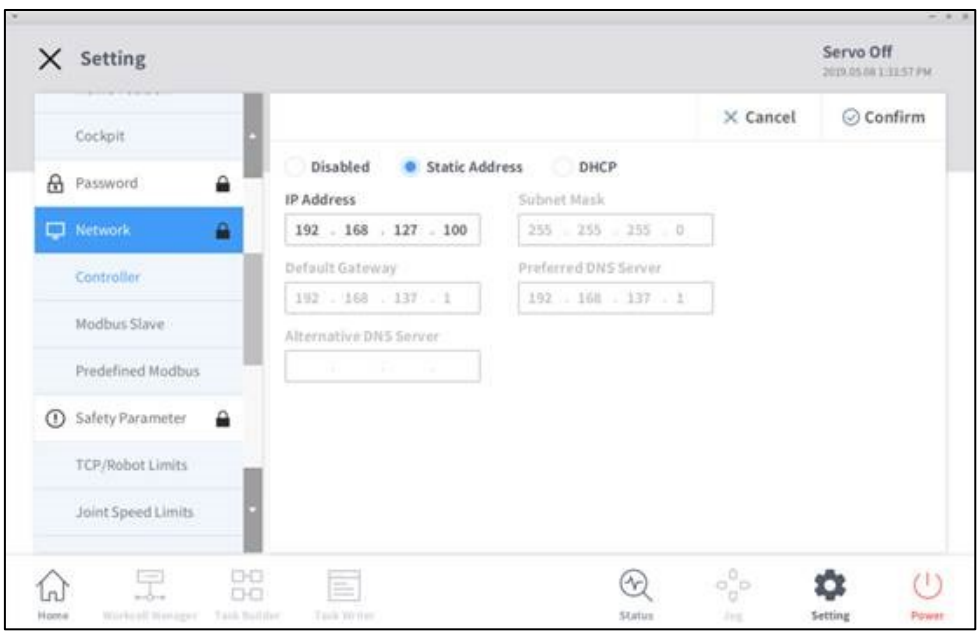


그림 2.3 TP 상에서 제어기 IP 확인

- Network 탭에서 설정된 제어기의 IP를 확인하고, 이 IP를 ROS 상에서 host := ROBOT_IP

로 사용합니다.

- ROS Control Node가 올바르게 실행되었다면, TP의 제어권이 ROS로 넘어갑니다.
- 제어권이 넘어간 TP 화면에는 아래와 같은 팝업이 출력됩니다.

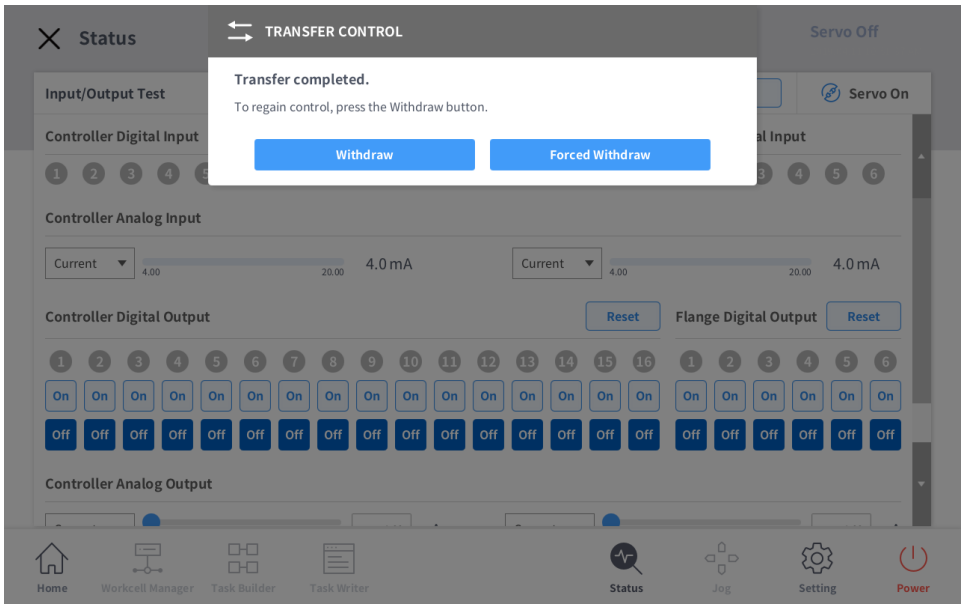


그림 2.4 제어권 이전 시 발생 팝업

3. dsr_description

3.1 dsr_description <robot_model>.launch

- **기능**
  - Rviz 상에 로봇 모델을 띄우고, Joint_state_publisher 를 로딩합니다.
  - 로딩된 Joint_state_publisher를 통하여 로봇을 움직입니다.

- **파라미터**

인수명	자료형	기본값	설명
model	-	m1013	M-Series 로봇 모델 . m0609, m0617, m1013, m1509 A-Series 로봇 모델 . a0509
color	-	white	로봇 컬러 . white or blue
gripper	-	none	gripper 사용 유무 . none : gripper 미 사용 . robotiq_2f : robotiq two finger 장착

- **예제**

```
$ 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
```

하기 3.1 그림과 같이 Rviz 상에 로봇과 Joint_state_publisher 가 로딩 됨.
Joint_state_publisher 를 통하여 로봇을 구동 시킴.

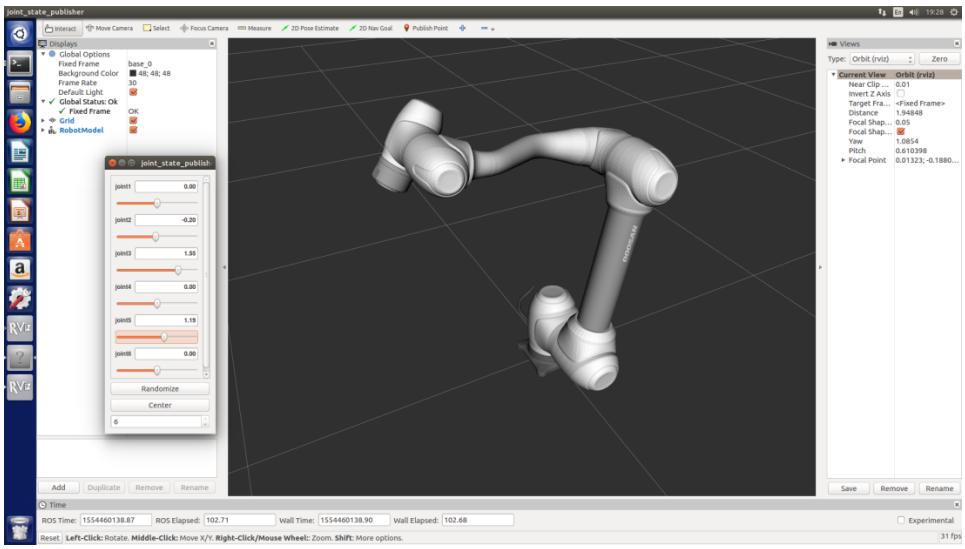


그림 3.1 Rviz 상에 로봇

4. dsr_moveit_config

4.1 dsr_moveit_config

- **기능**
  - Rviz 상에 로봇 모델을 띄우고 moveit을 통하여 로봇을 제어합니다.
  - 시뮬레이션 모드로만 동작합니다.

- **파라미터**

인수명	자료형	기본값	설명
color	-	white	로봇 컬러 . white or blue

- **예제**

```
$ 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
```

하기 4.1 그림과 같이 Rviz 상에 로봇과 MotionPlanning 창이 로딩 됨.
MotionPlanning 를 통하여 로봇을 가상으로 구동 시킴.

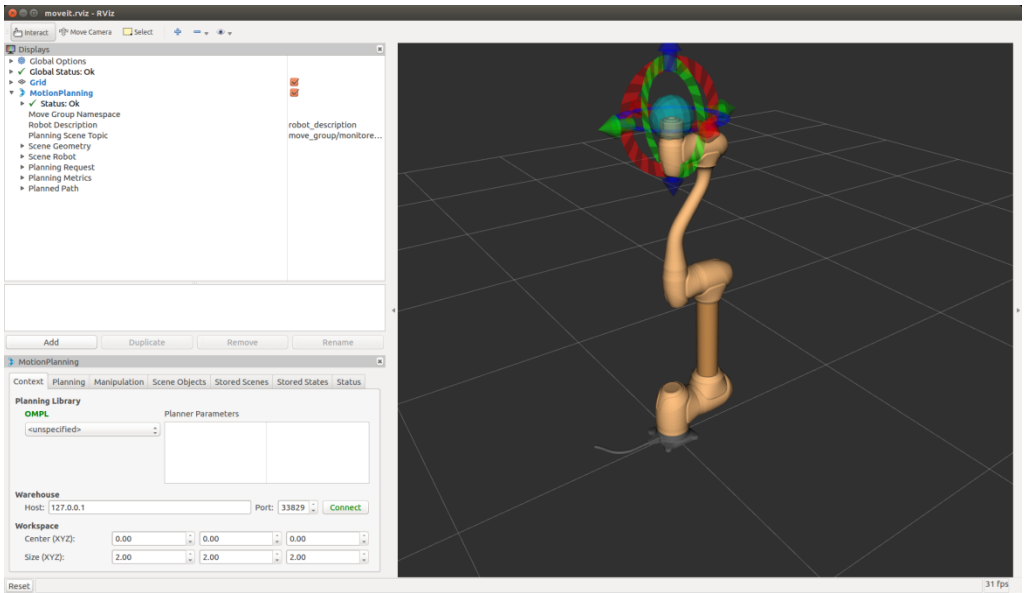


그림 4.1 Rviz + MoveIt

4.2 dsr_control + moveit

▪ 기능

- Rviz 상에 로봇 모델을 띄우고 moveit을 통하여 로봇을 제어합니다.
- 애뮬레이터 모드나 실제 로봇과 연결하여 동작합니다.
- 애뮬레이터 모드는 virtual mode만 작동합니다.

▪ 파라미터

인수명	자료형	기본값	설명
host	-	127.0.0.1	로봇 제어기 IP . 애뮬레이터 : 127.0.0.1 . 실제 로봇제어기 : 192.168.127.100
Port	-	12345	서비스 port
mode	-	virtual	로봇 동작 모드 - virtual : 가상 동작 - real : 실제 동작
model	-	m1013	M-Series 로봇 모델 . m0609, m0617, m1013, m1509 A-Series 로봇 모델 . a0509 로봇 모델 (4종) . m0609, m0617, m1013, m1509
color	-	white	로봇 컬러 . white or blue
gripper	-	none	gripper 사용 유무 . none : gripper 미 사용 . robotiq_2f : robotiq two finger 장착

▪ 예제

<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>

로봇제어기 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**

하기 4.2 그림과 같이 Rviz 상에 로봇과 MotionPlanning 창이 로딩 됨.
MotionPlanning 를 통하여 로봇을 구동 시킴.

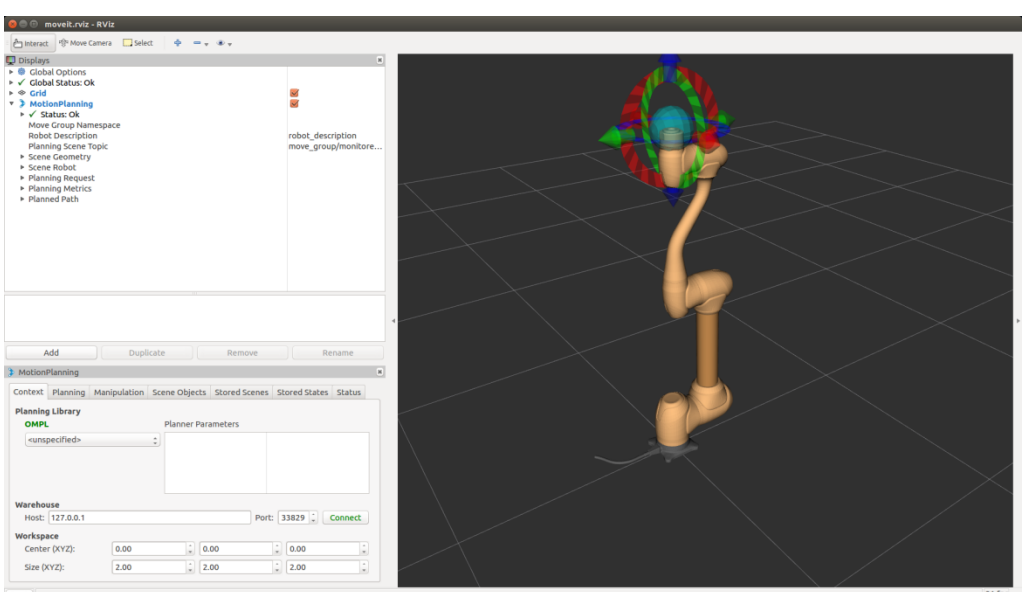


그림 4.2 Rviz + dsr_control

4.3 MoveIt Commander

- **기능**
  - Moveit Commander 를 통하여 로봇을 제어합니다.

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

- **예제**

<virtual mode>

$ roslaunch dsr_control dsr_moveit.launch model:=**m1013 mode:=virtual**

$ ROS_NAMESPACE=/dsr01**m1013** rosrun moveit_commander moveit_commander_cmdline.py robot_description:=/dsr01**m1013**/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>

로봇제어기 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=/dsr01**a0509** rosrun moveit_commander moveit_commander_cmdline.py robot_description:=/dsr01**a0509**/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)
```

5. dsr_launcher

5.1 dsr_launcher

▪ 기능

- dsr_launcher 를 통하여 다양한 로봇 환경을 구성합니다.
- 파라미터에 따라 Single Robot, Multi Robot, Gripper, mobile 환경을 구축할 수 있습니다.
- Multi Robot 구성은 Single Robot의 구성의 확장입니다. 본 패키지에 기본적으로 제공되는 Multi Robot 구성 2대의 로봇에 대한 예제입니다. 자신의 환경에 맞는 구성을 위해서는 dsr_launcher/multi-robot*.launch 파일들을 적절히 수정하시길 바랍니다. (ns, host, port, model 등을 올바르게 수정해야 합니다.)
- dsr_launcher 로딩 후, 각 환경에 맞는 dsr_example 을 rosrun 으로 구동시킵니다.
  (상세 내역은 6장. dsr_example 을 참조하시기 바랍니다.)

▪ 파라미터

인수명	자료형	기본값	설명
ns	-	dsr01	ROBOT name space . single robot : dsr01 . multi robot: dsr01 부터 순차적으로 dsr02, dsr03, dsr04 …
host	-	127.0.0.1	로봇 제어기 IP . 애뮬레이터 : 127.0.0.1 . 실제 로봇제어기 : 192.168.127.100
port	-	12345	서비스 port
mode	-	virtual	로봇 동작 모드 - virtual : 가상 동작 - real : 실제 동작
model	-	m1013	M-Series 로봇 모델 . m0609, m0617, m1013, m1509 A-Series 로봇 모델

인수명	자료형	기본값	설명
			. a0509
color	-	white	로봇 컬러 . white or blue
gripper	-	none	gripper 사용 유무 . none : gripper 미 사용 . robotiq_2f : robotiq two finger 장착
mobile	-	none	Mobile robot 사용 유무 . none : 미 사용 . husky : husky 모바일 로봇 사용

- **예제**

<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
```

6. dsr_example

▪ 기능

- dsr_launcher 를 통하여 구성된 로봇 환경에 따른 로봇 구동 예제를 제공합니다.
  (자세한 로봇 환경 구성은 5장. dsr_launcher 을 참조하시기 바랍니다.)
- 예제 파일은 python 으로 작성 되어 있습니다.
  - python 소스 위치: ~/catkin_ws/src/doosan-robot/dsr_example/py/scripts

6.1 Single Robot

▪ 기능

- Single Robot 구동 예제를 제공합니다.
  (자세한 로봇 환경 구성은 5장. dsr_launcher 을 참조하시기 바랍니다.)
- 예제 파일은 python 으로 작성 되어 있습니다.
  - python 소스 위치: ~/catkin_ws/src/doosan-robot/dsr_example/py/scripts/simple

▪ dsr_launcher 인자

인수명	자료형	기본값	설명
ns	-	dsr01	ROBOT name space . single robot : dsr01 . multi robot: dsr01 부터 순차적으로 dsr02, dsr03, dsr04 …
host	-	127.0.0.1	로봇 제어기 IP . 애뮬레이터 : 127.0.0.1 . 실제 로봇제어기 : 192.168.127.100
port	-	12345	서비스 port
mode	-	virtual	로봇 동작 모드 - virtual : 가상 동작 - real : 실제 동작
model	-	m1013	M-Series 로봇 모델 . m0609, m0617, m1013, m1509 A-Series 로봇 모델 . a0509
color	-	white	로봇 컬러 . white or blue
gripper	-	none	gripper 사용 유무 . none : gripper 미 사용 . robotiq_2f : robotiq two finger 장착

인수명	자료형	기본값	설명
mobile	-	none	Mobile robot 사용 유무 . none : 미 사용 . husky : husky 모바일 로봇 사용

- **예제**

1. 로봇 제어기 default IP/Port

- IP : 192.168.127.100 , port = 12345

2. launch

원하는 구성에 맞게 dsr_launher 를 실행합니다.

- 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

- 예제 파일 수정

. 구동하고자 하는 예제 파일을 열어 ROBOT_ID 와 ROBOT_MODEL을 맞게 수정합니다.

.. ex>

ROBOT_ID = "dsr01"

ROBOT_MODEL = "m1013"

$ rosrun dsr_example_py **single_robot_simple.py**

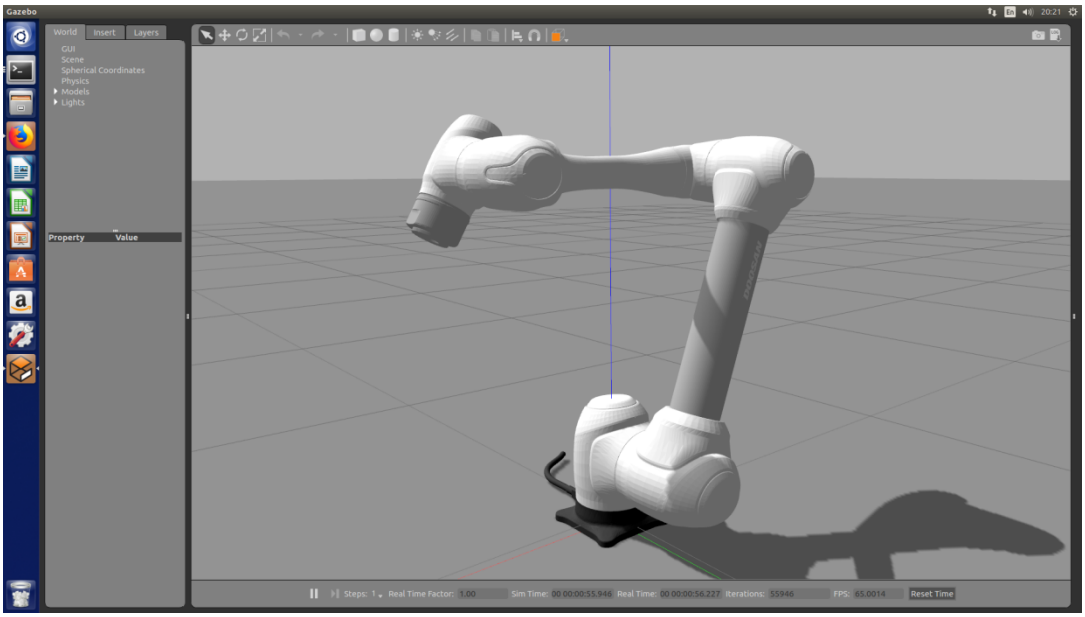


그림 6.2 single robot

6.2 Multi Robot

▪ 기능

- Multi Robot 구동 예제를 제공합니다.

  (자세한 로봇 환경 구성은 5장. dsr_launcher 을 참조하시기 바랍니다.)

- 예제 파일은 python 으로 작성 되어 있습니다.

  - python 소스 위치: ~/catkin_ws/src/doosan-robot/dsr_example/py/scripts/simple

▪ dsr_launcher 인자

인수명	자료형	기본값	설명
ns	-	dsr01	ROBOT name space . single robot : dsr01 . multi robot: dsr01 부터 순차적으로 dsr02, dsr03, dsr04 …
host	-	127.0.0.1	로봇 제어기 IP . 애뮬레이터 : 127.0.0.1 . 실제 로봇제어기 : 192.168.127.100
port	-	12345	서비스 port
mode	-	virtual	로봇 동작 모드 - virtual : 가상 동작 - real : 실제 동작
model	-	m1013	M-Series 로봇 모델 . m0609, m0617, m1013, m1509 A-Series 로봇 모델 . a0509
color	-	white	로봇 컬러 . white or blue
gripper	-	none	gripper 사용 유무 . none : gripper 미 사용 . robotiq_2f : robotiq two finger 장착

인수명	자료형	기본값	설명
mobile	-	none	Mobile robot 사용 유무 . none : 미 사용 . husky : husky 모바일 로봇 사용

- **예제**

1. 로봇 제어기 default IP/Port
- IP : 192.168.127.100 , port = 12345
- multi robot 인 경우 각 로봇 제어기의 IP를 다르게 설정합니다.

1. launch
- launch 파일 수정
. $ cd ~/catkin_ws/src/doosan-robot/dsr_launcher/launch
. multi_robot_*. launch 파일을 각 상황에 맞게 수정합니다.
.. ns, host, port, model 등을 올바르게 수정해야 합니다.
- 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
- 예제 파일 수정
. 구동하고자 하는 예제 파일을 열어 robot_id 와 robot_model을 맞게 수정합니다.
.. ex>
robot_id1 = "dsr01"; robot_model1 = "m1013"
robot_id2 = "dsr02"; robot_model2 = "m1013"

$ rosrun dsr_example_py **multi_robot_simple.py**

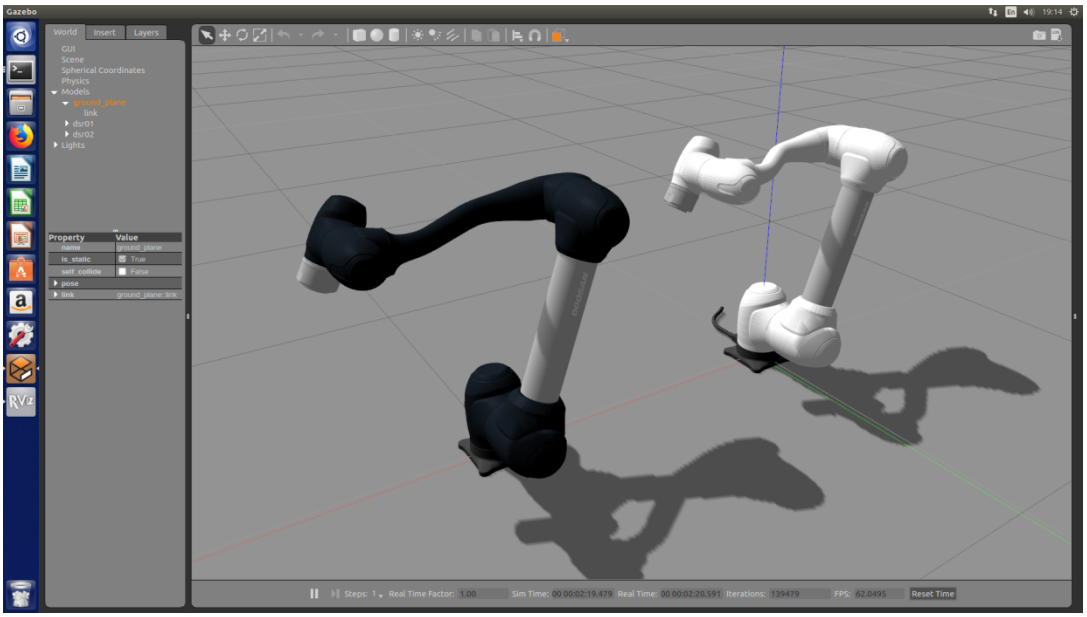


그림 6.3 multi robot

6.3 Gripper

▪ 기능

- Gripper (robotiq_2f) 사용 예제를 제공합니다.
  (자세한 로봇 환경 구성은 5장. dsr_launcher 을 참조하시기 바랍니다.)
- dsr_launcher 실행 시, gripper:=robotiq_2f 인자를 줍니다.
- 예제 파일은 python 으로 작성 되어 있습니다.
  - python 소스 위치: ~/catkin_ws/src/doosan-robot/dsr_example/py/scripts/gripper

▪ dsr_launcher 인자

인수명	자료형	기본값	설명
ns	-	dsr01	ROBOT name space . single robot : dsr01 . multi robot: dsr01 부터 순차적으로 dsr02, dsr03, dsr04 …
host	-	127.0.0.1	로봇 제어기 IP . 애뮬레이터 : 127.0.0.1 . 실제 로봇제어기 : 192.168.127.100
port	-	12345	서비스 port
mode	-	virtual	로봇 동작 모드 - virtual : 가상 동작 - real : 실제 동작
model	-	m1013	M-Series 로봇 모델 . m0609, m0617, m1013, m1509 A-Series 로봇 모델 . a0509
color	-	white	로봇 컬러 . white or blue
gripper	-	none	gripper 사용 유무 . none : gripper 미 사용 . robotiq_2f : robotiq two finger 장착

인수명	자료형	기본값	설명
mobile	-	none	Mobile robot 사용 유무 . none : 미 사용 . husky : husky 모바일 로봇 사용

- **예제**

1. 로봇 제어기 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

- 예제 파일 수정

. 구동하고자 하는 예제 파일을 열어 ROBOT_ID 와 ROBOT_MODEL을 맞게 수정합니다.

.. ex>

ROBOT_ID = "dsr01"

ROBOT_MODEL = "m1013"

$ rosrun dsr_example_py **pick_and_place.py**

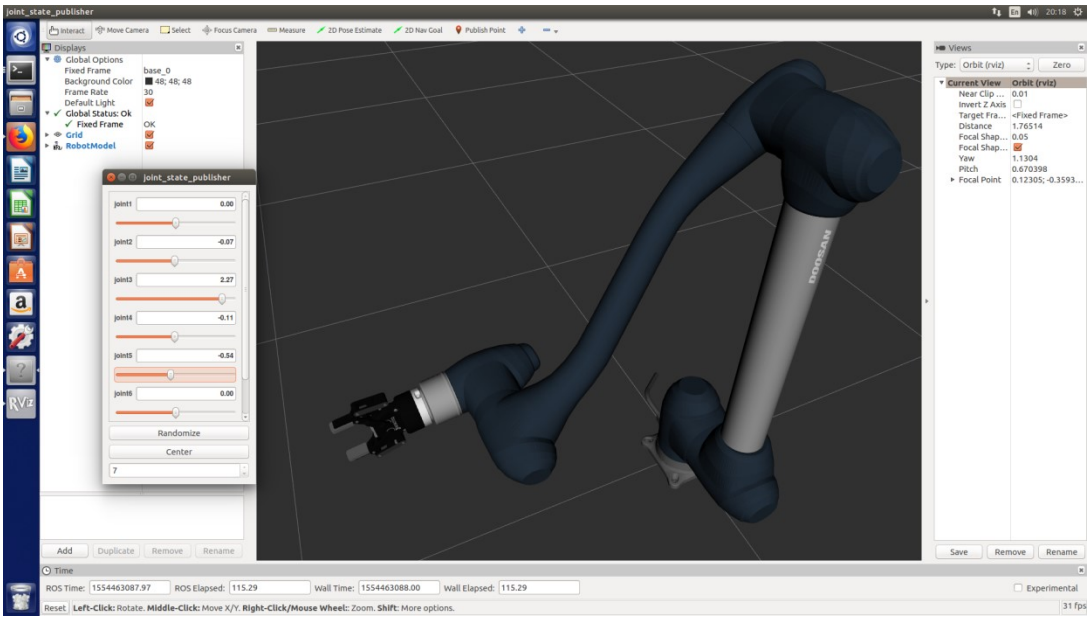


그림 6.4 robot + gripper

6.4 Mobile robot

▪ 기능

- 모바일 로봇(husky) 예제를 제공합니다.

  (자세한 로봇 환경 구성은 5장. dsr_launcher 을 참조하시기 바랍니다.)
- dsr_launcher 실행 시, mobile:=husky 인자를 줍니다.
- 예제 파일은 python 으로 작성 되어 있습니다.

  - python 소스 위치: ~/catkin_ws/src/doosan-robot/dsr_example/py/scripts/mobile

▪ dsr_launcher 인자

인수명	자료형	기본값	설명
ns	-	dsr01	ROBOT name space . single robot : dsr01 . multi robot: dsr01 부터 순차적으로 dsr02, dsr03, dsr04 …
host	-	127.0.0.1	로봇 제어기 IP . 애뮬레이터 : 127.0.0.1 . 실제 로봇제어기 : 192.168.127.100
port	-	12345	서비스 port
mode	-	virtual	로봇 동작 모드 - virtual : 가상 동작 - real : 실제 동작
model	-	m1013	M-Series 로봇 모델 . m0609, m0617, m1013, m1509 A-Series 로봇 모델 . a0509
color	-	white	로봇 컬러 . white or blue
gripper	-	none	gripper 사용 유무 . none : gripper 미 사용 . robotiq_2f : robotiq two finger 장착
mobile	-	none	Mobile robot 사용 유무

인수명	자료형	기본값	설명
			. none : 미 사용 . husky : husky 모바일 로봇 사용

- **예제**

1. 로봇 제어기 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

- 예제 파일 수정

. 구동하고자 하는 예제 파일을 열어 ROBOT_ID 와 ROBOT_MODEL을 맞게 수정합니다.

.. ex>

ROBOT_ID = "dsr01"

ROBOT_MODEL = "m1013"

$ rosrun dsr_example_py **single_robot_moblie.py**

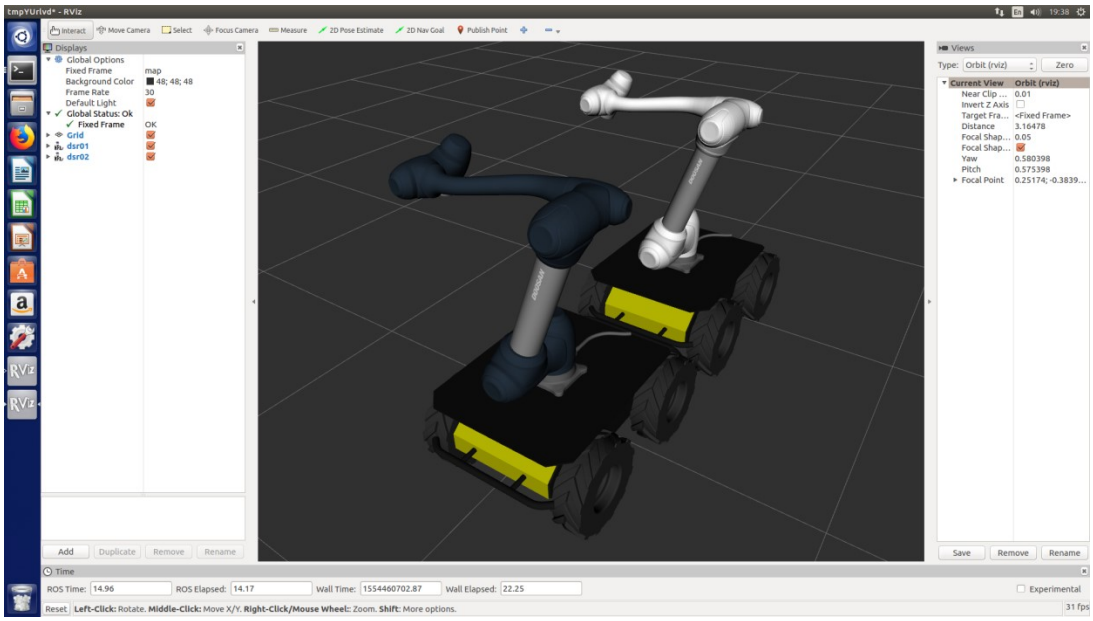


그림 6.5robot on mobile

7. dsr_msgs

7.1 Topic

7.1.1 RobotState.msg

- **기능**

로봇 상태 토픽 메시지

- **인수**

인수명	자료형	기본값	설명
robot_state	int32		로봇 상태 : enum.ROBOT_STATE 참조.
robot_state_str	string		로봇 상태를 string으로 표기
actual_mode	int8		로봇 제어 모드: enum.CONTROL_MODE 참조
actual_space	int8		로봇 제어 공간: enum.ROBOT_SPACE 참조
current_posj	float64[6]		6개의 현재 조인트 각도 정보
joint_abs	float64[6]		6개의 현재 조인트 각도 정보(Absolute Encorder)
current_velj	float64[6]		6개의 현재 조인트 속도 정보
joint_err	float64[6]		6개의 현재 조인트 오차 정보
target_posj	float64[6]		6개의 타겟 조인트 위치 정보
target_velj	float64[6]		6개의 타겟 조인트 속도 정보
current_posx	float64[6]		6개의 현재 Task 위치 정보
current_tool_posx	float64[6]		6개의 현재 Flange 위치 정보
current_velx	float64[6]		6개의 현재 Task 속도 정보
task_err	float64[6]		6개의 현재 Task 오차 정보
target_posx	float64[6]		6개의 타겟 Task 위치 정보
target_velx	float64[6]		6개의 타겟 Task 속도 정보

인수명	자료형	기본값	설명
solution_space	int8		로봇의 자세 정보(0~7)
rotation_matrix	std_msgs/Float64MultiArray[]		로봇 회전 행렬 정보: float[3][3]
dynamic_tor	float64[6]		6개의 현재 계산된 다이나믹 토크 정보
actual_jts	float64[6]		6개의 현재 측정된 JTS 센서 정보
actual_ejt	float64[6]		6개의 현재 Joint 축별 외력 정보
actual_ett	float64[6]		6개의 현재 Task 기반 외력 정보
actual_w2b	float64[6]		6개의 World 좌표계-Base 좌표계 간 위치 편차 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
current_posw	std_msgs/Float64MultiArray[]		World 좌표계기준 6개의 현재 Task 위치 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
current_velw	float64[6]		World 좌표계 기준 6개의 현재 Task 속도 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
world_ett	float64[6]		World 좌표계 기준 6개의 현재 Task 기반 외력 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
target_posw	float64[6]		World 좌표계 기준 6개의 타겟 Task 위치 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
target_velw	float64[6]		World 좌표계 기준 6개의 타겟 Task 속도 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**

인수명	자료형	기본값	설명
rotation_matrix_world	std_msgs/Float64MultiArray[]		World 좌표계 기준 로봇 회전 행렬 정보: float[3][3] **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
actual_UCN	int8		사용자 좌표계 ID 정보(101 ~ 200) **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
parent	int8		User좌표계의 Parent 좌표계 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
current_posu	float64[6]		사용자 좌표계 기준 6개의 현재 Task 위치 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
current_velu	float64[6]		사용자 좌표계 기준 6개의 현재 Task 속도 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
user_ett	float64[6]		사용자 좌표계 기준 6개의 현재 Task 기반 외력 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
target_posu	float64[6]		사용자 좌표계 기준 6개의 타겟 Task 위치 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
target_velu	float64[6]		사용자 좌표계 기준 6개의 타겟 Task 속도 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
rotation_matrix_user	std_msgs/Float64MultiArray[]		사용자 좌표계 기준 로봇 회전 행렬 정보: float[3][3] **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
sync_time	int8		내부 클락 카운터 정보

인수명	자료형	기본값	설명
flange_digital_output	int8[6]		6개의 Flange Digital Output 정보
flange_digital_input	int8[6]		6개의 Flange Digital Input 정보
actual_bk	int8[6]		6개의 브레이크 상태 정보
actual_bt	int8[5]		5개의 로봇 버튼 정보
actual_mc	float64[6]		6개 모터의 전류 소모량 정보
actual_mt	float64[6]		6개 인버터 온도 정보
ctrlbox_digital_input	int8[6]		16개 컨트롤 박스 digital intput 정보
actual_ai	int8[2]		2개의 Analog Input 수치 정보
actual_sw	int8[3]		3개의 Switch 상태 정보
actual_si	int8[2]		2개의 Safety Input 정보
actual_at	int8[2]		2개의 Analog Input Type 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
ctrlbox_digital_output	int8[16]		16개 컨트롤 박스 digital output 정보
target_ao	float64[2]		2개의 Analog Output 수치 정보
target_at	int8[2]		2개의 Analog Output Type 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
actual_es	int8[2]		2개의 Encorder 상태 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
actual_ed	int8[2]		2개의 Encorder Raw 데이터 수치 정보
actual_er	int8[2]		2개의 Encorder Reset 상태 정보 **해당 인수는 M2.50 이상의 버전에서만 사용 가능합니다.**
modbus_state	ModbusState[]		모드버스 상태 : ModbusState.msg 참조
access_control	int32		제어권 상태: enum.ACCESS_CONTROL 참조

인수명	자료형	기본값	설명
homming_completed	bool		로봇의 Homming 여부
tp_initialized	bool		TP 초기화 상태
mastering_need	int8		로봇 마스터링z 필요 여부
drl_stopped	bool		DRL(Doosan Robot Language)정지 상태
disconnected	bool		통신 접속 상태

enum.ROBOT_STATE

순번	상수명	설명
0	STATE_INITIALIZING	T/P 어플리케이션에 의해서 자동으로 진입하는 상태로 각종 파라미터 설정을 위한 초기화 상태임.
1	STATE_STANDBY	운용 가능한 기본 상태 지령 대기 상태임
2	STATE_MOVING	지령 대기 상태에서 지령 수신 후 동작시 자동으로 전환되는 지령 동작 상태임. 동작 완료시 자동 지령 대기 상태로 전환됨.
3	STATE_SAFE_OFF	기능 및 동작 오류로 인한 로봇 정지 모드로, 서보 오프 상태(제어 정지 후 모터 및 브레이크 전원을 차단한 상태)임
4	STATE_TEACHING	직접교시 상태
5	STATE_SAFE_STOP	기능 및 동작 오류로 인한 로봇 정지 모드로, 안전 정지 상태(제어 정지만 수행한 상태, 자동모드인 경우 프로그램 일시 정지 상태)
6	STATE_EMERGENCY_STOP:	비상 정지 상태
7	STATE_HOMMING	홈밍 모드 상태(로봇을 하드웨어적으로 정렬하는 상태).
8	STATE_RECOVERY	로봇 구동 범위 초과 등과 같은 오류로 인한 로봇 정지시, 구동 범위 이내로 이동시키기 위한 복구 모드 상태임.
9	eSTATE_SAFE_STOP2	eSTATE_SAFE_STOP 상태와 동일하나, 로봇 구동 범위 초과로 인해 복구 모드로 전환해야 하는 상태
10	STATE_SAFE_OFF2	eSTATE_SAFE_OFF 상태와 동일하나, 로봇 구동 범위 초과로 인해 복구 모드로 전환해야 하는 상태

순번	상수명	설명
11	STATE_RESERVED1	예약 사용
12	STATE_RESERVED2	예약 사용

enum.CONTROL_MODE

순번	상수명	설명
0	CONTROL_MODE_POSITION	위치 제어 모드
1	CONTROL_MODE_TORQUE	토크 제어 모드

enum.ROBOT_STATE

순번	상수명	설명
0	ROBOT_SPACE_JOINT	관절 공간
1	ROBOT_SPACE_TASK	작업 공간

enum.ACCESS_CONTROL

순번	상수명	설명
0	MANAGE_ACCESS_CONTROL_FORCE_REQUEEST	제어권 강제 회수 메시지 송신
1	MANAGE_ACCESS_CONTROL_REQUEST,	제어권 이양 요청 메시지 송신
2	MANAGE_ACCESS_CONTROL_RESPONSE_YES	제어권 이양 승락 메시지 송신
3	MANAGE_ACCESS_CONTROL_RESPONSE_NO	제어권 이양 거절 메시지 송신

7.1.2 RobotStop.msg

- **기능**

로봇 정지 토픽 메시지

- **인수**

인수명	자료형	기본값	설명
stop_mode	int32		robot stop mode : enum.STOP _MODE 참조.

enum.STOP_MODE

순번	상수명	설명
0	DR_QSTOP_STO	내부 예약 사용
1	DR_QSTOP	빠른 정지(모션 궤적 유지)
2	DR_SSTOP	느린 정지(모션 궤적 유지)
3	DR_HOLD	긴급 정지

7.1.3 RobotError.msg

- **기능**

로봇 로그 및 알람 토픽 메시지

- **인수**

인수명	자료형	기본값	설명
Level	int32	-	로그 레벨 : enum.LOG _LEVEL 참조.
Group	int32	-	로그 그룹 : enum.LOG _GROUP 참조.
Code	int32	-	error code
msg1	string	-	error msg 1
msg2	string	-	error msg 2
msg3	string	-	error msg 3

enum.LOG_LEVEL

순번	상수명	설명
0	LOG_LEVEL_RESERVED	내부 예약 상태
1	LOG_LEVEL_SYSINFO	단순 기능 및 동작 오류에 대한 정보용 메시지
2	LOG_LEVEL_SYSWARN	단순 기능 및 동작 오류로 인한 로봇이 정지된 상태
3	LOG_LEVEL_SYSERROR	안전 이슈나 장치 오류로 인한 로봇이 정지된 상태

enum.LOG_GROUP

순번	상수명	설명
0	LOG_GROUP_RESERVED	
1	LOG_GROUP_SYSTEMFMK	하위 제어기(프레임워크)
2	eLOG_GROUP_MOTIONLIB,	하위 제어기(알고리즘)
3	LOG_GROUP_SMARTTP	상위 제어기 프로그램(GUI)

순번	상수명	설명
4	LOG_GROUP_INVERTER	로봇 인버터 보드
5	LOG_GROUP_SAFETYCONTROLLER	안전 보드(Safety Controller)

7.1.4 LogAlarm.msg

- **기능**

로봇 로그 및 알람 토픽 메시지

- **인수**

인수명	자료형	기본값	설명
level	int32	-	로그 레벨 : enum.LOG _LEVEL 참조.
group	int32	-	로그 그룹 : enum.LOG _GROUP 참조.
index	int32	-	error code
param	string[3]	-	로그 메시지[3]

enum.LOG_LEVEL

순번	상수명	설명
0	LOG_LEVEL_RESERVED	내부 예약 상태
1	LOG_LEVEL_SYSINFO	단순 기능 및 동작 오류에 대한 정보용 메시지
2	LOG_LEVEL_SYSWARN	단순 기능 및 동작 오류로 인한 로봇이 정지된 상태
3	LOG_LEVEL_SYSERROR	안전 이슈나 장치 오류로 인한 로봇이 정지된 상태

enum.LOG_GROUP

순번	상수명	설명
0	LOG_GROUP_RESERVED	
1	LOG_GROUP_SYSTEMFMK	하위 제어기(프레임워크)
2	eLOG_GROUP_MOTIONLIB,	하위 제어기(알고리즘)
3	LOG_GROUP_SMARTTP	상위 제어기 프로그램(GUI)
4	LOG_GROUP_INVERTER	로봇 인버터 보드
5	LOG_GROUP_SAFETYCONTROLLER	안전 보드(Safety Controller)

7.1.5 ModbusState.msg

- **기능**

  모드버스 상태 토픽 메시지

- **인수**

인수명	자료형	기본값	설명
level	string	-	Modbus Signal Name.
group	int32	-	Modbus Register Value (Unsigned : 0 ~ 65535)

7.1.6 JogMultiAxis.msg

- **기능**

다축 조그 제어 토픽 메시지

다축 조그 속도 = (250mm/s)/√3 x 단위벡터 크기 x speed[%]

- **인수**

인수명	자료형	기본값	설명
jog_axis	float64[6]	-	Task space [Tx, Ty, Tz, Rx, Ry, Rz] 의 단위 백터 방향 (-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)

7.2 Service/motion

7.2.1 Trans.srv

- **기능**

- ref 좌표계 기준으로 정의된 pos(포즈)를 동일 좌표계를 기준으로 delta만큼 이동/회전 하여 ref_out 좌표계 기준으로 변환하기 위한 서비스입니다.
- 단, ref 인자가 Tool Coordinate인 경우, ref_out인자는 무시되며 pos와 동일한 좌표계 기준의 결과를 반환합니다.

- **인수**

인수명	자료형	기본값	설명
pos	float64[6]	-	6개의 Task Space 정보
delta	float64[6]	-	6개의 Task Space 정보
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

알아두기

· ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.
· ref_out 인자는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
trans_pos	float64[6]	-	6개의 Task Space 정보

7.2.2 Fkin.srv

- **기능**

조인트 공간에서 조인트각도 또는 이에 상응하는 자료형(float[6])을 입력받아 ref 좌표계 기준의 TCP의 포즈(태스크공간의 위치 및 자세)를 계산하기 위한 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
pos	float64[6]	-	6개의 Joint Space 정보
ref	int8	0	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_WORLD=2

알아두기

· ref 인자는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
conv_posx	float64[6]	-	6개의 Task Space 정보

7.2.3 Ikin.srv

- **기능**

작업 공간내 기준 좌표계(ref)의 로봇 포즈에 상응하는 8개의 관절형상 중 지정한 관절형상(sol_space)에 해당하는 관절각도 계산하기 위한 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
pos	float64[6]	-	6개의 Task Space 정보
sol_space	int	-	solution space
ref	int	0	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_WORLD=2

알아두기

· ref 인자는 M2.40 이상의 버전에서만 사용 가능합니다.

- **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

- **리턴**

인수명	자료형	기본값	설명
conv_posj	float64[6]	-	6개의 Joint Space 정보

7.2.4 SetRefCoord.srv

- **기능**

기준 좌표계를 설정하기 위한 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
coord	int	-	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_TOOL=1 · MOVE_REFERENCE_WORLD=2 · MOVE_REFERENCE_USER=101~120

알아두기

· ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.5 MoveJoint.srv

- **기능**

로봇제어기에서 로봇을 현재 관절위치에서 목표 관절위치까지 이동시키기 위한 서비스 입니다.

- **인수**

인수명	자료형	기본값	설명
pos	float64[6]	-	6개 축에 대한 목표 관절 위치
vel	float64	-	속도
acc	float64	-	가속도
time	float64	0.0	도달 시간 [sec]
radius	float64	0.0	blending시 radius
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

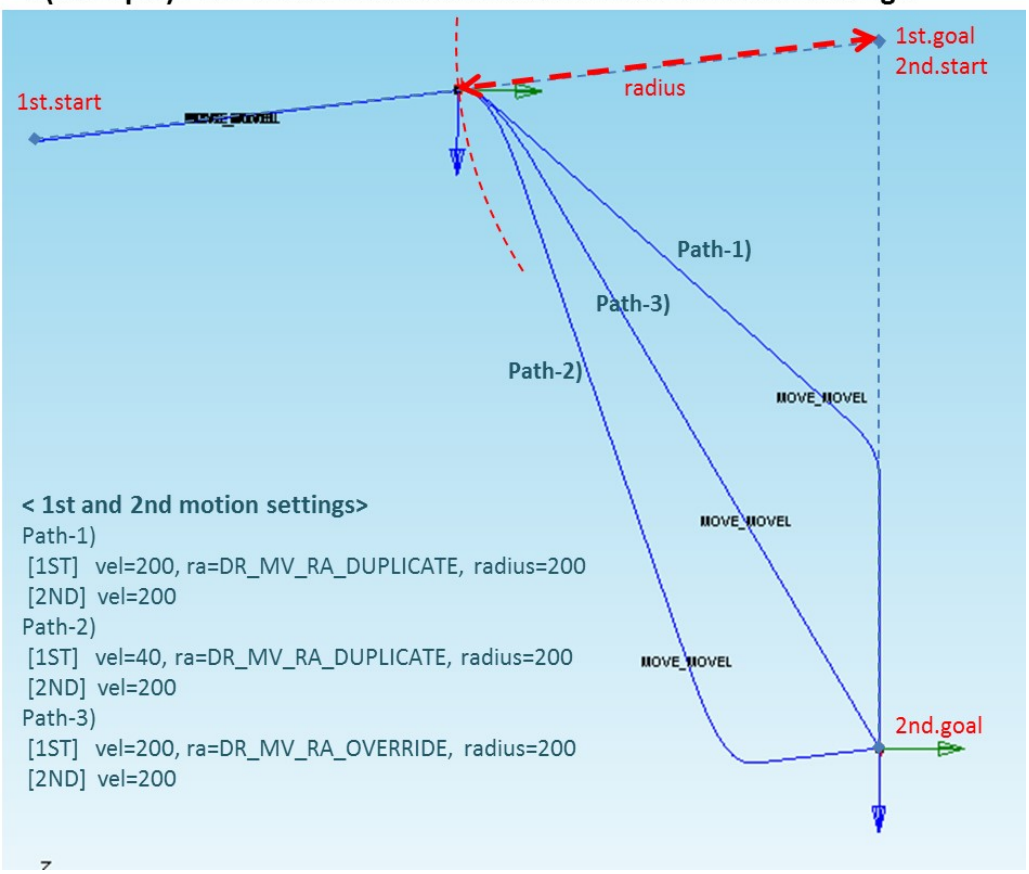
알아두기

· time 지정 시, vel, acc 를 무시하고 time 기준으로 처리됩니다.

주의

blendType 이 BLENDING_SPEED_TYPE_DUPLICATE 이고 radius 가 0 보다 큰 조건으로 후속 모션이 블렌딩 될 경우 선행모션의 잔여거리, 속도, 가속도로 결정되는 잔여모션시간이 후행모션의 모션시간보다 큰 경우 후행모션이 먼저 종료된 후 선행모션이 종료될 수 있습니다. 관련한 사항은 아래 이미지를 참고하십시오.

< (Example) Path differences accord. to 1st and 2nd motion settings>



리턴

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.6 MoveLine.srv

- **기능**

로봇제어기에서 로봇을 작업 공간 안에서 목표 위치(pos)로 직선을 따라 이동시키기 위한 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
pos	float64[6]	-	6개 축에 대한 목적 TCP 위치
vel	float64[2]	-	선속도, 각속도
acc	float64[2]	-	선가속도, 각가속도
time	float64	0.0	도달 시간 [sec] * time 지정 시, vel, acc를 무시하고 time 기준으로 처리
radius	float64	0.0	blending시 radius
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

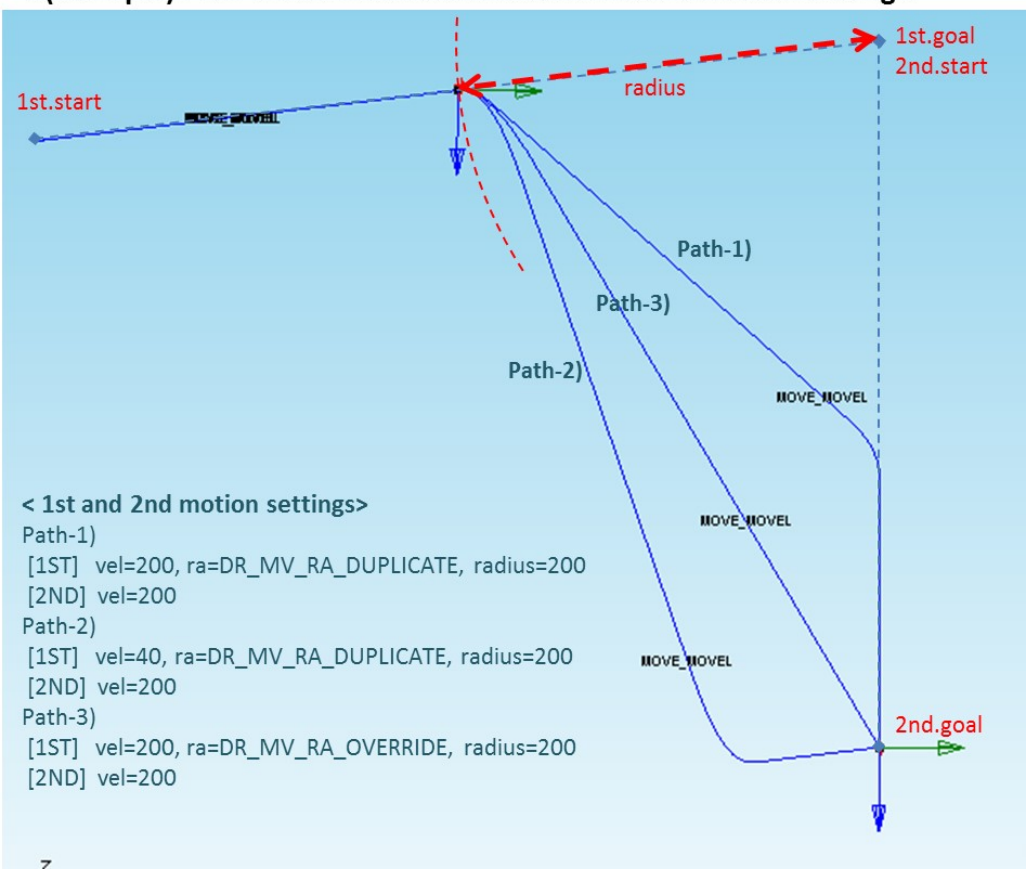
알아두기

- vel 에 하나의 인자를 입력한 경우(예를들어, vel ={30, 0}) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- acc 에 하나의 인자를 입력한 경우(예를들어, acc ={60, 0}) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- time 지정 시, vel, acc 를 무시하고 time 기준으로 처리됩니다.
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**

⚠ 주의

blendType 이 BLENDING_SPEED_TYPE_DUPLICATE 이고 radius 가 0 보다 큰 조건으로 후속 모션이 블렌딩 될 경우 선행모션의 잔여거리, 속도, 가속도로 결정되는 잔여모션시간이 후행모션의 모션시간보다 큰 경우 후행모션이 먼저 종료된 후 선행모션이 종료될 수 있습니다. 관련한 사항은 아래 이미지를 잠고하십시오.

< (Example) Path differences accord. to 1st and 2nd motion settings>



- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.7 MoveJointx.srv

- **기능**

로봇제어기에서 로봇을 관절 공간 안에서 목표 위치로 이동시키기 위한 서비스 입니다. 목표 위치는 작업공간 상의 위치임으로 MoveL과 동일하게 이동하지만 로봇의 모션은 관절공간에서 이루어지기 때문에 목표 위치까지 직선경로가 보장되지 않습니다. 추가적으로 하나의 작업공간좌표에 대응하는 8가지의 관절조합형태(robot configuration)중 하나를 iSolutionSpace(solution space)에 지정해야 합니다.

- **인수**

인수명	자료형	기본값	설명
pos	float64[6]	-	6개 축에 대한 목적 TCP 위치
vel	float64	-	속도
acc	float64	-	가속도
time	float64	0.0	도달 시간 [sec]
radius	float64	0.0	blending시 radius
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	관절조합형태(아래 설명 참조)
syncType	int8	0	• SYNC = 0 • ASYNC = 1

알아두기

- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리된다.
- 상대모션으로 입력하는 경우(eMoveMode = MOVE_MODE_RELATIVE), 선행모션에 블렌딩을 사용하는 경우 에러가 발생하므로 MoveJoint() 또는 MoveLine()을 이용하여 블렌딩하는 것을 권장한다.
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**

- 옵션 blendType 및 vel / acc 에 따른 블렌딩을 수행할 경우 MoveJoint.srv, MoveLine.srv 설명을 참조하십시오.

▪ Robot configuration (형태 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

▪ 리턴

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.8 MoveCircle.srv

- **기능**

로봇 제어기에서 작업공간을 기준으로 로봇이 현재 위치에서 경유 지점를 지나 목표 위치까지 원호 또는 지정한 각도로 원호를 따라 이동시키기 위한 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
pos	std_msgs/Float64MultiArray[]	-	target[2][6] • 경유 지점 • 목표 위치
vel	float64[2]	-	선속도, 각속도
acc	float64[2]	-	선가속도, 각가속도
time	float64	0.0	도달 시간 [sec]
radius	float64	0.0	blending시 radius
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
angle1	float64	0.0	angle1
angle2	float64	0.0	angle2
blendType	int8	0	• BLENDING_SPEED_TYPE_DUPLICATE =0 • BLENDING_SPEED_TYPE_OVERRIDE =1
syncType	int8	0	• SYNC = 0 • ASYNC = 1

알아두기

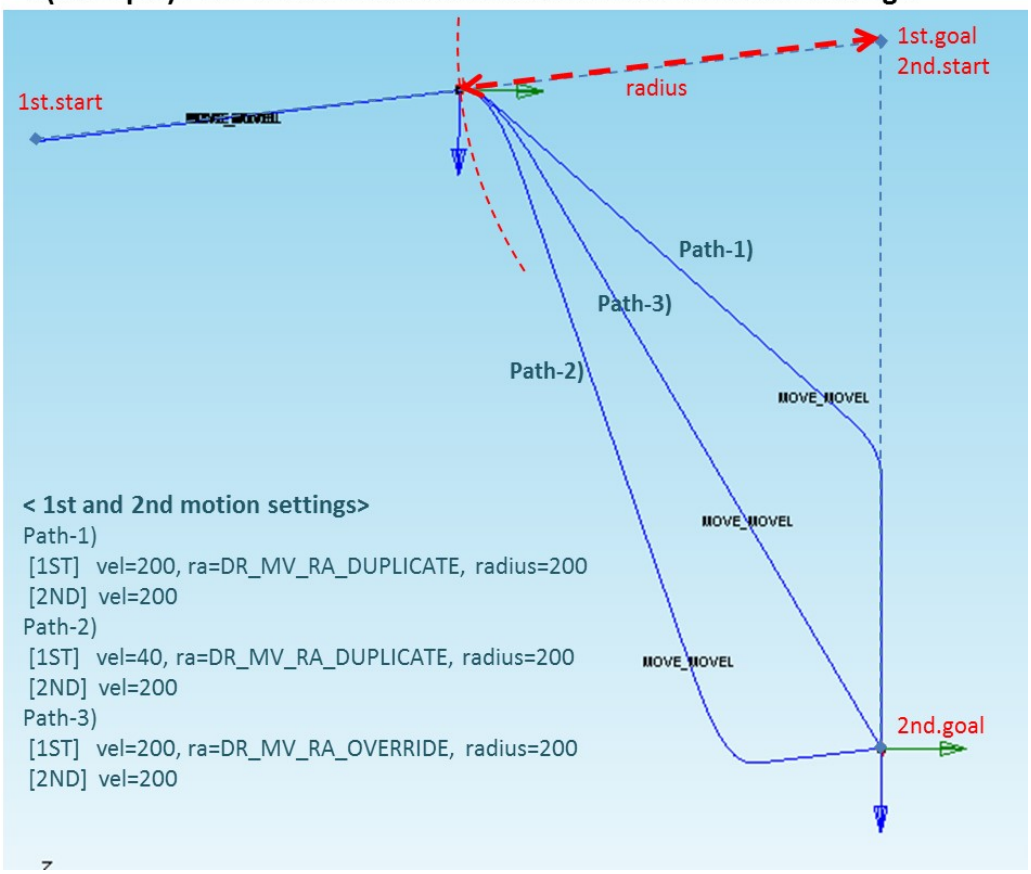
・ vel 에 하나의 인자를 입력한 경우(예를들어, vel ={30, 0}) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.

- acc 에 하나의 인자를 입력한 경우(예를들어, acc ={60, 0}) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- time 지정 시, vel, acc 를 무시하고 time 기준으로 처리됩니다.
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- mode 가 MOVE_MODE_RELATIVE 인 경우 pos[0] 과 pos[1] 는 각각 앞 선 위치값에 대한 상대좌표로 정의됩니다. (pos[0]은 시작점 대비 상대좌표, pos[1]는 pos[0]대비 상대좌표)
- angle1 이 0 보다 크고, angle2 이 0 인 경우 angle1 은 Circular path 상의 총 회전각이 적용됩니다.
- angle1 과 angle2 가 0 보다 큰 경우, angle1 은 circular path 상에서 정속으로 이동하는 총 회전각을, angle2 는 가속과 감속을 위한 회전 구간의 회전각을 의미합니다. 이때 총 이동각은 angle1+ 2 X angle2 만큼 circular path 상을 움직입니다.

⚠ 주의

blendType 이 BLENDING_SPEED_TYPE_DUPLICATE 이고 radius 가 0 보다 큰 조건으로 후속 모션이 블렌딩 될 경우 선행모션의 잔여거리, 속도, 가속도로 결정되는 잔여모션시간이 후행모션의 모션시간보다 큰 경우 후행모션이 먼저 종료된 후 선행모션이 종료될 수 있습니다. 관련한 사항은 아래 이미지를 잠고하십시오.

< (Example) Path differences accord. to 1st and 2nd motion settings>



- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.9 MoveSplineJoint.srv

- **기능**

로봇 제어기에서 로봇을 현재 위치에서 관절공간의 경유점들을 거쳐 목표위치(마지막 경유점)까지 연결되는 스플라인 곡선경로를 따라 이동시키기 위한 서비스 입니다. 입력된 속도/가속도는 경로 중 최대 속도/가속도를 의미하며 입력되는 경유점의 위치에 따라 모션 중의 감속 또는 가속이 결정됩니다.

▪ 인수

인수명	자료형	기본값	설명
pos	std_msgs/Float64MultiArray[]	-	target pos [100][6] 최대 100개까지의 경유점 리스트
posCnt	int8	-	유효 경유점 개수
vel	float64	-	속도
acc	float64	-	가속도
time	float64	0.0	도달 시간 [sec]
mode	int8	0	• MOVE_MODE_ABSOLUTE =0 • MOVE_MODE_RELATIVE =1
syncType	int8	0	• SYNC = 0 • ASYNC = 1

알아두기

- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- mode 가 MOVE_MODE_RELATIVE 인 경우 position list 의 각 pos 는 앞 선 pos 에 대한 상대좌표로 정의됩니다. (position list=[q1, q2, ...,q(n-1), q(n)]로 이루어질 때 q1 은 시작점 대비 상대각도, q(n)은 q(n-1) 대비 상대좌표)
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

▪ 리턴

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.10 MoveSplineTask.srv

▪ 기능

로봇 제어기에서 로봇을 현재 위치에서 작업공간의 경유점들을 거쳐 목표위치(마지막 경유점)까지 연결되는 스플라인 곡선경로를 따라 이동시키기 위한 서비스 입니다. 입력된 속도/가속도는 경로 중 최대 속도/가속도이며 정속모션 옵션을 선택할 경우 조건에 따라 입력한 속도로 정속도의 모션을 수행합니다.

▪ 인수

인수명	자료형	기본값	설명
pos	std_msgs/Float64MultiArray[]	-	target pos [100][6] 최대 100개까지의 경유점 리스트
posCnt	int8	-	유효 경유점 개수
vel	float64[2]	-	선속도, 각속도
acc	float64[2]	-	선가속도, 각가속도
time	float64	0.0	도달 시간 [sec]
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
opt	int8	0	• SPLINE_VELOCITY_OPTION_DEFAULT=0 • SPLINE_VELOCITY_OPTION_CONST=1
syncType	int8	0	• SYNC = 0 • ASYNC = 1

알아두기

- vel 에 하나의 인자를 입력한 경우(예를들어, vel ={30, 0}) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- acc 에 하나의 인자를 입력한 경우(예를들어, acc ={60, 0}) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- time 지정 시, vel, acc 를 무시하고 time 기준으로 처리됩니다.
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- mode 가 MOVE_MODE_RELATIVE 인 경우 position list 의 각 pos 는 앞 선 pos 에 대한 상대좌표로 정의됩니다. (position list=[q1, q2, ...,q(n-1), q(n)]로 이루어질 때 q1 은 시작점 대비 상대각도, q(n)은 q(n-1) 대비 상대좌표)
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

주의

opt 을 SPLINE_VELOCITY_OPTION_CONST 옵션(등속모션)을 선택할 경우 입력된 경유점 간 거리와 속도 조건에 따라 등속모션을 사용할 수 없을 수 있으며, 이 경우에 변속모션 (opt =SPLINE_VELOCITY_OPTION_DEFAULT)으로 자동 전환됩니다.

▪ 리턴

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.11 MoveBlending.srv

▪ 기능

로봇 제어기에서 하나 이상의 라인 또는 원호 구성된 경로 정보을 받아 로봇을 경로 정보에 설정된 blending radius로 블렌딩하여 등속으로 이동시키기 위한 서비스 입니다.

▪ 인수

인수명	자료형	기본값	설명
pos	std_msgs/Float64MultiArray[]	-	(pos1[6]:pos2[6]:type[1]:radius[1]) x 50(max) 최대 50개까지의 경로 정보
posCnt	int8		유효 경로 정보 개수
vel	float64[2]	-	선속도, 각속도
acc	float64[2]	-	선가속도, 각가속도
time	float64	0.0	도달 시간 [sec]
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
syncType	int8	0	• SYNC = 0 • ASYNC = 1

알아두기

- vel 에 하나의 인자를 입력한 경우(예를들어, vel ={30, 0}) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- acc 에 하나의 인자를 입력한 경우(예를들어, acc ={60, 0}) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- time 지정 시, vel, acc 를 무시하고 time 기준으로 처리됩니다
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- mode 가 MOVE_MODE_RELATIVE 인 경우 posb list 의 각 pos 는 앞 선 pos 에 대한 상대좌표로 정의됩니다.

주의

- posb 에서 blending radius 가 0 인 경우, 사용자 입력 오류가 나타납니다.
- 연속된 Line-Line segment 가 같은 방향을 가질 경우 Line 의 중복입력으로 사용자 입력 오류가 나타납니다.
- 블렌딩 구간에서 조건에 따라 급격하게 방향전환이 발생하게 되는 경우 급가속을 방지하기 위해 사용자 입력오류가 나타납니다.
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

▪ 리턴

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.12 MoveSpiral.srv

▪ 기능

로봇제어기에서 방사형 방향으로 반경이 증가하며 회전하는 Spiral motion과 축 방향으로 병행하면서 이동시키기 위한 서비스 입니다. 현재 위치에서 eMoveReference 로 지정한 좌표계 상의 axis 방향에 수직인 평면에서의 나선궤적과 axis 방향으로의 직선궤적을 동시에 따라 이동합니다.

▪ 인수

인수명	자료형	기본값	설명
revolution	float64	-	총 회전수 [revolution]
maxRadius	float64		spiral 최종 반경 [mm]
maxLength	float64		axis 방향으로 이동하는 거리 [mm]
vel	float64[2]	-	선속도, 각속도
acc	float64[2]	-	선가속도, 각가속도
time	float64	0.0	총 수행시간 [sec]
taskAxis	int8	0	• TASK_AXIS_X = 0 • TASK_AXIS_Y = 1 • TASK_AXIS_Z = 2
ref	int8	0	• MOVE_REFERENCE_BASE =0 • MOVE_REFERENCE_TOOL=1 • MOVE_REFERENCE_WORLD=2
syncType	int8	0	• SYNC = 0 • ASYNC = 1

알아두기

- revolution 는 spiral 모션의 총 회전수를 의미합니다.
- maxRadius 는 spiral 모션의 최대 반경을 의미합니다.
- maxLength 는 모션 동안 axis 방향으로 병진하는 거리를 의미합니다. 단, 음수인 경우 –axis 방향 병진합니다.
- vel 은 spiral 모션의 이동 속도를 의미합니다.
- acc 는 spiral 모션의 이동 가속도를 의미합니다.
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- taskAxis 는 Spiral 모션이 정의하는 평면에 수직인 축을 정의합니다.
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**

- ref 는 spiral 모션이 정의하는 기준 좌표계를 의미합니다.
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

⚠ 주의

- 경로 생성 시 Spiral 경로에 의한 회전각 가속도를 연산하여 값이 큰 경우 안정적인 모션을 위하여 에러가 발생나타날 수 있습니다. 이 경우 vel, acc 또는 time 값을 작게 조정하는 것을 권장합니다.

▪ 리턴

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.13 MovePeriodic.srv

- **기능**

로봇제어기에서 현재 위치에서 시작하는 상대 모션으로 입력된 기준 좌표계(eMoveReference)의 각 축(병진 및 회전)에 대한 Sine 함수 기반으로 주기 모션을 수행합니다. 각 axis 별 모션의 특성은 fAmplitude와 fPeriodic 에 의해 결정되고, 가감속 시간과 총 모션 시간은 주기, 반복, 횟수에 의해 설정됩니다.

- **인수**

인수명	자료형	기본값	설명
amp	float64[6]	-	Amplitude(-amp에서 +amp사이 모션) [mm] or [deg]
periodic	float64[6]	-	period(1주기 소요 시간)[sec]
acc	float64	-	Acceleration
time	float64	-	Acc-, dec- time [sec]
repeat	int8	-	반복 횟수
ref	int8	0	• MOVE_REFERENCE_BASE =0 • MOVE_REFERENCE_TOOL=1 • MOVE_REFERENCE_WORLD=2
syncType	int8	0	• SYNC = 0 • ASYNC = 1

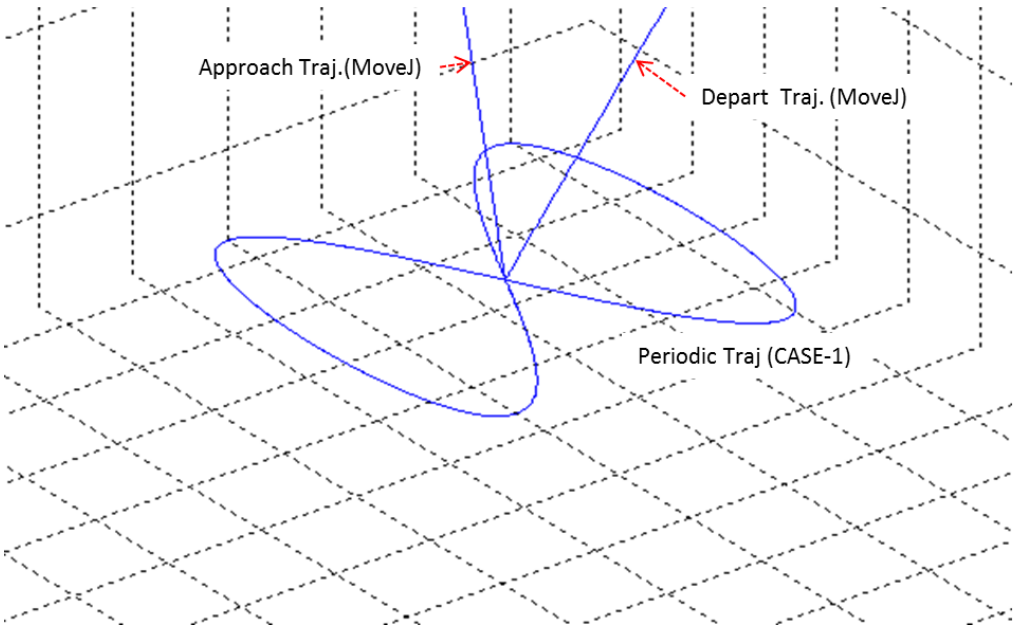
알아두기

- amp 는 진폭(amplitude)을 의미하며, 각 축(x, y, z, rx, ry, rz) 별로 amp 를 값으로 하는 6 개 원소의 list 형태로 입력해야 합니다. 단, 주기 모션을 진행하지 않는 축 방향은 amp 를 0 으로 입력해야 합니다.
- periodic 는 해당 방향 모션의 1 회 반복 시간을 의미하며, 각 축(x, y, z, rx, ry, rz) 별 period 를 값으로 하는 총 6 개 원소의 list 형태로 입력하거나 대표값을 입력해야합니다.
- acc 은 주기모션의 시작과 끝의 가속 및 감속 시간을 의미합니다. 입력된 가감속시간과 최대주기*1/4 중 큰 값이 적용됩니다. 입력된 가감속 시간이 전체모션시간의 1/2 을 초과하는 경우 에러가 발생합니다.
- repeat 은 가장 큰 period 값을 가지는 축(기준 축)의 반복 횟수를 정의하며, 이에 따라 총 모션 시간이 결정됩니다. 나머지 축의 반복 횟수는 모션 시간에 따라 자동결정됩니다.모션이 정상 종료되는 경우 종료 위치가 시작 위치와 일치하게 하도록 나머지 축 모션은 기준 축

모션이 종료되기 전에 먼저 종료될 수 있습니다. 모든 축의 모션이 동시 종료되지 않는 경우 감속구간에서의 경로는 이전 경로에서 벗어나게 됩니다. 관련한 사항은 아래 이미지를 참조하십시오

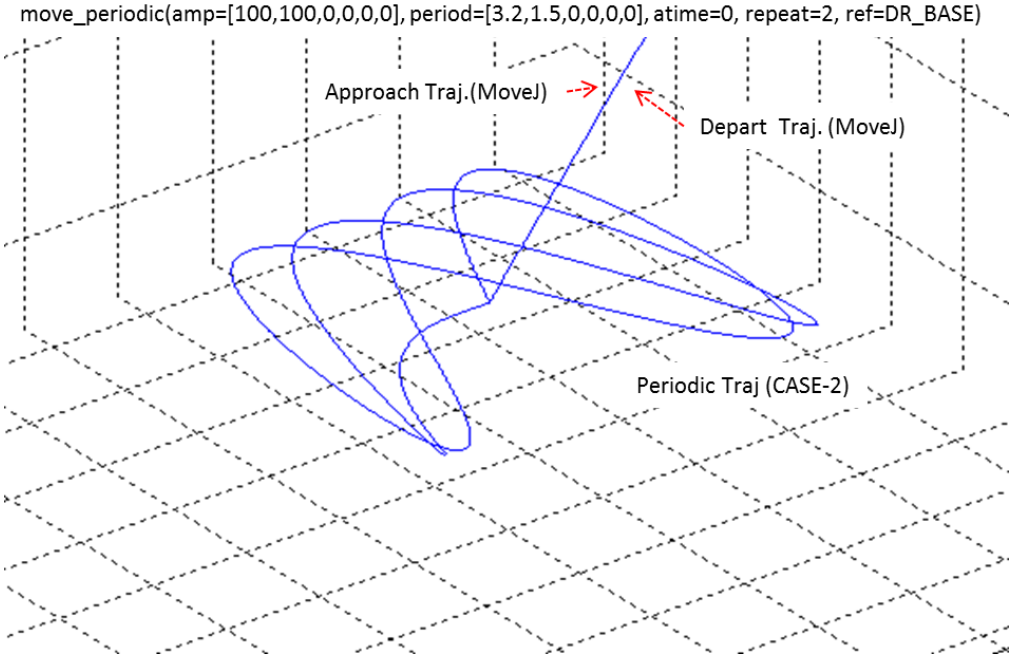
CASE-1) All-axis motions end at the same time

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 는 반복 모션의 기준 좌표계를 의미합니다.
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- 모션명령 수행 시 최대속도 에러가 발생하는 경우 다음의 식을 참조하여 진폭 및 주기를 조정할 것을 제안합니다.
  최대속도=진폭(amp)*2*pi(3.14)/주기(period)
  (예, 진폭=10mm, 주기=1 초인 경우 최대속도=62.83mm/sec)
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

▪ 리턴

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.14 MoveWait.srv

- **기능**

로봇제어기에서 선행된 모션명령어의 동작이 종료되기를 기다리기 위한 서비스 입니다. 비동기 모션 명령어와 본 함수를 결합하여 사용하면 동기 모션 명령어와 동일한 동작을 수행할 수 있습니다.

- **인수**

인수명	자료형	기본값	설명
없음	-	-	-

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.15 MovePause.srv

- **기능**

현재 진행중인 로봇의 모션을 감속시켜 일시중지시키기 위한 서비스입니다.
진행중인 로봇 모션이 없는 경우에는 무시됩니다.

- **인수**

인수명	자료형	기본값	설명
없음	-	-	-

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.16 MoveResume.srv

- **기능**

일시 중지된 로봇의 모션을 재개하기 위한 서비스입니다.
진행중인 로봇 경로모션이 없는 경우에는 무시됩니다.

- **인수**

인수명	자료형	기본값	설명
없음	-	-	-

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.17 MoveStop.srv

- **기능**

로봇제어기에서 수행 중인 모션을 정지시키기 위한 서비스입니다.
인자에 따라 따라 다르게 정지하며, E-stop을 제외한 모든 Stop 모드는 현재 수행하고 있는 구간의 모션을 정지합니다.

- **인수**

인수명	자료형	기본값	설명
Stop_mode	int32	-	0 : reserved 1 : 빠른 정지(모션 궤적 유지) 2 : 느린 정지(모션 궤적 유지)

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.18 Jog.srv

- **기능**

로봇 제어기에서 로봇의 단 축에 대한 조그 제어를 수행하기 위한 서비스 입니다.
단축 조그 속도 = (250mm/s) x speed [%]

- **인수**

인수명	자료형	기본값	설명
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

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.19 multiJog.srv

- **기능**

로봇 제어기에서 로봇의 다 축에 대한 조그 제어를 수행하기 위한 서비스 입니다.
다축 조그 속도 = (250mm/s)/√3 x 단위벡터 크기 x speed[%]

- **인수**

인수명	자료형	기본값	설명
jog_axis	float64[6]	-	Task space [Tx, Ty, Tz, Rx, Ry, Rz] 의 단위 백터 방향 (-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)

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.20 CheckMotion.srv

- **기능**

  현재 진행 중인 모션의 상태를 확인하기 위한 서비스입니다.

- **인수**

  없음

- **리턴**

인수명	자료형	기본값	설명
status	int8	-	0 : 수행 중인 모션이 없음 1 : 모션 연산 중 2 : 모션이 수행 중

7.2.21 ChangeOperationSpeed.srv

- **기능**

작동 속도를 조정하기 위한 서비스입니다.. 인자는 현재 설정된 속도에 대한 상대적인 비율을 백분율로 환산한 값으로 1에서 100까지의 값을 갖습니다. 따라서 50은 현재 설정된 속도의 50 Percent로 속도를 줄인다는 의미입니다.

- **인수**

인수명	자료형	기본값	설명
speed	int8	-	operation speed(1~100)

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.22 EnableAlterMotion.srv

- **기능**

본 서비스는 M2.40 이상의 버전에서만 사용 가능합니다.

경로 수정 기능을 활성화 하기 위한 서비스입니다. 경로 생성의 단위 주기는 100msec 이며 입력 인자 n 을 설정하여 경로 생성 주기(n*100msec)를 변경 할 수 있습니다. 입력 인자 mode 를 통해 alter_motion()의 입력값의 의미를 2 가지 모드(누적량 모드, 증분량 모드) 중 하나로 선택하여 사용 할 수 있습니다. 누적량 모드의 경우 현재의 모션경로에 대한 절대적 증분위치/자세만큼 경로 수정량이 반영 됩니다. 증분량 모드의 경우 바로 현재의 절대적 증분위치/자세에 입력된 증분위치/자세만큼 경로 수정량이 추가되어 반영 됩니다. 입력 인자 ref 를 통해 기준 좌표계를 설정할 수 있습니다. 입력 인자 limit_dPOS, limit_dPOS_per 를 통해 각각 누적량, 증분량의 한계치를 설정 할 수 있습니다. 한계치를 벗어나는 위치 값의 한계치에 수렴한 값으로 경로 수정량이 재 조정 됩니다.

- **인수**

인수명	자료형	기본값	설명
n	int32	-	경로 생성 주기
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]	-	첫번째 값 : 이동량 제한 값[mm] 두번째 값 : 회전량 제한 값[deg]
limit_dPOS_per	float64[2]	-	첫번째 값 : 이동량 제한 값[mm] 두번째 값 : 회전량 제한 값[deg]

알아두기

- alter_motion()은 사용자 thread 내에서만 동작합니다.
- ref 가 None 인 경우, _g_coord 적용(_g_coord 초기값은 DR_BASE 이며, set_ref_coord 명령에 의해 설정 가능)
- limit_dPOS, limit_dPOS_per 를 설정하지 않을 시 누적량, 증분량의 한계를 제한하지 않습니다.

- 리턴

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.23 AlterMotion.srv

- 기능

본 서비스는 M2.40 이상의 버전에서만 사용 가능합니다.

입력 인자 pos에 해당하는 양만큼 경로 수정을 하기 위한 서비스입니다.

주의

- alter_motion()은 사용자 thread 내에서만 동작합니다.

알아두기

- alter_motion()은 enable_alter_motion()을 통해 경로보정기능이 활성화 된 경우에만 유효합니다.
- enable_alter_motion 의 설정 값 limit_dPOS, limit_dPOS_per 에 따라 경로 수정량은 조정될 수 있습니다.
- pos 의 방향값은 Fixed XYZ 로 설정하여야 됩니다.

- 인수

인수명	자료형	기본값	설명
pos	float64[6]	-	position list

- 리턴

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.24 DisableAlterMotion.srv

- **기능**

본 서비스는 M2.40 이상의 버전에서만 사용 가능합니다.

경로 수정 기능을 비활성화 하기 위한 서비스입니다.

- **인수**

없음

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.2.25 SetSingularityHandling.srv

- **기능**

task motion에서 특이점의 영향으로 path deviation이 발생할 경우 대응 정책을 사용자가 선택 할 수 있도록 하기 위한 서비스입니다. mode의 설정은 아래와 같은 설정이 가능 합니다.

- 자동회피 모드(Default) : SINGULARITY_AVOIDANCE_AVOID
- 경로 우선 : SINGULARITY_AVOIDANCE_STOP
- 속도 가변 : SINGULARITY_AVOIDANCE_VEL

기본 설정은 자동회피 모드이며, 이 설정의 경우 특이점으로 인한 불안정성을 감소시키지만 path tracking 정확도가 감소합니다. 경로 우선 설정의 경우 singularity 의 영향으로 불안정성이 발생할 가능성이 있는 경우, 감속 후 warning 메시지를 출력하고 해당 Task를 종료합니다. 속도 가변 설정의 경우 특이점으로 인한 불안정을 감소시키면서 path tracking 정확도를 높입니다. 하지만 특이점 구간에서 TCP 속도 변경이 발생합니다.

- **인수**

인수명	자료형	기본값	설명
mode	int8	0	SINGULARITY_AVOIDANCE_AVOID = 0 SINGULARITY_AVOIDANCE_STOP = 1 SINGULARITY_AVOIDANCE_VEL = 2

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.3 Service/system

7.3.1 GetRobotMode.srv

- **기능**

로봇 제어기의 현재 운용 모드 정보를 확인하기 위한 서비스입이다. 자동모드는 일련의 순서로 구성된 동작(프로그램)을 자동으로 수행하기 위한 모드이며, 수동모드는 조그와 같은 단일 동작을 수행하기 위한 모드입니다.

- **인수**

없음

- **리턴**

인수명	자료형	기본값	설명
robot_mode	int8	-	enum.ROBOT_MODE 정의 참조.

- **enum.ROBOT_MODE**

순번	상수명	설명
0	ROBOT_MODE_MANUAL	수동 모드
1	ROBOT_MODE_AUTONOMOUS	자동 모드
2	ROBOT_MODE_MEASURE	측정 모드(현재는 지원하지 않음)

7.3.2 SetRobotMode.srv

- **기능**

  로봇 제어기의 현재 운용 모드 정보를 설정하기 위한 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
robot_mode	int8	-	enum.ROBOT_MODE 정의 참조.

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

- **enum.ROBOT_MODE**

순번	상수명	설명
0	ROBOT_MODE_MANUAL	수동 모드
1	ROBOT_MODE_AUTONOMOUS	자동 모드
2	ROBOT_MODE_MEASURE	측정 모드(현재는 지원하지 않음)

7.3.3 GetRobotSystem.srv

- **기능**

  로봇 제어기에서 현재 운용 로봇 시스템(가상 로봇, 실제 로봇) 정보를 확인하기 위한 서비스입이다.

- **인수**

  없음

- **리턴**

인수명	자료형	기본값	설명
robot_system	int8	-	enum.ROBOT_SYSTEM 정의 참조.

- **enum.ROBOT_SYSTEM**

순번	상수명	설명
0	ROBOT_SYSTEM_REAL	실제 로봇 시스템
1	ROBOT_SYSTEM_VIRTUAL	가상 로봇 시스템

7.3.4 SetRobotSystem.srv

- **기능**

로봇 제어기에서 현재 운용 로봇 시스템을 설정 및 변경하기 위한 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
robot_system	int8	-	enum.ROBOT_SYSTEM 정의 참조.

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

- **enum.ROBOT_SYSTEM**

순번	상수명	설명
0	ROBOT_SYSTEM_REAL	실제 로봇 시스템
1	ROBOT_SYSTEM_VIRTUAL	가상 로봇 시스템

7.3.5 GetRobotSpeedMode.srv

- **기능**

  로봇 제어기에서 현재 속도 모드(정상 모드, 감속 모드) 정보를 확인하기 위한 서비스 입니다.

- **인수**

  없음

- **리턴**

인수명	자료형	기본값	설명
Speed_mode	int8	-	enum.SPEED_MODE 정의 참조

- **enum.SPEED_MODE**

순번	상수명	설명
0	SPEED_NORMAL_MODE	정상 속도 모드.
1	SPEED_REDUCED_MODE	감속 속도 모드

7.3.6 SetRobotSpeedMode.srv

- **기능**

  로봇 제어기에서 현재 운용 중인 속도 모드를 설정 및 변경하기 위한 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
speed_mode	int8	-	enum.SPEED_MODE 정의 참조.

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

- **enum.SPEED_MODE**

순번	상수명	설명
0	SPEED_NORMAL_MODE	정상 속도 모드.
1	SPEED_REDUCED_MODE	감속 속도 모드

7.3.7 SetSafeStopResetType.srv

- **기능**

로봇 제어기의 운용 상태 정보가 SAFE_STOP 일 경우, SetRobotControl 함수를 이용하여 상태 전환 후 이후 자동으로 실행되는 일련의 동작을 정의하기 위한 서비스입니다. 로봇 운용 모드가 자동일 경우, 프로그램의 재실행 여부를 정의 및 설정할 수 있으며, 수동모드일 경우에는 이 설정은 무시됩니다.

- **인수**

인수명	자료형	기본값	설명
reset_type	int8	-	enum.SAFE_STOP_RESET_TYPE 정의 참조.

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

- **enum.SAFE_STOP_RESET_TYPE**

순번	상수명	설명
0	SAFE_STOP_RESET_TYPE_DEFAULT	단순 상태 해제(수동모드)
	SAFE_STOP_RESET_TYPE_PROGRAM_STOP	프로그램 종료(자동모드)
1	SAFE_STOP_RESET_TYPE_PROGRAM_RESUME	프로그램 재시작(자동모드)

7.3.8 GetCurrentPose.srv

- **기능**

로봇 제어기에서 좌표계(관절 공간 또는 작업공간)에 따른 로봇의 각 축별 현재 위치 정보를 확인하기 위한 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
space_type	int8	-	enum.ROBOT_SPACE 정의 참조.

- **리턴**

인수명	자료형	기본값	설명
pos	float64[6]	-	위치 정보

- **enum.ROBOT_SPACE**

순번	상수명	설명
0	ROBOT_SPACE_JOINT	관절 공간
1	ROBOT_SPACE_TASK	작업 공간

7.3.9 GetLastAlarm.srv

- **기능**

로봇 제어기에서 가장 최근에 발생한 로그 및 알람 코드를 확인하기 위한 서비스입니다.

- **인수**

없음

- **리턴**

인수명	자료형	기본값	설명
log_alarm	LogAlarm.msg	-	LogAlam.msg 정의 참조.

- **LogAlam.msg**

인수명	자료형	기본값	설명
level	int32	-	로그 레벨 : enum.LOG _LEVEL 참조.
group	int32	-	로그 그룹 : enum.LOG _GROUP 참조.
index	int32	-	error code
param	string[3]	-	param[3]

enum.LOG_LEVEL

순번	상수명	설명
0	LOG_LEVEL_RESERVED	내부 예약 상태
1	LOG_LEVEL_SYSINFO	단순 기능 및 동작 오류에 대한 정보용 메시지
2	LOG_LEVEL_SYSWARN	단순 기능 및 동작 오류로 인한 로봇이 정지된 상태
3	LOG_LEVEL_SYSERROR	안전 이슈나 장치 오류로 인한 로봇이 정지된 상태

enum.LOG_GROUP

순번	상수명	설명
0	LOG_GROUP_RESERVED	
1	LOG_GROUP_SYSTEMFMK	하위 제어기(프레임워크)
2	eLOG_GROUP_MOTIONLIB,	하위 제어기(알고리즘)
3	LOG_GROUP_SMARTTP	상위 제어기 프로그램(GUI)
4	LOG_GROUP_INVERTER	로봇 인버터 보드
5	LOG_GROUP_SAFETYCONTROLLER	안전 보드(Safety Controller)

로그 및 알람 메시지는 사전에 정의된 내용을 번호를 통해서 전달하며, 필요 시 관련 파라미터를 함께 송부하며 자세한 설명은 로그 및 알람 정의 부분 참조바랍니다.

7.4 Service/aux_control

7.4.1 GetControlMode.srv

- **기능**

현재 제어 모드를 확인하기 위한 서비스입니다.

- **인수**

해당 사항 없음

- **리턴**

인수명	자료형	기본값	설명
control_mode	int8	-	로봇 제어 모드 3 : Position Control Mode 4 : Torque Control mode

7.4.2 GetControlSpace.srv

- **기능**

현재 제어 공간을 확인하기 위한 서비스입니다.

- **인수**

없음

- **리턴**

인수명	자료형	기본값	설명
space	int8	-	로봇 제어 모드 1 : Joint space control 2 : Task space control

7.4.3 GetCurrentPosj.srv

- **기능**

현재 관절 각도를 확인하기 위한 서비스 입니다.

- **인수**

없음

- **리턴**

인수명	자료형	기본값	설명
pos	float64[6]	-	6개의 Joint Space 정보

7.4.4 GetCurrentVelj.srv

- **기능**

현재 관절 속도를 확인하기 위한 서비스입니다.

- **인수**

없음

- **리턴**

인수명	자료형	기본값	설명
joint_speed	float64[6]	-	Joint Speed

7.4.5 GetDesiredPosj.srv

- **기능**

  현재의 목표(target) 관절각을 확인하기 위한 서비스입니다. 단, movel, movec, movesx, moveb, move_spiral, move_periodic 명령어에서는 사용할 수 없습니다.

- **인수**

  없음

- **리턴**

인수명	자료형	기본값	설명
pos	float64[6]	-	관절각

7.4.6 GetDesiredVelj.srv

- **기능**

  현재의 목표(target) 관절각을 리턴합니다. 단, movel, movec, movesx, moveb, move_spiral, move_periodic 명령어에서는 사용할 수 없습니다.

- **인수**

  해당 사항 없음

- **리턴**

인수명	자료형	기본값	설명
joint_vel	float64[6]	-	target joint velocity

7.4.7 GetCurrentPosx.srv

- **기능**

현재 태스크 좌표계의 자세와 solution space를 확인하기 위한 서비스입니다.. 이때 자세는 ref coordinate 를 기준으로 합니다.

- **인수**

인수명	자료형	기본값	설명
ref	Int8	0	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_WORLD=2 · MOVE_REFERENCE_USER=101~120

알아두기

· ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
task_pos_info	std_msgs/Float64MultiArray[]	-	posx list : task_pos_info[0][0:5] solution space : task_pos_info[0][6]

7.4.8 GetCurruntToolFlangePosx.srv

- **기능**

입력된 기준좌표계(ref)에 해당하는 현재 툴 플랜지 포즈를 확인하기 위한 서비스입니다. 즉, tcp=(0,0,0,0,0,0)인 위치가 반환되는 것을 의미합니다.

- **인수**

인수명	자료형	기본값	설명
ref	Int8	0	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_WORLD=2

알아두기

· ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
pos	float64[6]	-	6개의 Tool Flange 좌표 정보

7.4.9 GetCurrentVelx.srv

- **기능**

입력된 기준좌표계(ref)에 해당하는 현재 툴 속도를 확인하기 위한 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
ref	Int8	0	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_WORLD=2

알아두기

· ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
vel	float64[6]	-	tool velocity

7.4.10 GetDesiredPosx.srv

- **기능**

현재의 툴의 목표(target) 자세를 확인하기 위한 서비스입니다. 이때 자세는 ref coordinate 기준으로 합니다.

- **인수**

인수명	자료형	기본값	설명
ref	Int8	0	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_WORLD=2 · MOVE_REFERENCE_USER=101~120

알아두기

· ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
pos	float64[6]	-	target tool position

7.4.11 GetDesiredVelx.srv

- **기능**

입력된 기준좌표계(ref)에 해당하는 현재 툴의 목표(target) 속도를 확인하기 위한 서비스 입니다. 단, movej, movejx, movesj 명령어에서는 사용할 수 없습니다.

- **인수**

인수명	자료형	기본값	설명
ref	Int8	0	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_WORLD=2

알아두기

· ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
vel	float64[6]	-	tool velocity

7.4.12 GetCurrentSiolutionSpace.srv

- **기능**

로봇 제어기에서 로봇의 자세 정보를 확인하기 위한 서비스입니다.

- **인수**

없음

- **리턴**

인수명	자료형	기본값	설명
solution_space	int8	-	로봇 자세 정보(0~7)

- **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

7.4.13 GetCurrentRotm.srv

- **기능**

입력된 기준좌표계(ref)에 해당하는 현재 툴의 회전행렬을 확인하기 위한 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
ref	Int8	0	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_WORLD=2

알아두기

· ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
rot_matrix	std_msgs/Float64MultiArray[]	-	Rotation Matrix

7.4.14 GetJointTorque.srv

- **기능**

현재 조인트의 센서 토크 값을 읽기 위한 서비스 입니다.

- **인수**

없음

- **리턴**

인수명	자료형	기본값	설명
joint_torque	float64[6]	-	JTS 토크값

7.4.15 GetExternalTorque.srv

- **기능**

현재 각 관절에서 외력에 의해 발생하는 토크 값을 확인하기 위한 서비스입니다.

- **인수**

해당 사항 없음

- **리턴**

인수명	자료형	기본값	설명
ext_torque	float64[6]	-	각 축의 External Torque 정보

7.4.16 GetToolForce.srv

- **기능**

현재 툴에 작용하는 외력 값을 읽기 위한 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
ref	Int8	0	・ MOVE_REFERENCE_BASE =0 ・ MOVE_REFERENCE_WORLD=2

알아두기

・ ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
tool_force	float64[6]	-	Tool에 작용하는 외력

7.4.17 GetSolutionSpace.srv

- **기능**

Solution space 값을 구하기 위한 함수입니다.

- **인수**

인수명	자료형	기본값	설명
pos	float64[6]	-	6개의 Task Space 정보

- **리턴**

인수명	자료형	기본값	설명
sol_space	int8	-	solution space : 0 ~ 7

7.4.18 GetOrientationError.srv

- **기능**

axis에 대한 임의의 pose xd와 xc 사이의 Orientation error 값을 확인하기 위한 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
xd	float64[6]	-	6개의 Task Space 정보
xc	float64[6]	-	6개의 Task Space 정보
axis	int8	-	axis · DR_AXIS_X: x축 · DR_AXIS_Y: y축 · DR_AXIS_Z: z축

- **리턴**

인수명	자료형	기본값	설명
ori_error	float32	-	Orientaion error 값

7.5 Service/tcp

7.5.1 ConfigCreateTcp.srv

- **기능**

로봇 TCP 정보를 안전상 사전에 등록하여 사용하기 위한 서비스 입니다, 본 서비스를 이용하여 등록된 TCP 정보는 메모리에 저장됨으로 재부팅 후 다시 설정해야 하지만, T/P 어플리케이션에서 등록한 경우에는 초기화 과정에서 추가됨으로 재사용이 가능합니다.

- **인수**

인수명	자료형	기본값	설명
name	string	-	TCP 이름
pos	float64[6]	-	TCP 정보

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.5.2 ConfigDeleteTcp.srv

- **기능**

로봇 제어기에 사전에 등록된 TCP 정보를 삭제하기 위한 서비스 입니다

- **인수**

인수명	자료형	기본값	설명
name	string	-	TCP 이름

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.5.3 GetCurrentTcp.srv

- **기능**

로봇 제어기에서 현재 설정된 TCP 정보를 가져오는 서비스 입니다. 설정된 Tool 정보가 없을 경우, 빈 문자열이 반환됩니다.

- **인수**

인수명	자료형	기본값	설명
없음	-	-	-

- **리턴**

인수명	자료형	기본값	설명
info	string	-	TCP 이름

7.5.4 SetCurrentTcp.srv

- **기능**

로봇 제어기에 사전에 등록되어 있는 TCP 정보 중 현재 장착된 TCP에 대한 정보를 설정하는 서비스 입니다. 현재 장착된 TCP가 없을 경우, 빈 문자열을 전달하면 현재 설정되어 있는 정보가 초기화됩니다.

인수

인수명	자료형	기본값	설명
name	string	-	Tool 이름

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.5.5 SetToolShape.srv

- **기능**

본 서비스는 M2.50 이상의 버전에서만 사용 가능합니다.

티치팬던트에 등록된 툴 형상 정보 중에서 입력된 name의 tool 형상을 활성화 합니다.

인수

인수명	자료형	기본값	설명
name	string	-	티치 팬던트에 등록된 툴 shape 이름

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.6 Service/tool

7.6.1 ConfigCreateTool.srv

- **기능**

로봇 끝단에 장착될 Tool 정보를 안전상 사전에 등록하여 사용하기 위한 서비스 입니다, 본 서비스를 이용하여 등록된 Tool 정보는 메모리에 저장됨으로 재부팅 후 다시 설정해야 하지만, T/P 어플리케이션에서 등록한 경우에는 초기화 과정에서 추가됨으로 재사용이 가능합니다.

- **인수**

인수명	자료형	기본값	설명
name	string	-	Tool 이름
weight	float		Tool 무게
cog	float64[3]		무게 중심
inertia	float64[6]		관성 정보

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.6.2 ConfigDeleteTool.srv

- **기능**

로봇 제어기에 사전에 등록된 Tool 정보를 삭제하기 위한 서비스 입니다

- **인수**

인수명	자료형	기본값	설명
name	string	-	Tool 이름

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.6.3 GetCurrentTool.srv

- **기능**

로봇 제어기에서 현재 설정된 Tool 정보를 가져오는 서비스 입니다. 설정된 Tool 정보가 없을 경우, 빈 문자열이 반환됩니다.

- **인수**

인수명	자료형	기본값	설명
없음	-	-	-

- **리턴**

인수명	자료형	기본값	설명
info	string	-	Tool 이름

7.6.4 SetCurrentTool.srv

- **기능**

로봇 제어기에 사전에 등록되어 있는 Tool 정보 중 현재 장착된 Tool에 대한 정보를 설정하는 서비스 입니다. 현재 장착된 Tool 이 없을 경우, 빈 문자열을 전달하면 현재 설정되어 있는 정보가 초기화 됩니다.

- **인수**

인수명	자료형	기본값	설명
name	string	-	Tool 이름

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.6.5 SetToolShape.srv

- **기능**

본 서비스는 M2.40 이상의 버전에서만 사용 가능합니다.

티치팬던트에 등록된 툴 형상 정보 중에서 입력된 name의 tool 형상을 활성화 하는 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
name	string	-	티치 팬던트에 등록된 툴 shape 이름

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.7 Service/force

7.7.1 ParallelAxis1.srv

- **기능**

입력된 기준좌표계(ref) 기준의 3개의 포즈(x1,x2,x3)가 이루는 평면의 normal vector(get_normal(x1, x2, x3) 참조)방향에 Tool좌표계의 지정축(axis)의 방향을 일치 시키기 위한 서비스입니다. 이때 로봇 TCP 위치는 현재 위치를 유지합니다.

- **인수**

인수명	자료형	기본값	설명
x1	float64[6]	-	6개의 Task Space 정보
x2	float64[6]	-	6개의 Task Space 정보
x3	float64[6]	-	6개의 Task Space 정보
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

알아두기

· ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
res	int8	-	성공 여부 0 : 성공 기타 : 실패

7.7.2 ParallelAxis2.srv

- **기능**

입력된 기준좌표계(ref) 기준의 벡터(vect) 방향에 Tool좌표계의 지정축(axis)의 방향을 일치시키기 위한 서비스입니다. 이때 로봇 TCP 위치는 현재 위치를 유지합니다.

- **인수**

인수명	자료형	기본값	설명
vect	float64[3]	-	vector
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

알아두기

・ ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
res	int8	-	성공 여부 0 : 성공 기타 : 실패

7.7.3 AlignAxis1.srv

- **기능**

입력된 기준좌표계(ref) 기준의 3개의 포즈(x1,x2,x3)가 이루는 평면의 normal vector(get_normal(x1, x2, x3) 참조)방향에 Tool좌표계의 지정축(axis)의 방향을 일치시키기 위한 서비스입니다. 이때 로봇 TCP 위치는 pos 위치로 이동합니다.

- **인수**

인수명	자료형	기본값	설명
x1	float64[6]	-	6개의 Task Space 정보
x2	float64[6]	-	6개의 Task Space 정보
x3	float64[6]	-	6개의 Task Space 정보
pos	float64[6]	-	6개의 Task Space 정보
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

알아두기

・ ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
res	int8	-	성공 여부 0 : 성공 기타 : 실패

7.7.4 AlignAxis2.srv

- **기능**

입력된 기준좌표계(ref) 기준의 벡터(vect) 방향에 Tool좌표계의 지정축(axis)의 방향을 일치시키기 위한 서비스입니다. 이때 로봇 TCP 위치는 pos 위치로 이동시킵니다.

- **인수**

인수명	자료형	기본값	설명
vect	float64[3]	-	vector
pos	float64[6]	-	6개의 Task Space 정보
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

알아두기

・ ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
res	int8	-	성공 여부 0 : 성공 기타 : 실패

7.7.5 IsDoneBoltTightening.srv

- **기능**

  툴의 조임 토크를 모니터링하기 위한 서비스입니다. 주어진 시간 내에 설정된 토크(m)에 도달한 경우는 True를 리턴하고, 주어진 시간을 초과한 경우에는 False를 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
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**

인수명	자료형	기본값	설명
res	int8	-	성공 여부 0 : 성공 기타 : 실패

7.7.6 ReleaseComplianceCtrl.srv

- **기능**

Compliance control을 종료하고 현재 위치에서 위치 제어를 시작하는 서비스입니다.

- **인수**

없음

- **리턴**

인수명	자료형	기본값	설명
res	int8	-	성공 여부 0 : 성공 기타 : 실패

7.7.7 TaskComplianceCtrl.srv

- **기능**

기존에 설정한 기준 좌표계를 기준으로 태스크 Compliance control을 시작하는 서비스입니다.

- **인수(강성값 TBD)**

인수명	자료형	기본값	설명
stx	float64[6]	[3000, 3000, 3000, 200, 200, 200]	Translational 강성 3개, 회전강성 3개
ref	int8	1	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_TOOL=1 · MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120
time	float	0	강성변화 시간 [sec] 범위 0~1.0 * 주어진 시간 동안 linear transition

- **리턴**

인수명	자료형	기본값	설명
res	int8	-	성공 여부 0 : 성공 기타 : 실패

7.7.8 SetStiffnessx.srv

- **기능**

전역으로 설정된 좌표계(set_ref_coord() 참조) 기준으로 강성값을 설정합니다. 현재 강성 또는 기본값으로부터 STX로 주어진 time값 동안 linear transition 합니다. Translation 강성의 사용자 범위는 0~20000N/m, Rotational 강성의 사용자 범위는 0~400Nm/rad입니다.

- **인수**

인수명	자료형	기본값	설명
stx	float64[6]	[500, 500, 500, 100, 100, 100]	Translational 강성3개, 회전강성 3개
ref	int8	1	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_TOOL=1 · MOVE_REFERENCE_WORLD=2 MOVE_REFERENCE_USER=101~120
time	float	0	강성변화 시간 [sec] 범위 0~1.0 * 주어진 시간 동안 linear transition

- **리턴**

인수명	자료형	기본값	설명
res	int8	-	성공 여부 0 : 성공 기타 : 실패

7.7.9 CalcCoord.srv

- 기능

본 서비스는 M2.50 이상의 버전에서만 사용 가능합니다.

지정한 좌표계(ref) 기준의 최대 4개의 입력점(x1~x4) 및 입력 모드(mod)를 기반으로 새로운 직교 좌표계를 계산하기 위한 서비스입니다. 여기서 입력 모드는 입력점의 개수가 2개인 경우에만 유효합니다.
입력점의 개수가 1개인 경우, x1의 위치와 회전으로 좌표계가 계산됩니다.
입력점의 개수가 2개인 경우 입력모드가 0일 때, x1에서 x2로 향하는 벡터가 x방향, x축에 직교하는 평면에 투영된 현재의 Tool-z방향이 사용자 좌표계의 z방향으로 정의되며 x1의 위치가 원점이 되도록 좌표계가 계산됩니다.
입력점의 개수가 2개인 경우 입력모드가 1일 때, x1에서 x2로 향하는 벡터가 x방향, x축에 직교하는 평면에 투영된 x1의 z축방향이 사용자 좌표계의 z방향으로 정의되며 x1의 위치가 원점이 되도록 좌표계가 계산됩니다.
입력점의 개수가 3개인 경우, x1에서 x2로 향하는 벡터가 x방향으로 정의되며, x1에서 x3으로 향하는 벡터를 v라고 하였을 경우 z방향은 오른손법칙에 따라 x방향 벡터 곱하기 v로 정의되며, x1의 위치가 원점이 되도록 좌표계가 계산됩니다.
입력점의 개수가 4개인 경우, 입력점의 개수가 3개인 경우와 축의 방향은 동일하며 원점의 위치가 x4의 위치가 되도록 좌표계가 계산됩니다.

- 인수

인수명	자료형	기본값	설명
input_pos_cnt	int8		입력 posx의 개수
x1	float64[6]	-	6개의 Task Space 정보
x2	float64[6]	-	6개의 Task Space 정보
x3	float64[6]	-	6개의 Task Space 정보
x4	float64[6]	-	6개의 Task Space 정보
ref	int8	-	• MOVE_REFERENCE_BASE =0 • MOVE_REFERENCE_WORLD=2
mod	int8	-	입력 모드 (입력점개수가 2개인 경우에만 유효함) • 0: 현재 Tool-z방향 기준으로 사용자 좌표계의 z방향 정의 • 1: x1의 z방향 기준으로 사용자 좌표계의 z방향 정의

- **리턴**

인수명	자료형	기본값	설명
conv_posx	float64[6]	-	6개의 Task Space 정보

7.7.10 SetUserCartCoord1.srv

- **기능**

기준 좌표계(ref) 기반의 새로운 사용자좌표계를 설정하기 위한 서비스입니다. Workcell Item에서 설정한 좌표계를 포함하여 총 20개의 사용자좌표계를 설정할 수 있으며, 20개가 넘어가면 새로운 직교 좌표계를 설정할 수 없습니다. 명령어를 통해 설정한 사용자좌표계는 프로그램 실행 종료 시 삭제되므로, 사용자좌표계 정보를 유지하려면 Workcell Item에서 사용자좌표계를 설정하세요.

- **인수**

인수명	자료형	기본값	설명
pos	float64[6]	-	사용자좌표계 정보 (위치 및 방향)
ref	int8	-	reference coordinate · DR_BASE: base coordinate · DR_WORLD: world coordinate

알아두기

· ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
id	int8	-	사용자 좌표계 ID(101~120) 혹은 실패(-1)

7.7.11 SetUserCartCoord2.srv

- **기능**

사용자가 입력좌표계(ref) 기준의 포즈 x1, x2, x3를 사용하여 새로운 직교 좌표계를 설정하기 위한 서비스입니다. [1]x1x2의 단위 벡터를 ux, x1x2로부터 x3까지 최단거리로 잇는 vector의 단위벡터를 uy로 하여, ux, uy, uz를 각 축의 방향 벡터, 원점은 입력좌표계(ref) 기준의 pos에 위치한 직교 좌표계를 생성합니다. Workcell Item에서 설정한 좌표계를 포함하여 총 20개의 사용자좌표계를 설정할 수 있으며, 20개가 넘어가면 새로운 직교 좌표계를 설정할 수 없습니다. 명령어를 통해 설정한 사용자좌표계는 프로그램 실행 종료 시 삭제되므로, 사용자좌표계 정보를 유지하려면 Workcell Item에서 사용자좌표계를 설정하세요.

[1]M2.0.2 이전 버전에서는 x2x1의 단위 벡터를 ux로 사용

- **인수**

인수명	자료형	기본값	설명
x1	float64[6]	-	6개의 Task Space 정보
x2	float64[6]	-	6개의 Task Space 정보
x3	float64[6]	-	6개의 Task Space 정보
pos	float64[6]	-	6개의 Task Space 정보
ref	int8	0	・ MOVE_REFERENCE_BASE =0 ・ MOVE_REFERENCE_WORLD=2

알아두기

・ ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
id	int8	-	사용자 좌표계 ID(101~120) 혹은 실패(-1)

7.7.12 SetUserCartCoord3.srv

- **기능**

사용자가 입력좌표계(ref) 기준의 벡터 u1과 v1를 사용하여 새로운 직교 좌표계를 설정하기 위한 서비스입니다. 직교 좌표계의 원점은 입력좌표계(ref) 기준의 pos에 위치하고, x축/y축 basis는 vector u1과 v1에 주어집니다. 나머지 방향은 u1 x v1에 의해 정해집니다. u1과 v1이 orthogonal 하지 않은 경우, u1과 v1이 span 하는 평면상에 u1과 수직인 v1'를 y축의 방향 vector로 설정합니다. Workcell Item에서 설정한 좌표계를 포함하여 총 20개의 사용자좌표계를 설정할 수 있으며, 20개가 넘어가면 새로운 직교 좌표계를 설정할 수 없습니다. 명령어를 통해 설정한 사용자좌표계는 프로그램 실행 종료 시 삭제되므로, 사용자좌표계 정보를 유지하려면 Workcell Item에서 사용자좌표계를 설정하세요.

- **인수**

인수명	자료형	기본값	설명
u1	float64[3]	-	X축 단위벡터
v1	float64[3]	-	y축 단위벡터
pos	float64[6]	-	6개의 Task Space 정보
ref	int8	0	・ MOVE_REFERENCE_BASE =0 ・ MOVE_REFERENCE_WORLD=2

알아두기

・ ref 의 인자에서 MOVE_REFERENCE_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

- **리턴**

인수명	자료형	기본값	설명
id	int8	-	사용자 좌표계 ID(101~120) 혹은 실패(-1)

7.7.13 OverwriteUserCartCoord.srv

- **기능**

본 서비스는 M2.50 이상의 버전에서만 사용 가능합니다.

요청하는 ID(id)의 사용자 좌표계의 좌표계 위치(pos), 기준 좌표계(ref) 정보를 변경하기 위한 서비스입니다.

인수명	자료형	기본값	설명
id	int8	-	사용자 좌표계 ID
pos	float64[6]	-	사용자좌표계 정보 (위치 및 방향)
ref	int8	0	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_WORLD=2

- **리턴**

인수명	자료형	기본값	설명
id	int8	-	사용자 좌표계 ID(101~120) 혹은 실패(-1)

7.7.14 GetUserCartCoord.srv

- **기능**

본 서비스는 M2.50 이상의 버전에서만 사용 가능합니다.

해당하는 ID(id)의 사용자 좌표계의 정보인 참조 기준 및 위치정보를 조회하기 위한 서비스입니다.

- **인수**

인수명	자료형	기본값	설명
id	int8	-	사용자 좌표계 ID

- **리턴**

인수명	자료형	기본값	설명
conv_posx	float64[6]	-	6개의 Task Space 정보
ref	int8	-	사용자 좌표계 정보(위치 및 방향)

7.7.15 SetDesiredForce.srv

- **기능**

전역으로 설정된 좌표계(SetRefCoord.srv 참조) 기준으로 힘 제어 목표값, 힘 제어 방향, 힘 목표값, 외력참조모드를 설정합니다.

- **인수**

인수명	자료형	기본값	설명
Fd	float64[6]	[0, 0, 0, 0, 0, 0]	(Translational) 힘 성분 3개, (Rotational) 모멘트 성분 3개
dir	int8[6]	[0, 0, 0, 0, 0, 0]	1이면 해당 방향 힘 제어 0이면 해당 방향 compliance 제어
ref	int8	1	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_TOOL=1 · MOVE_REFERENCE_WORLD=2 · MOVE_REFERENCE_USER=101~120
time	float64	0.0	힘을 증가시키는데 소요되는 시간 [sec] 범위 0~1.0
mod	int8	0	FORCE_MODE_ABSOLUTE(0): 힘제어 시 외력을 센서값 그대로(절대적) 참조 FORCE_MODE_RELATIVE(1): 힘제어 초기의 센서값을 기준으로 상대적인 외력만 참조

- **리턴**

인수명	자료형	기본값	설명
res	int8	-	성공 여부 0 : 성공 기타 : 실패

7.7.16 ReleaseForce.srv

- **기능**

힘 제어 목표값을 time 값 동안 0으로 줄이고 작업 공간을 순응 제어로 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
time	float64	0.0	힘을 감소시키는데 소요되는 시간 범위 0~1.0

- **리턴**

인수명	자료형	기본값	설명
res	int8	-	성공 여부 0 : 성공 기타 : 실패

7.7.17 CheckPositionCondition.srv

- **기능**

주어진 위치 상태를 확인하기 위한 서비스입니다. while 또는 if 조건과 함께 사용하여 해당 조건을 반복 확인할 수 있습니다. axis, pos는 입력좌표계(ref) 기준의 축방향 및 포즈입니다.

- **인수**

인수명	자료형	기본값	설명
axis	int8	-	・ FORCE_AXIS_X = 0 ・ FORCE_AXIS_Y = 1 ・ FORCE_AXIS_Z = 2
min	float64	DR_COND_NONE	최소값
max	float64	DR_COND_NONE	최대값
ref	int8	1	・ MOVE_REFERENCE_BASE =0 ・ MOVE_REFERENCE_TOOL=1 ・ MOVE_REFERENCE_WORLD=2 ・ MOVE_REFERENCE_USER=101~120
mod	int8	-	・ MOVE_MODE_ABSOLUTE = 0 ・ MOVE_MODE_RELATIVE = 1
pos	float64[6]	-	6개의 Task Space 정보

알아두기

・ mod 가 DR_MV_MOD_ABS 인 경우는 절대 위치 기준으로 확인합니다.
・ mod 가 DR_MV_MOD_REL 인 경우는 pos 위치 기준으로 확인합니다.
・ pos 는 mod 가 DR_MV_MOD_REL 인 경우에만 의미가 있습니다.

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.7.18 CheckForceCondition.srv

▪ 기능

주어진 힘 상태를 확인하기 위한 서비스입니다. 단, 힘의 방향은 고려하지 않고 크기로만 비교합니다. while 또는 if 조건과 함께 사용하여 해당 조건을 반복 확인할 수 있습니다. 힘 측정 시 axis는 입력좌표계(ref) 기준의 축방향 이고 모멘트 측정 시 axis는 툴 좌표계 기준의 축방향 입니다.

▪ 인수

인수명	자료형	기본값	설명
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
min	float64	-	최소값 (min ≥ 0)
max	float64	-	최대값 (max ≥ 0)
ref	int8	None	• MOVE_REFERENCE_BASE =0 • MOVE_REFERENCE_TOOL=1 • MOVE_REFERENCE_WORLD=2 • MOVE_REFERENCE_USER=101~120

▪ 리턴

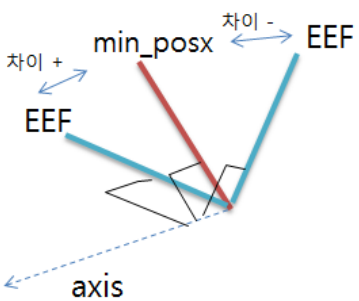
인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.7.19 CheckOrientationCondition1.srv

- **기능**

현재 로봇 엔드이펙터의 자세 정보와 주어진 위치 자세 간 차이의 상태를 확인하기 위한 서비스입니다. 현재 자세와 주어진 자세 간의 차이는 알고리즘 내부에서 회전행렬로 변환되어 “AngleAxis” 기법으로 차이 값(rad 단위)을 리턴합니다. 차이가 + 값이면 true를, - 값이면 false를 리턴합니다. 현재 자세를 기준으로, 주어진 position보다 차이가 +인지 –인지 확인할 때 사용합니다. 사용 예로, 직접교시 position을 이용하여 현재 위치와 차이가 + 방향인지, - 방향인지를 판단한 후 orientation limit에 대한 조건을 만들 수 있습니다. 또한 while 또는 if 조건과 함께 사용하여 해당 조건을 반복 확인할 수 있습니다.

- Min만 설정 시: 차이가 +이면 True, -이면 False
- Min, Max 설정 시: min과의 차이가 -이고, max 차이가 +이면 True, 반대면 False
- Max만 설정 시: max 차이가 -이면 True, +이면 False



- **인수**

인수명	자료형	기본값	설명
axis	int8	-	· FORCE_AXIS_A = 10 · FORCE_AXIS_B = 11 · FORCE_AXIS_C = 12
min	float64[6]	-	6개의 Task Space 정보
max	float64[6]	-	6개의 Task Space 정보
ref	int8	1	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_TOOL=1 · MOVE_REFERENCE_WORLD=2

인수명	자료형	기본값	설명
			• MOVE_REFERENCE_USER=101~120
mod	int8	0	• MOVE_MODE_ABSOLUTE = 0 • MOVE_MODE_RELATIVE = 1

- **리턴**

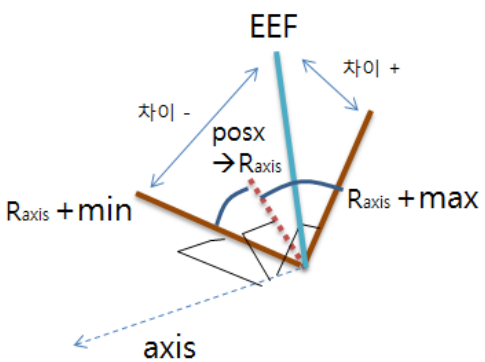
인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.7.20 CheckOrientationCondition2.srv

- **기능**

현재 로봇 엔드이펙터의 자세와 회전각 범위 차이에 대한 상태를 확인하기 위한 서비스 입니다. 현재 자세와 회전각 범위에 대한 차이는 알고리즘 내부에서 회전행렬로 변환되어 “AngleAxis” 기법으로 차이 값(rad 단위)을 리턴합니다. 차이가 + 값이면 true를, -값이면 false를 리턴합니다. 현재 자세를 기준으로, 주어진 position과 회전각 범위 차이가 +인지 –인지 확인할 때 사용합니다. 사용 예로, 어떤 기준이 되는 position에서 min, max로 회전각 범위를 설정하여, 현재 위치와 차이가 + 방향인지, - 방향인지 판단한 후 orientation limit에 대한 조건을 만들 수 있습니다. 또한 while 또는 if 조건과 함께 사용하여 해당 조건을 반복 확인할 수 있습니다.

- Min만 설정 시: 차이가 +이면 True, -이면 False
- Min, Max 설정 시: min과의 차이가 -이고, max 차이가 +이면 True, 그 반대이면 False
- Max만 설정 시: max 차이가 -이면 True, +이면 False



알아두기

회전각 범위: 주어진 position 에서 설정된 axis 를 기준으로, 상대적인 각도 범위(min, max)를 말합니다. 인자 ref 에 따라 주어진 position 의 기준 좌표계가 정해집니다.

- **인수**

인수명	자료형	기본값	설명
axis	int8	-	・ FORCE_AXIS_X = 0 ・ FORCE_AXIS_Y = 1 ・ FORCE_AXIS_Z = 2
min	float64	-	최솟값

인수명	자료형	기본값	설명
max	float64	-	최댓값
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]	-	6개의 Task Space 정보

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.7.21 CoordTransform.srv

- **기능**

'ref_in' 기준 좌표계에서 표현되는 'pose_in' Task 좌표를 'ref_out' 기준 좌표계에서 표현되는 Task 좌표로 변환하기 위한 서비스입니다. 아래의 경우에 대한 좌표변환 계산을 지원 합니다.

- (ref_in) 월드 기준 좌표계 → (ref_out) 월드 기준 좌표계
- (ref_in) 월드 기준 좌표계 → (ref_out) 베이스 기준 좌표계
- (ref_in) 월드 기준 좌표계 → (ref_out) 툴 기준 좌표계
- (ref_in) 월드 기준 좌표계 → (ref_out) 사용자 기준 좌표계
- (ref_in) 베이스 기준 좌표계 → (ref_out) 월드 기준 좌표계
- (ref_in) 베이스 기준 좌표계 → (ref_out) 베이스 기준 좌표계
- (ref_in) 베이스 기준 좌표계 → (ref_out) 툴 기준 좌표계
- (ref_in) 베이스 기준 좌표계 → (ref_out) 사용자 기준 좌표계
- (ref_in) 툴 기준 좌표계 → (ref_out) 월드 기준 좌표계
- (ref_in) 툴 기준 좌표계 → (ref_out) 베이스 기준 좌표계
- (ref_in) 툴 기준 좌표계 → (ref_out) 툴 기준 좌표계
- (ref_in) 툴 기준 좌표계 → (ref_out) 사용자 기준 좌표계
- (ref_in) 사용자 기준 좌표계 → (ref_out) 월드 기준 좌표계
- (ref_in) 사용자 기준 좌표계 → (ref_out) 베이스 기준 좌표계
- (ref_in) 사용자 기준 좌표계 → (ref_out) 툴 기준 좌표계
- (ref_in) 사용자 기준 좌표계 → (ref_out) 사용자 기준 좌표계

- **인수**

인수명	자료형	기본값	설명
pose_in	float64[6]	-	6개의 Task Space 정보
ref_in	int8	DR_COND_NONE	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_TOOL=1 · MOVE_REFERENCE_WORLD=2 · MOVE_REFERENCE_USER=101~120

인수명	자료형	기본값	설명
ref_out	int8	DR_COND_NONE	· MOVE_REFERENCE_BASE =0 · MOVE_REFERENCE_TOOL=1 · MOVE_REFERENCE_WORLD=2 · MOVE_REFERENCE_USER=101~120

- **리턴**

인수명	자료형	기본값	설명
conv_posx	float64[6]	-	6개의 Task Space 정보

7.7.22 GetWorkpieceWeight.srv

- **기능**

작업물의 무게를 측정하기 위한 서비스입니다.

- **인수**

없음

- **리턴**

인수명	자료형	기본값	설명
weight	float64	-	0 이상의 값 : 측정 무게 값 음수 : 오류

7.7.23 ResetWorkpieceWeight.srv

- **기능**

소재의 무게를 측정하기 전 알고리즘의 초기화를 위해 소재의 무게정보를 초기화 하기 위한 서비스입니다.

- **인수**

없음

- **리턴**

인수명	자료형	기본값	설명
res	int8	-	성공 여부 0 : 성공 기타 : 실패

7.8 Service/io

7.8.1 SetCtlBoxDigitalOutput.srv

- **기능**

로봇 제어기에서 컨트롤 박스에 장착된 디지털 접점에 신호를 출력하기 위한 서비스 입니다.

- **인수**

인수명	자료형	기본값	설명
index	int8	-	제어기 내 디지털 출력 접점 번호(1~16)
value	int8		출력 값 : OFF =0, ON =1

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.8.2 GetCtlBoxDigitalInput.srv

- **기능**

로봇 제어기에서 컨트롤 박스에 장착된 디지털 접점의 신호를 확인하기 위한 서비스 입니다.

- **인수**

인수명	자료형	기본값	설명
index	int8	-	제어기 내 디지털 입력 접점 번호(1~16)

- **리턴**

인수명	자료형	기본값	설명
value	bool	-	OFF =0, ON =1

7.8.3 SetToolDigitalOutput.srv

- **기능**

로봇 제어기에서 로봇 끝단에 장착된 디지털 접점에 신호를 출력하기 위한 서비스 입니다.

- **인수**

인수명	자료형	기본값	설명
index	int8	-	로봇 플랜지 단 디지털 출력 접점 번호(1~6)
value	int8		출력 값 : OFF =0, ON =1

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.8.4 GetToolDigitalIntput.srv

- **기능**

로봇 제어기에서 로봇 끝단에 장착된 디지털 접점의 신호를 확인하기 서비스 입니다.

- **인수**

인수명	자료형	기본값	설명
index	int8	-	로봇 플랜지 단 디지털 입력 접점 번호(1~6)

- **리턴**

인수명	자료형	기본값	설명
value	bool	-	OFF =0, ON =1

7.8.5 SetCtlBoxAnalogOutputType.srv

- **기능**

로봇 제어기에서 컨트롤 박스에 장착된 아날로그 출력 접점에 대한 채널 모드를 설정하기 위한 서비스 입니다.

- **인수**

인수명	자료형	기본값	설명
channel	int8	-	제어기 내 아날로그 출력 채널 : 1 or 2
mode	int8	-	동작 모드 • current =0 • voltage =1

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.8.6 SetCtlBoxAnalogInputType.srv

- **기능**

로봇 제어기에서 컨트롤 박스에 장착된 아날로그 입력 접점에 대한 채널 모드를 설정하기 위한 서비스 입니다.

- **인수**

인수명	자료형	기본값	설명
channel	int8	-	제어기 내 아날로그 입력 채널 : 1 or 2
mode	int8	-	동작 모드 • current =0 • voltage =1

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.8.7 SetCtlBoxAnalogOutput.srv

- **기능**

로봇 제어기에서 컨트롤 박스에 장착된 아날로그 접점에 신호를 출력하기 위한 서비스 입니다.

- **인수**

인수명	자료형	기본값	설명
channel	int8	-	제어기 내 아날로그 출력 채널 : 1 or 2
value	float64	-	아날로그 신호 출력 • 전류 모드인 경우: 4.0~20.0 [mA] • 전압 모드인 경우: 0~10.0 [V]

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.8.8 GetCtlBoxAnalogInput.srv

- **기능**

로봇 제어기에서 컨트롤 박스에 장착된 아날로그 접점의 신호를 확인하기 위한 서비스 입니다.

- **인수**

인수명	자료형	기본값	설명
channel	int8	-	제어기 내 아날로그 입력 채널 : 1 or 2

- **리턴**

인수명	자료형	기본값	설명
value	float	-	해당 채널의 입력 값 • 전류 모드인 경우: 4.0~20.0 [mA] • 전압 모드인 경우: 0~10.0 [V]

7.8.9 GetCtrlBoxDigitalOutput.srv

7.9 Service/modbus

7.9.1 ConfigCreateModbus.srv

- **기능**

로봇 제어기에서 Modbus의 I/O 신호 사전에 등록하여 사용하기 위한 서비스이다, 본 함수를 이용하여 등록된 Modbus I/O 신호 정보는 메모리에 저장됨으로 재부팅 후 다시 설정해야 하지만, T/P 어플리케이션에서 등록한 경우에는 초기화 과정에서 추가됨으로 재사용이 가능합니다.

- **인수**

인수명	자료형	기본값	설명
name	string	-	modbus signal 이름
ip	string	-	modbus 모듈 ip 주소
port	int8	-	modbus 모듈 port
reg_type	int8	-	modbus 레지스터 타입 • MODBUS_REGISTER_TYPE_DISCRETE_INPUTS • MODBUS_REGISTER_TYPE_COILS • MODBUS_REGISTER_TYPE_INPUT_REGISTER • MODBUS_REGISTER_TYPE_HOLDING_REGISTER
index	int8	-	Modbus signal의 index
value	int8	-	type이 MODBUS_REGISTER_TYPE_COILS 또는 MODBUS_REGISTER_TYPE_HOLDING_REGISTER 일 때 출력값 (그 외 경우에는 무시됩니다.)
slaveid	int	255	• Slave ID of the ModbusTCP module (0 or 1-247 or 255) 0 : Broadcase address 255 : Default value for ModbusTCP

알아두기

- slaveid 인자는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.9.2 ConfigDeleteModbus.srv

▪ 기능

로봇 제어기에 사전에 등록된 Modbus I/O 신호 정보를 삭제하기 위한 서비스 입니다.

▪ 인수

인수명	자료형	기본값	설명
name	string	-	등록된 modbus 신호의 이름

▪ 리턴

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.9.3 SetModbusOutput.srv

- **기능**

로봇 제어기에서 Modbus I/O 신호 접점에 신호를 출력하기 위한 서비스 입니다.

- **인수**

인수명	자료형	기본값	설명
name	string	-	modbus 이름
value	int32	-	• Modbus digital I/O 인 경우: 0 or 1 • Modbus analog I/O 인 경우: 데이터

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.9.4 GetModbuInput.srv

- **기능**

로봇 제어기에서 Modbus I/O 신호 접점의 신호를 확인하기 위한 서비스 입니다.

- **인수**

인수명	자료형	기본값	설명
name	string	-	modbus 이름

- **리턴**

인수명	자료형	기본값	설명
value	int32	-	• Modbus Digital I/O 인 경우: 0 or 1 • Modbus Analog 모듈인 경우: 데이터

7.10 Service/gripper

7.10.1 SerialSendData.srv

- **기능**

실질적인 serial 통신을 통하여 gripper를 제어합니다.
serial_node_example

- **인수**

인수명	자료형	기본값	설명
data	string	-	전송하고자 하는 스트링

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

7.10.2 RobotiqMove.srv

- **기능**

  시뮬레이터 환경에서 robotiq 사의 gripper를 제어하는 서비스 입니다.

- **인수**

인수명	자료형	기본값	설명
width	float	-	2핑거 그립퍼 폭 : 0.0(open)~0.8(close)

- **리턴**

인수명	자료형	기본값	설명
success	bool	-	성공 여부 : True or False

8. 모션 관련 함수

8.1 posj(q1=0, q2=0, q3=0, q4=0, q5=0, q6=0)

- **기능**

조인트 공간 각도를 좌표값으로 지정합니다.

- **인수**

번호	자료형	기본값	설명
q1	float list posj	0	1축 angle 또는 angle list 또는 posj
q2	float	0	2축 angle
q3	float	0	3축 angle
q4	float	0	4축 angle
q5	float	0	5축 angle
q6	float	0	6축 angle

- **리턴**

posj

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

- **예제**

```
q1 = posj()                     # q1=posj(0,0,0,0,0,0)
q2 = posj(0, 0, 90, 0, 90, 0)
q3 = posj([0, 30, 60, 0, 90, 0])  # q3=posj(0,30,60,0,90,0)
```

- **관련 명령어**

movej()/amovej()/movesj()/amovesj()

8.2 posx(x=0, y=0, z=0, w=0, p=0, r=0)

▪ 기능

작업 공간을 좌표값으로 지정합니다.

▪ 인수

인수명	자료형	기본값	설명
x	float list posx	0	z position 또는 position list 또는 posx
y	float	0	y position
z	float	0	z position
w	float	0	w orientation(기준좌표계의 Z방향 회전)
p	float	0	p orientation(w회전된 좌표계의 Y방향 회전)
r	float	0	r orientation(w,p회전된 좌표계의 Z방향 회전)

▪ 리턴

posx

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
movej([0,0,90,0,90,0], v=10, a=20)
x2 = posx(400, 300, 500, 0, 180, 0)
x3 = posx([350, 350, 450, 0, 180, 0])      #x3=posx(350, 350, 450, 0, 180, 0)
x4 = posx(x2)                              #x4=posx(400, 300, 500, 0, 180, 0)
movel(x2, v=100, a=200)
```

▪ 관련 명령어

movel()/movec()/movejx()/amovel()/amovec()/amovejx()

8.3 trans(pos, delta, ref, ref_out)

▪ 기능

- ref 좌표계 기준으로 정의된 pos(포즈)를 동일 좌표계를 기준으로 delta만큼 이동/회전하여 ref_out 좌표계 기준으로 변환한 후 리턴합니다.
- 단, ref 인자가 Tool Coordinate인 경우, ref_out인자는 무시되며 pos와 동일한 좌표계 기준의 결과를 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
pos	posx list (float[6])	-	posx 또는 position list
delta	posx list (float[6])	-	posx 또는 position list
ref	int	None	reference coordinate ・ DR_BASE : base coordinate ・ DR_WORLD : world coordinate ・ DR_TOOL : tool coordinate ・ user coordinate: 사용자 정의
ref_out	int	DR_BASE	reference coordinate ・ DR_BASE : base coordinate ・ DR_WORLD : world coordinate ・ user coordinate: 사용자 정의

알아두기

・ ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.
・ ref_out 인자는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
posx list (float[6])	task space point

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
p0 = posj(0,0,90,0,90,0)
movej(p0, v=30, a=30)

x1 = posx(200, 200, 200, 0, 180, 0)
delta = [100, 100, 100, 0, 0, 0]
x2 = trans(x1, delta, DR_BASE, DR_BASE)

x1_base = posx(500, 45, 700, 0, 180, 0)
x4 = trans(x1_base, [10, 0, 0, 0, 0, 0], DR_TOOL)
movel(x4, v=100, a=100, ref=DR_BASE)

uu1 = [1, 1, 0]
vv1 = [-1, 1, 0]
pos = posx(559, 34.5, 651.5, 0, 180.0, 0)
DR_userTC1 = set_user_cart_coord(uu1, vv1, pos) #사용자 정의 좌표계 등록
x1_userTC1 = posx(30, 20, 100, 0, 180, 0)          #사용자좌표계 상의 posx
x9 = trans(x1_userTC1, [0, 0, 50, 0, 0, 0], DR_userTC1, DR_BASE)
movel(x9, v=100, a=100, ref=DR_BASE)
```

- **관련 명령어**

posx()/addto()

8.4 posb(seg_type, posx1, posx2=None, radius=0)

▪ 기능

- 정속 블렌딩 모션(moveb, amoveb)의 입력 인자로, 각 경유점의 좌표와 단위 경로 형태(라인 또는 원호)의 정보를 갖는 posb는 블랜딩되는 trajectory의 단위 세그먼트 객체를 정의합니다.
- seg_type이 line인 경우(DR_LINE)는 posx1만 입력, circle인 경우(DR_CIRCLE)는 posx2까지 입력합니다. radius는 이어지는 segment와의 blending 반경을 설정합니다.

▪ 인수

인수명	자료형	기본값	설명
seg_type	Int	-	DR_LINE DR_CIRCLE
posx1	posx	-	1st task posx
posx2	posx	-	2nd task posx
radius	float	0	Blending radius [mm]

▪ 리턴

posb

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

- **예제**

```
q0 = posj(0, 0, 90, 0, 90, 0)
movej(q0,vel=30,acc=60)
x0 = posx(564,   34, 690, 0, 180, 0)
movel(x0, vel=200, acc=400)      # 시작위치로 이동

x1 = posx(564, 200, 690, 0, 180, 0)
seg1 = posb(DR_LINE, x1, radius=40)
x2 = posx(564, 100, 590, 0, 180, 0)
x2c = posx(564, 200, 490, 0, 180, 0)
seg2 = posb(DR_CIRCLE, x2, x2c, radius=40)
x3 = posx(564, 300, 490, 0, 180, 0)
seg3 = posb(DR_LINE, x3, radius=40)
x4 = posx(564, 400, 590, 0, 180, 0)
x4c = posx(564, 300, 690, 0, 180, 0)
seg4 = posb(DR_CIRCLE, x4, x4c, radius=40)
x5 = posx(664, 300, 690, 0, 180, 0)
seg5 = posb(DR_LINE, x5, radius=40)
x6 = posx(564, 400, 690, 0, 180, 0)
x6c = posx(664, 500, 690, 0, 180, 0)
seg6 = posb(DR_CIRCLE, x6, x6c, radius=40)
x7 = posx(664, 400, 690, 0, 180, 0)
seg7 = posb(DR_LINE, x7, radius=40)
x8 = posx(664, 400, 590, 0, 180, 0)
x8c = posx(564, 400, 490, 0, 180, 0)
seg8 = posb(DR_CIRCLE, x8, x8c, radius=0)          # 마지막 radius는 0이어야 함
       # 만약 0이 아닌 경우 0으로 처리됨

b_list = [seg1, seg2, seg3, seg4, seg5, seg6, seg7, seg8]

moveb(b_list, vel=200, acc=400)
```

- **관련 명령어**

posx()/moveb()/amoveb()

8.5 fkin(pos, ref)

▪ 기능

조인트 공간에서 조인트각도 또는 이에 상응하는 자료형(float[6])을 입력받아 ref 좌표계 기준의 TCP의 포즈(태스크공간의 위치 및 자세)를 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
pos	posj list (float[6])	-	posj 또는 position list
ref	int	DR_BASE	reference coordinate • DR_BASE : base coordinate • DR_WORLD : world coordinate

알아두기

• ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
posx	Task space point

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

▪ 예제

```
q1 = posj(0, 0, 90, 0, 90, 0)
movej(q1,v=10,a=20)
q2 = posj(30, 0, 90, 0, 90, 0)
x2 = fkin(q2, DR_WORLD)    # x2: 관절값 q2에 대응하는 로봇끝단(TCP)의 공간좌표
movel(x2,v=100,a=200,ref=DR_WORLD)    # x2로 직선모션 수행
```

8.6 ikin(pos, sol_space, ref)

▪ 기능

작업 공간내 기준 좌표계(ref)의 로봇 포즈에 상응하는 8개의 관절형상 중 지정한 관절형상(sol_space)에 해당하는 관절각도를 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
pos	posx list (float[6])	-	posx 또는 position list
sol_space	int	-	solution space
ref	int	DR_BASE	reference coordinate ・ DR_BASE : base coordinate ・ DR_WORLD : world coordinate

알아두기

・ ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 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

▪ 리턴

값	설명
posj	Joint space point

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
x1 = posx(370.9, 719.7, 651.5, 90, -180, 0)
q1 = ikin(x1, 2, DR_BASE) # 로봇끝단의 좌표가 x1이 되는 관절각 q1 (8가지 경우 중 둘째)
# q1=posj(60.3, 81.0, -60.4, -0.0, 159.4, -29.7) (M1013, tcp=(0,0,0)경우)
movej(q1,v=10,a=20)
```

8.7 set_velj(vel)

- **기능**

본 명령어를 사용한 이후 조인트 모션(movej, movejx, amovej, amovejx)에서의 전역 속도를 설정합니다. 전역적으로 설정된 vel은 이후 movej() 호출 시 속도 인자를 명시적으로 입력하지 않는 경우에 default 속도로 적용됩니다.

- **인수**

인수명	자료형	기본값	설명
vel	float list (float[6])	-	velocity(모든 축에 동일) 또는 velocity(축별 velocity)

- **리턴**

값	설명
0	성공

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

- **예제**

```
#1
Q1 = posj(0,0,90,0,90,0)
Q2 = posj(0,0,0,0,90,0)
movej(Q1, vel=10, acc=20)
set_velj(30) # 전역 조인트 속도를 30(deg/sec)로 설정하십시오.
set_accj(60) # 전역 조인트 가속도를 60(deg/sec2)로 설정하십시오. [set_accj() 참조]
movej(Q2) # Q2로의 조인트 모션 속도는 전역 속도인 30(deg/sec)입니다.
movej(Q1, vel=20, acc=40) # Q1으로의 조인트 모션 속도는 지정 속도인
20(deg/sec)입니다.
#2
set_velj(20.5) # 소수점 입력 가능합니다.
set_velj([10, 10, 20, 20, 30, 10]) # 축별 전역 속도 지정 가능합니다.
```

- **관련 명령어**

set_accx()/movej()/movejx()/movesj()amovej()/amovejx()/amovesj()

8.8 set_accj(acc)

- **기능**

본 명령어를 사용한 이후의 조인트 모션(movej, movejx, amovej, amovejx)에서의 전역 가속도를 설정합니다. 전역적으로 설정된 가속도는 이후 movej() 호출 시 가속도 인자를 명시적으로 입력하지 않는 경우에 default 가속도로 적용됩니다.

- **인수**

인수명	자료형	기본값	설명
acc	float list (float[6])	-	acceleration(모든 축에 동일) 또는 acceleration(축별 acceleration)

- **리턴**

값	설명
0	성공

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

- **예제**

```
#1
Q1 = posj(0,0,90,0,90,0)
Q2 = posj(0,0,0,0,90,0)
movej(Q1, vel=10, acc=20)
set_velj(30) # 전역 조인트 속도를 30(deg/sec)로 설정하십시오. #[set_velj() 참조]
set_accj(60) # 전역 조인트 가속도를 60(deg/sec2)로 설정하십시오.
movej(Q2) # Q2로의 조인트 모션 가속도는 전역 가속도인 60(deg/sec2)입니다.
movej(Q1, vel=20, acc=40) # Q1으로의 조인트 모션 가속도는 지정 가속도인
40(deg/sec2)입니다.
#2
set_accj(30.55)
set_accj([30, 40, 30, 30, 30, 10])
```

- **관련 명령어**

set_velj()/movej()/movejx()/movesj()/amovej()/amovejx()amovesj()

8.9 set_velx(vel1, vel2)

- **기능**

작업 공간 모션의 속도를 전역적으로 설정합니다. 전역적으로 설정된 속도 velx는 movel(), amovel(), movec(), movesx()과 같은 태스크 모션을 호출할 때 속도 값을 입력하지 않는 경우에는 default 속도로 적용됩니다. 설정값 중 vel1은 TCP의 선속도를, vel2는 TCP의 회전 속도를 정의합니다.

- **인수**

인수명	자료형	기본값	설명
vel1	float	-	velocity 1
vel2	float	-	velocity 2

- **리턴**

값	설명
0	성공

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

- **예제**

```
#1
P0 = posj(0,0,90,0,90,0)
movej(P0)
P1 = posx(400,500,800,0,180,0)
P2 = posx(400,500,500,0,180,0)
movel(P1, vel=10, acc=20)
set_velx(30,20)    # 전역 태스크 속도를 30(mm/sec), 20(deg/sec)로 설정하십시오.
set_accx(60,40)    # 전역 태스크 가속도를 60(mm/sec2), 40(deg/sec2)로 설정십시오.
movel(P2)    # P2로의 태스크 모션 속도는 전역 속도인 30(mm/sec), 20(deg/sec)
movel(P1, vel=20, acc=40) # P1으로의 태스크 모션 속도는 지정 속도인
20(mm/sec), 20(deg/sec)입니다.
```

```
#2
set_velx(10.5, 19.4) # 소수점 입력 가능합니다.
```

- **관련 명령어**

set_accx()/movel()/movec()/movesx()/moveb()/move_spiral()/amovel()/amovec()/amovesx()/amoveb()/amove_spiral()

8.10 set_velx(vel)

- **기능**

작업 공간 모션의 선속도를 전역적으로 설정합니다. 전역적으로 설정된 속도 vel은 movel(), amovel(), movec(), movesx()과 같은 태스크 모션에서 속도 값을 입력하지 않는 경우의 default 속도로 적용됩니다. 설정값 vel은 TCP의 선속도를 정의하며 TCP의 회전 속도는 선속도에 비례하여 결정됩니다.

- **인수**

인수명	자료형	기본값	설명
vel	float	-	velocity

- **리턴**

값	설명
0	성공

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

- **예제**

```
#1
p0 = posj(0,0,90,0,90,0)
movej(p0)

P1 = posx(400,500,800,0,180,0)
P2 = posx(400,500,500,0,180,0)
movel(P1, vel=10, acc=20)
set_velx(30) # 전역 태스크 속도를 30(mm/sec)로 설정하십시오. 전역 태스크 각속도는 자동결정됨
set_accx(60) # 전역 태스크 가속도를 60(mm/sec2)로 설정십시오. 전역 태스크 각가속도는 자동결정됨
movel(P2) # P2로의 태스크 모션 선속도는 전역 속도인 30(mm/sec)
movel(P1, vel=20, acc=40) # P1으로의 태스크 모션 선속도는 지정 속도인
```

```
20(mm/sec) 입니다.
#2
set_velx(10.5) # 소수점 입력 가능합니다.
```

- **관련 명령어**

set_accx()/movel()/movec()/movesx()/moveb()/move_spiral()/amovel()/amovec()/amovesx()/amoveb()/amove_spiral()

8.11 set_accx(acc1, acc2)

- **기능**

작업 공간 모션의 가속도를 전역적으로 설정합니다. 전역적으로 설정된 가속도 accx는 movel(), amovel(), movec(), movesx()과 같은 태스크 모션을 호출할 때 가속도에 대한 값을 입력하지 않는 경우에는 default 가속도로 적용됩니다. 설정값 중 acc1은 TCP의 선 가속도를, acc2는 TCP의 회전 가속도를 정의합니다.

- **인수**

인수명	자료형	기본값	설명
acc1	float	-	acceleration 1
acc2	float	-	acceleration 2

- **리턴**

값	설명
0	성공

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

- **예제**

```
P0 = posj(0,0,90,0,90,0)
movej(P0)
P1 = posx(400,500,800,0,180,0)
P2 = posx(400,500,500,0,180,0)
movel(P1, vel=10, acc=20)
set_velx(30,20) # 전역 태스크 속도를 30(mm/sec), 20(deg/sec)로 설정하십시오.
set_accx(60,40) # 전역 태스크 가속도를 60(mm/sec2), 40(deg/sec2)로 설정하십시오.
movel(P2)      # P2로의 태스크 모션 가속도는 전역 가속도인 60(mm/sec2),
40(deg/sec2)
movel(P1, vel=20, acc=40)  # P1으로의 태스크 모션 가속도는 지정 가속도인
40(mm/sec), 40(deg/sec2)입니다.
```

- **관련 명령어**

set_velx()/movel()/movec()movesx()/moveb()/move_spiral()/amovel()/amovec()/amovesx()/amoveb()/amove_spiral()

8.12 set_accx(acc)

- **기능**

작업 공간 모션의 선가속도를 전역적으로 설정합니다. 전역적으로 설정된 가속도 acc는 movel(), amovel(), movec(), movesx()과 같은 태스크 모션에서 가속도에 대한 값을 입력하지 않는 경우에 default 가속도로 적용됩니다. 설정값 acc는 TCP의 선가속도를 정의하며 TCP의 회전 가속도는 선가속도에 비례하여 결정됩니다.

- **인수**

인수명	자료형	기본값	설명
acc	float	-	acceleration

- **리턴**

값	설명
0	성공

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

- **예제**

```
P0 = posj(0,0,90,0,90,0)
movej(P0)
P1 = posx(400,500,800,0,180,0)
P2 = posx(400,500,500,0,180,0)
movej(P0, vel=10, acc=20)
movel(P1, vel=10, acc=20)
set_velx(30) # 전역 태스크 속도를 30(mm/sec)로 설정하십시오. 전역 태스크 각속도는 자동결정됨
set_accx(60) # 전역 태스크 가속도를 60(mm/sec2)로 설정십시오. 전역 태스크 각가속도는 자동결정됨
movel(P2) # P2로의 태스크 모션 선가속도는 전역 가속도인 60(mm/sec2)
movel(P1, vel=20, acc=40) # P1으로의 태스크 모션 선가속도는 지정 가속도인 40(mm/sec2) 입니다.
```

- **관련 명령어**

set_velx()/movel()/movec()movesx()/moveb()/move_spiral()/amovel()/amovec()/amovesx()/amoveb()/amove_spiral()

8.13 set_tcp(name)

- **기능**

티치 팬던트에 등록된 tcp의 이름을 호출하여 현재 tcp로 설정합니다.

- **인수**

인수명	자료형	기본값	설명
name	string	-	TP에 등록된 tcp 이름

- **리턴**

값	설명
0	성공
음수값	실패

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
P0 = posj(0,0,90,0,90,0)
movej(P0)
set_tcp("tcp1") # TP에 tcp1으로 등록된 tcp 정보를 호출하여 현재의 tcp 값으로 설정하십시오.
P1 = posx(400,500,800,0,180,0)
movel(P1, vel=10, acc=20) # 인식한 tool 중심이 P1 위치로 이동
```

- **관련 명령어**

fkin()/ikin()/movel()/movejx()/movec()/movesx()/moveb()/amovel()/amovejx()/amovec()/amovesx()/amoveb()

8.14 set_ref_coord(coord)

▪ 기능

기준 좌표계를 설정합니다.

▪ 인수

인수명	자료형	기본값	설명
coord	int	-	기준 좌표계 • DR_BASE: Base Coordinate • DR_WORLD: World Coordinate • DR_TOOL: Tool Coordinate • user Coordinate: 사용자 정의

알아두기

• ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

리턴

값	설명
0	성공
음수값	실패

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

8.15 movej

- **기능**

로봇이 현재 관절위치에서 목표 관절위치(pos)로 이동합니다.

- **인수**

인수명	자료형	기본값	설명
pos	posj list (float[6])	-	posj 또는 joint angle list
vel (v)	float list (float[6])	None None	velocity(모든 축에 동일) 또는 velocity(축별 velocity)
acc (a)	float list (float[6])	None None	acceleration(모든 축에 동일) 또는 acceleration(축별 acceleration)
time (t)	float	None	도달 시간 [sec]
radius (r)	float	None	blending시 radius
mod	int	DR_MV_MOD_ABS	이동 기준 • DR_MV_MOD_ABS : 절대 • DR_MV_MOD_REL : 상대
ra	int	DR_MV_RA_DUPLICATE	Reactive motion mode • DR_MV_RA_DUPLICATE: duplicate • DR_MV_RA_OVERRIDE: override

알아두기

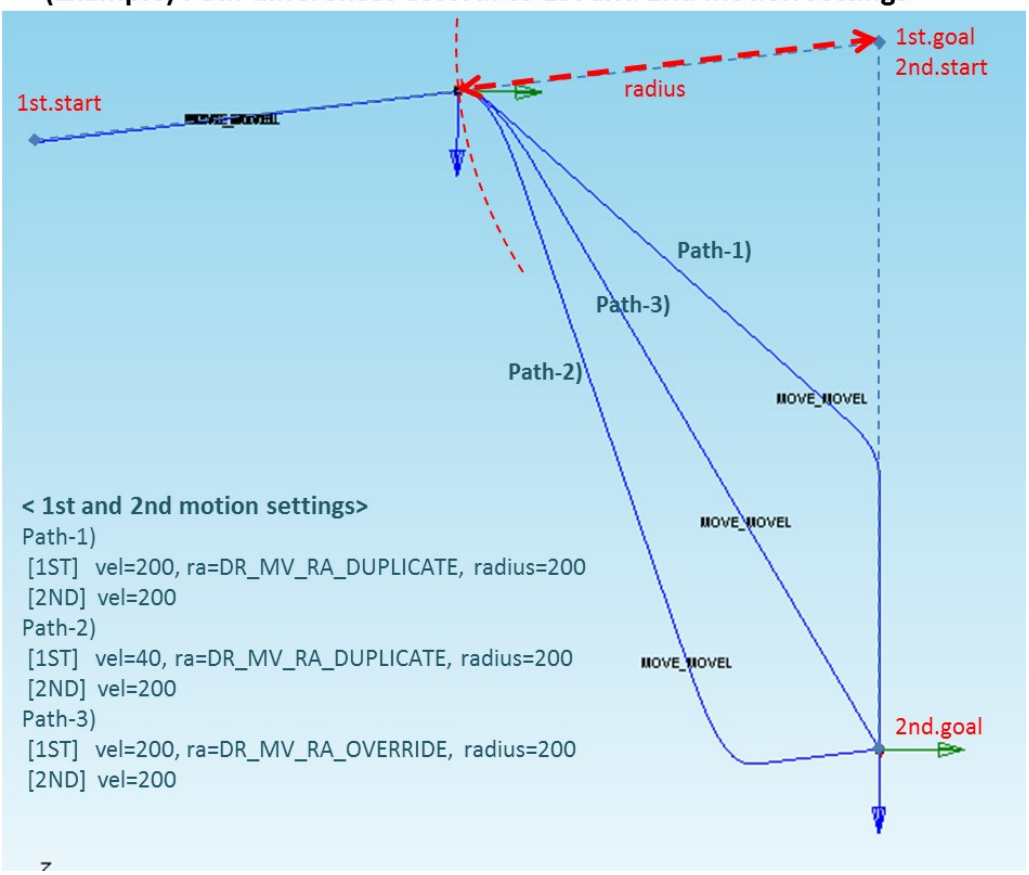
- 단축 인수를 지원합니다. (v:vel, a:acc, t:time, r:radius)
- vel 이 None 인 경우, _global_velj 이 적용됩니다. (_global_velj 초깃값은 0.0 이며, set_velj 에 의해 설정 가능)
- acc 이 None 인 경우, _global_accj 이 적용됩니다. (_global_accj 초깃값은 0.0 이며, set_accj 에 의해 설정 가능)
- time 지정 시, vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우, 0 으로 처리됩니다.

- radius 가 None 인 경우, 블렌딩 구간인 경우는 blending radius 로 처리되며 아닌 경우는 0 으로 처리됩니다.

⚠ 주의

ra=DR_MV_RA_DUPLICATE 및 radius>0 조건으로 후속 모션이 블렌딩 될 경우 선행모션의 잔여거리, 속도, 가속도로 결정되는 잔여모션시간이 후행모션의 모션시간보다 큰 경우 후행모션이 먼저 종료된 후 선행모션이 종료될 수 있습니다. 관련한 사항은 아래 이미지를 잠고하십시오.

< (Example) Path differences accord. to 1st and 2nd motion settings>



- **리턴**

값	설명
0	성공
음수값	실패

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
Q1 = posj(0,0,90,0,90,0)
Q2 = posj(0,0,0,0,90,0)
movej(Q1, vel=10, acc=20)
        # 속도 10(deg/sec), 가속도 20(deg/sec^2)로 Q1관절각으로 이동
movej(Q2, time=5)
        # Q2관절각까지 5초의 도착시간을 가지고 이동
movej(Q1, v=30, a=60, r=200)
        # Q1관절각으로 이동하며 Q1의 공간위치로부터 200mm의 거리가 될 때 다
        음
        # 모션을 수행하도록 설정
movej(Q2, v=30, a=60, ra= DR_MV_RA_OVERRIDE)
        # 직전모션을 즉시 종료시키며 Blending하여 Q2관절각으로 이동
```

- **관련 명령어**

posj()/set_velj()/set_accj()/amovej()

8.16 movel

▪ 기능

로봇이 작업 공간 안에서 목표 위치(pos)로 직선을 따라 이동합니다.

▪ 인수

인수명	자료형	기본값	설명
pos	posx list (float[6])	-	posx 또는 position list
vel (v)	float list (float[2])	None None	velocity 또는 velocity1, velocity2
acc (a)	float list (float[2])	None None	acceleration 또는 acceleration1, acceleration2
time (t)	float	None	도달 시간 [sec] * time 지정 시, vel, acc를 무시하고 time 기준으로 처리
radius (r)	float	None	blending시 radius
ref	int	None	reference coordinate ・ DR_BASE: base coordinate ・ DR_TOOL: tool coordinate ・ user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 ・ DR_MV_MOD_ABS : 절대 ・ DR_MV_MOD_REL : 상대
ra	int	DR_MV_RA_DUPLICATE	Reactive motion mode ・ DR_MV_RA_DUPLICATE: duplicate ・ DR_MV_RA_OVERRIDE: override

알아두기

- 단축 인수를 지원합니다. (v:vel, a:acc, t:time, r:radius)
- vel 가 None 인 경우, _global_velx 이 적용됩니다. (_global_velx 초깃값은 0.0 이며, set_velx 에 의해 설정 가능)
- acc 가 None 인 경우, _global_accx 이 적용됩니다. (_global_accx 초깃값은 0.0 이며, set_accx 에 의해 설정 가능)

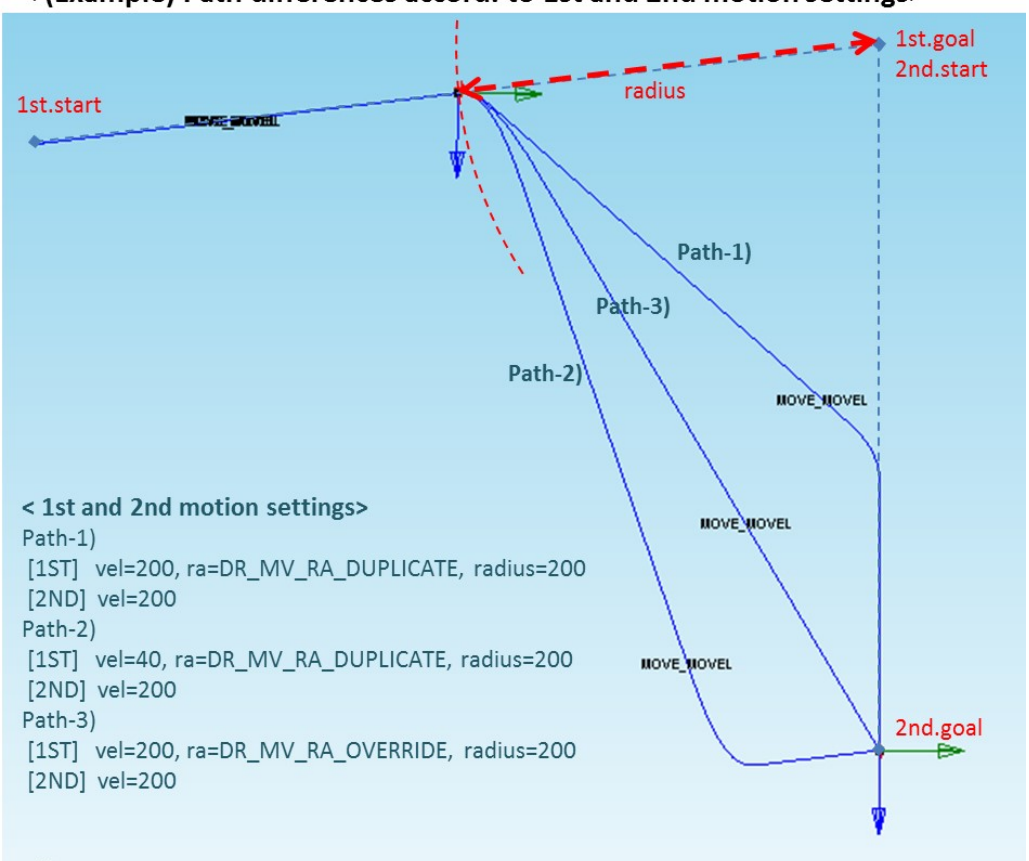
- vel 에 하나의 인자를 입력한 경우(예를들어, vel=30) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- acc 에 하나의 인자를 입력한 경우(예를들어, acc=60) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- time 지정 시, vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우, 0 으로 처리됩니다.
- radius 가 None 이고 블렌딩 구간인 경우는 blending radius 로 처리되고 아닌 경우는 0 으로 처리됩니다.
- ref 가 None 인 경우에는 _g_coord 이 적용됩니다. (_g_coord 초깃값은 DR_BASE 이며, set_ref_coord 명령에 의해 설정 가능)

  ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

⚠ 주의

ra=DR_MV_RA_DUPLICATE 및 radius>0 조건으로 후속 모션이 블렌딩 될 경우 선행모션의 잔여거리, 속도, 가속도로 결정되는 잔여모션시간이 후행모션의 모션시간보다 큰 경우 후행모션이 먼저 종료된 후 선행모션이 종료될 수 있습니다. 관련한 사항은 아래 이미지를 참고하십시오.

< (Example) Path differences accord. to 1st and 2nd motion settings>



▪ 리턴

값	설명
0	성공
음수값	실패

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

▪ 예제

```
P0 = posj(0,0,90,0,90,0)
movej(P0, v=30, a=30)
P1 = posx(400,500,800,0,180,0)
P2 = posx(400,500,500,0,180,0)
P3 = posx(30,30,30,0,0,0)
movel(P1, vel=30, acc=100)
        # 속도 30(mm/sec), 가속도 100(mm/sec2)로 P1위치로 이동
movel(P2, time=5)
        # P2위치로 5초의 도착시간을 가지고 이동
movel(P3, time=5, ref=DR_TOOL, mod=DR_MV_MOD_REL)
        # 시작위치에서 Tool좌표계기준으로 P3만큼의 상대위치로 5초의 도착시간을
        # 가지고 이동시킴
movel(P2, time=5, r=10)
        # P2위치까지 5초의 도착시간을 가지고 이동시키며 P2 위치로부터 10mm의
        # 거리가 될 때 다음 모션을 수행하도록 설정
```

▪ 관련 명령어

posx()/set_velx()/set_accx()/set_tcp()/set_ref_coord()/amovel()

8.17 movejx

- **기능**

로봇이 관절 공간 안에서 목표 위치(pos)로 이동합니다.

목표 위치는 작업공간 상의 posx형으로 입력하므로 movel과 동일하게 이동합니다. 하지만 이 로봇의 모션은 관절공간에서 이루어지기 때문에 목표 위치까지 직선경로가 보장되지 않습니다. 추가적으로 하나의 작업공간좌표(posx)에 대응하는 8가지의 관절조합형태(robot configuration)중 하나를 sol(solution space)에 지정하여야 합니다.

- **인수**

인수명	자료형	기본값	설명
pos	posx list (float[6])	-	posx 또는 position list
vel (v)	float list (float[6])	None None	velocity(모든 축에 동일) 또는 velocity(축별 velocity)
acc (a)	float list (float[6])	None None	acceleration(모든 축에 동일) 또는 acceleration(축별 acceleration)
time (t)	float	None	도달 시간 [sec]
radius (r)	float	None	blending시 radius
ref	int	None	reference coordinate • DR_BASE: base coordinate • DR_TOOL: tool coordinate • user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 • DR_MV_MOD_ABS: 절대 • DR_MV_MOD_REL: 상대
ra	int	DR_MV_RA_DUPLICATE	Reactive motion mode • DR_MV_RA_DUPLICATE: duplicate • DR_MV_RA_OVERRIDE: override
sol	int	0	Solution space

알아두기

- 단축 인수를 지원합니다. (v:vel, a:acc, t:time, r:radius)
- vel 이 None 인 경우, _global_velj 가 적용됩니다. (_global_velj 초깃값은 0.0 이며, set_velj 에 의해 설정 가능)
- acc 가 None 인 경우, _global_accj 가 적용됩니다. (_global_accj 초깃값은 0.0 이며, set_accj 에 의해 설정 가능)
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우, 0 으로 처리됩니다.
- radius 가 None 이고 블렌딩 구간인 경우는 blending radius 로 처리되며 아닌 경우는 0 으로 처리됩니다.
- ref 가 None 인 경우 _g_coord 가 적용됩니다. (_g_coord 초깃값은 DR_BASE 이며, set_ref_coord 명령에 의해 설정 가능)
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- 상대모션으로 입력하는 경우(mod=DR_MV_MOD_REL), 선행모션에 블렌딩을 사용하는 경우 에러가 발생하므로 movej() 또는 movel()을 이용하여 블렌딩하는 것을 권장합니다.
- 옵션 ra 및 vel/acc 에 따른 블렌딩을 수행할 경우 movej(), movel() 설명을 참조하십시오.

▪ Robot configuration (형태 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

▪ 리턴

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
P0 = posj(0,0,90,0,90,0)
movej(P0, v=30, a=30)
P1 = posx(400,500,800,0,180,0)
P2 = posx(400,500,500,0,180,0)
movel(P2, vel=100, acc=200)     # P2로 직선이동
X_tmp, sol_init = get_current_posx()    # P2위치에서 현재의 solution space를 얻어옴
movejx(P1, vel=30, acc=60, sol=sol_init)
# (가)속도 30(deg/sec), 60(deg/sec2)로 TCP끝단이 P1위치일 때의 관절각으로
# 이동 (직전P2위치에서의 solution space유지)
movejx(P2, time=5, sol=2)
# TCP끝단이 P2위치일 때의 관절각으로 5초의 도착시간을 가지고
# 이동 (solution space를 강제로 2로 지정)
movejx(P1, vel=[10, 20, 30, 40, 50, 60], acc=[20, 20, 30, 30, 40, 40], radius=100,
sol=2)
# TCP끝단이 P1위치일 때의 관절각으로 이동시키며 P1 위치로부터 100mm의
# 거리가 될 때 다음 모션을 수행하도록 설정
movejx(P2, v=30, a=60, ra= DR_MV_RA_OVERRIDE, sol=2)
# 직전모션을 즉시 종료시키며 Blending하여 TCP끝단이 P2위치일 때의
# 관절각으로 이동
```

- **관련 명령어**

posx()/set_velj()/set_accj()/get_current_posx()/amovejx()

8.18 movec

▪ 기능

작업공간(task space)을 기준으로 로봇이 현재 위치에서 경유점(pos1)를 지나 목표위치(pos2)까지의 원호 또는 지정한 각도까지 원호를 따라 이동합니다.

▪ 인수

인수명	자료형	기본값	설명
pos	posj list (float[6])	-	posx 또는 position list
pos2	posx list (float[6]		posx 또는 position list
vel (v)	float list (float[2])	None None	velocity 또는 velocity1, velocity2
acc (a)	float list (float[2])	None None	acceleration 또는 acceleration1, acceleration2
time (t)	float	None	도달 시간 [sec]
radius (r)	float	None	blending시 radius
ref	int	None	reference coordinate ・ DR_BASE: base coordinate ・ DR_TOOL: tool coordinate ・ user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 ・ DR_MV_MOD_ABS: 절대 ・ DR_MV_MOD_REL: 상대
angle (an)	float list (float[2])	None	angle 또는 angle1, angle2
ra	int	DR_MV_RA_DUPLICATE	Reactive motion mode ・ DR_MV_RA_DUPLICATE: duplicate ・ DR_MV_RA_OVERRIDE: override

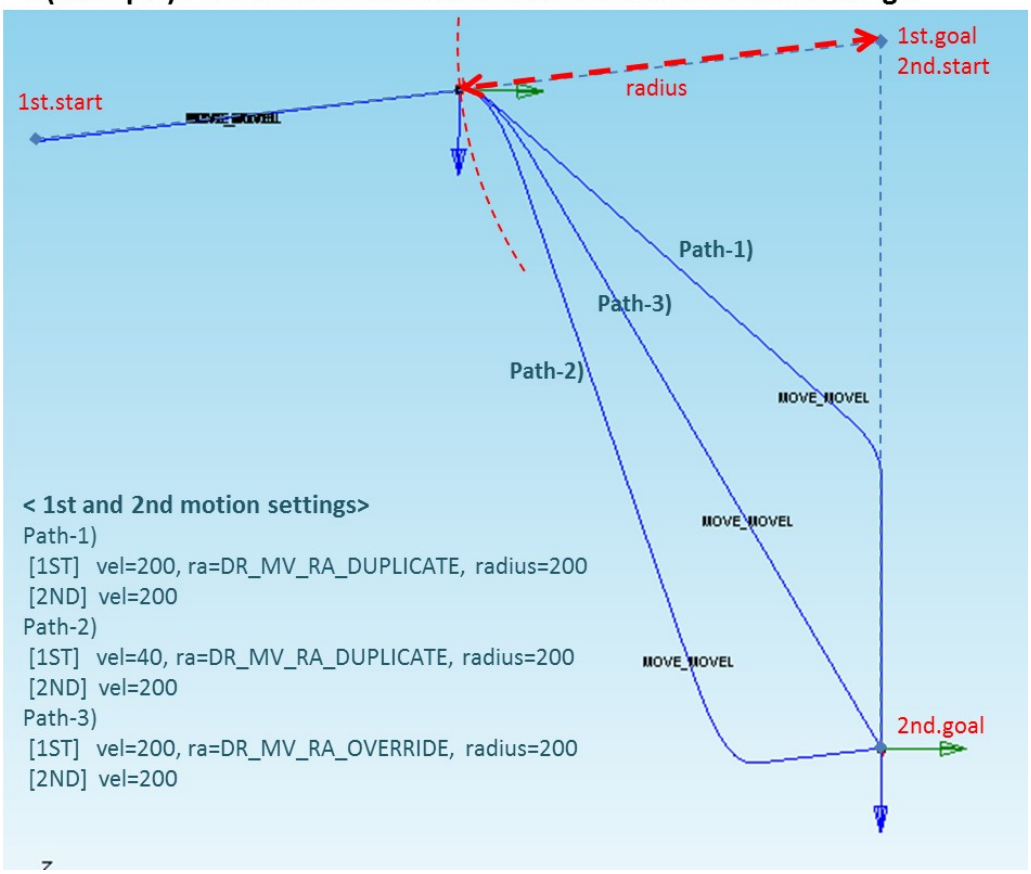
알아두기

- 단축 인수를 지원합니다. (v:vel, a:acc, t:time, r:radius, angle:an)
- vel 이 None 인 경우 _global_velx 가 적용됩니다. (_global_velx 초깃값은 0.0 이며, set_velx 에 의해 설정 가능)
- acc 가 None 인 경우 _global_accx 가 적용됩니다. (_global_accx 초깃값은 0.0 이며, set_accx 에 의해 설정 가능)
- vel 에 하나의 인자를 입력한 경우(예를들어, vel=30) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- acc 에 하나의 인자를 입력한 경우(예를들어, acc=60) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우 0 으로 처리됩니다.
- radius 가 None 이고 블렌딩 구간인 경우는 blending radius 로 처리되며 아닌 경우는 0 으로 처리됩니다.
- ref 가 None 인 경우 _g_coord 가 적용됩니다. (_g_coord 초깃값은 DR_BASE 이며, set_ref_coord 명령에 의해 설정 가능)
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- mod 가 DR_MV_MOD_REL 인 경우 pos1 과 pos2 는 각각 앞 선 pos 에 대한 상대좌표로 정의됩니다. (pos1 은 시작점 대비 상대좌표, pos2 는 pos1 대비 상대좌표)
- angle 이 None 일 경우 0 으로 처리됩니다.
- angle 이 한 개만 입력된 경우 angle 은 Circular path 상의 총 회전각이 적용됩니다.
- angle 이 두 개가 입력된 경우, angle1 은 circular path 상에서 정속으로 이동하는 총 회전각을, angle2 는 가속과 감속을 위한 회전 구간의 회전각을 의미합니다. 이때 총 이동각은 angle1 + 2 X angle2 만큼 circular path 상을 움직입니다.

주의

ra=DR_MV_RA_DUPLICATE 및 radius>0 조건으로 후속 모션이 블렌딩 될 경우 선행모션의 잔여거리, 속도, 가속도로 결정되는 잔여모션시간이 후행모션의 모션시간보다 큰 경우 후행모션이 먼저 종료된 후 선행모션이 종료될 수 있습니다. 관련한 사항은 아래 이미지를 잠고하십시오.

< (Example) Path differences accord. to 1st and 2nd motion settings>



▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
#1
P0 = posj(0,0,90,0,90,0)
movej(P0)
set_velx(30,20)  # 전역 태스크 속도를 30(mm/sec), 20(deg/sec)로 설정
set_accx(60,40)  # 전역 태스크 가속도를 60(mm/sec2), 40(deg/sec2)로 설정

P1 = posx(400,500,800,0,180,0)
P2 = posx(400,500,500,0,180,0)
P3 = posx(100, 300, 700, 45, 0, 0)
P4 = posx(500, 400, 800, 45, 45, 0)

movec(P1, P2, vel=30)
# 속도 30(mm/sec), 전역가속도 60(mm/sec2)로 P1을 경유하여 P2에 이르는
# 원호궤적을 따라 이동
movej(P0)
movec(P3, P4, vel=30, acc=60)
# 속도 30(mm/sec), 가속도 60(mm/sec2)로 P3를 경유하여 P4에 이르는
# 원호궤적을 따라 이동
movej(P0)
movec(P2, P1, time=5)
# 전역(가)속도 30(mm/sec), 60(mm/sec2)로 P2를 경유하여 시점 5초에
# P1에 이르는 원호궤적을 이동
movec(P3, P4, time=3, radius=100)
# P3를 경유하여 P4로 이동하는 원호궤적을 3초의 도착시간을 가지고
# 이동시키며 P4 위치로부터 100mm의 거리가 될 때 다음 모션을 수행하도록
# 설정
movec(P2, P1, ra=DR_MV_RA_OVERRIDE)
# 직전모션을 즉시 종료시키며 Blending하여 P1위치로 이동
```

- **관련 명령어**

posx()/set_velx()/set_accx()/set_tcp()/set_ref_coord()/amovec()

8.19 movesj

▪ 기능

현재 위치에서 pos_list로 입력된 관절공간(joint space)의 경유점들을 거쳐 목표위치(pos_list의 마지막 경유점)까지 연결되는 스플라인 곡선경로를 따라 이동합니다.

입력된 속도/가속도는 경로 중 최대 속도/가속도를 의미하며 입력되는 경유점의 위치에 따라 모션 중의 감속, 가속이 결정됩니다.

▪ 인수

인수명	자료형	기본값	설명
pos_list	list (posj)	-	posj list
vel (v)	float list (float[6])	None	velocity(모든 축에 동일) 또는 velocity(축별 velocity)
acc (a)	float list (float[6])	None	acceleration(모든 축에 동일) 또는 acceleration(축별 acceleration)
time (t)	float	None	도달 시간 [sec]
mod	int	DR_MV_MOD_ABS	이동 기준 • DR_MV_MOD_ABS : 절대 • DR_MV_MOD_REL : 상대

알아두기

- 단축 인수를 지원합니다. (v:vel, a:acc, t:time)
- vel 이 None 인 경우 _global_velj 가 적용됩니다. (_global_velj 초깃값은 0.0 이며, set_velj 에 의해 설정 가능)
- acc 이 None 인 경우 _global_accj 가 적용됩니다. (_global_accj 초깃값은 0.0 이며, set_accj 에 의해 설정 가능)
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우 0 으로 처리됩니다.
- mod 가 DR_MV_MOD_REL 인 경우 pos_list 의 각 pos 는 앞 선 pos 에 대한 상대좌표로 정의됩니다. (pos_list=[q1, q2, ...,q(n-1), q(n)]로 이루어질 때 q1 은 시작점 대비 상대각도, q(n)은 q(n-1) 대비 상대좌표)
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
#CASE 1) 절대각도 입력 (mod= DR_MV_MOD_ABS)
q0 = posj(0,0,0,0,0,0)
movej(q0, vel=30, acc=60)  # 초기위치(q0)로 joint모션 이동
q1 = posj(10, -10, 20, -30, 10, 20)  # posj 변수(관절각) q1 정의
q2 = posj(25, 0, 10, -50, 20, 40)
q3 = posj(50, 50, 50, 50, 50, 50)
q4 = posj(30, 10, 30, -20, 10, 60)
q5 = posj(20, 20, 40, 20, 0, 90)

qlist = [q1, q2, q3, q4, q5] # q1~q5를 경유점 집합으로 하는 리스트(qlist) 정의

movesj(qlist, vel=30, acc=100)
# qlist에 정의된 경유점 집합을 연결하는 스플라인 곡선을 최대속도
# 30(mm/sec), 최대가속도 100(mm/sec2)로 움직임

#CASE 2) 상대각도 입력 (mod= DR_MV_MOD_REL)
q0 = posj(0,0,0,0,0,0)
```

```
movej(q0, vel=30, acc=60)          # 초기위치(q0)로 joint모션 이동
dq1 = posj(10, -10, 20, -30, 10, 20)    # q0에 대한 상대관절각 dq1 정의
(q1=q0+dq1)
dq2 = posj(15, 10, -10, -20, 10, 20)    # q1에 대한 상대관절각 dq2 정의
(q2=q1+dq2)
dq3 = posj(25, 50, 40, 100, 30, 10)    # q2에 대한 상대관절각 dq3 정의
(q3=q2+dq3)
dq4 = posj(-20, -40, -20, -70, -40, 10) # q3에 대한 상대관절각 dq4 정의
(q4=q3+dq4)
dq5 = posj(-10, 10, 10, 40, -10, 30)    # q4에 대한 상대관절각 dq5 정의
(q5=q4+dq5)

dqlist = [dq1, dq2, dq3, dq4, dq5]
 # dq1~dq5를 상대경유점 집합으로 하는 리스트(dqlist ) 정의

movesj(dqlist, vel=30, acc=100, mod= DR_MV_MOD_REL )
# dqlist에 정의된 상대경유점 집합을 연결하는 스플라인 곡선을 최대속도
# 30(mm/sec), 최대가속도 100(mm/sec2)로 움직임 (CASE-1과 동일한 모션)
```

- **관련 명령어**

posj()/set_velj()/set_accj()/amovesj()

8.20 movesx

▪ 기능

로봇이 현재 위치에서 pos_list로 입력된 작업공간(task space)의 경유점들을 거쳐 목표위치(pos_list의 마지막 경유점)까지 연결되는 스플라인 곡선경로를 따라 이동합니다.

입력된 속도/가속도는 경로 중 최대 속도/가속도이며 정속모션 옵션을 선택할 경우 조건에 따라 입력한 속도로 정속도의 모션을 수행합니다.

▪ 인수

인수명	자료형	기본값	설명
pos_list	list (posx)	-	posx list
vel (v)	float list (float[2])	None	velocity 또는 velocity1, velocity2
acc (a)	float list (float[2])	None	acceleration 또는 acceleration1, acceleration2
time (t)	float	None	도달 시간 [sec]
ref	int	None	reference coordinate ・ DR_BASE: base coordinate ・ DR_TOOL: tool coordinate ・ user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 ・ DR_MV_MOD_ABS: 절대 ・ DR_MV_MOD_REL: 상대
vel_opt	int	DR_MVS_VEL_NONE	속도 옵션 ・ DR_MVS_VEL_NONE: 없음 ・ DR_MVS_VEL_CONST: 등속

알아두기

- 단축 인수를 지원합니다. (v:vel, a:acc, t:time)
- vel 이 None 인 경우 _global_velx 가 적용됩니다. (_global_velx 초깃값은 0.0 이며, set_velx 에 의해 설정 가능)
- acc 이 None 인 경우 _global_accx 가 적용됩니다. (_global_accx 초깃값은 0.0 이며, set_accx 에 의해 설정 가능)
- vel 에 하나의 인자를 입력한 경우(예를들어, vel=30) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- acc 에 하나의 인자를 입력한 경우(예를들어, acc=60) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우 0 으로 처리됩니다.
- ref 가 None 인 경우 _g_coord 가 적용됩니다. (_g_coord 초깃값은 DR_BASE 이며, set_ref_coord 명령에 의해 설정 가능)
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- mod 가 DR_MV_MOD_REL 인 경우 pos_list 의 각 pos 는 앞 선 pos 에 대한 상대좌표로 정의됩니다. (pos_list=[p1, p2, ...,p(n-1), p(n)]로 이루어질 때 p1 은 시작점 대비 상대각도, p(n)은 p(n-1) 대비 상대좌표)
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

⚠ 주의

vel_opt= DR_MVS_VEL_CONST 옵션(등속모션)을 선택할 경우 입력된 경유점 간 거리와 속도 조건에 따라 등속모션을 사용할 수 없을 수 있으며, 이 경우에 변속모션 (vel_opt= DR_MVS_VEL_NONE)으로 자동 전환됩니다.

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

예외	설명
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
#CASE 1) 절대좌표 입력 (mod= DR_MV_MOD_ABS)
P0 = posj(0,0,90,0,90,0)
movej(P0, v=30, a=30)
x0 = posx(600, 43, 500, 0, 180, 0) # posx 변수(공간좌표/자세) x0 정의
movel(x0, vel=100, acc=200) # 초기위치 x0로 line모션
x1 = posx(600, 600, 600, 0, 175, 0) # posx 변수(공간좌표/자세) x1 정의
x2 = posx(600, 750, 600, 0, 175, 0)
x3 = posx(150, 600, 450, 0, 175, 0)
x4 = posx(-300, 300, 300, 0, 175, 0)
x5 = posx(-200, 700, 500, 0, 175, 0)
x6 = posx(600, 600, 400, 0, 175, 0)

xlist = [x1, x2, x3, x4, x5, x6] # x1~x6를 경유점 집합으로 하는 리스트 xlist 정의

movesx(xlist, vel=[100, 30], acc=[200, 60], vel_opt=DR_MVS_VEL_NONE)
 # 현재위치에서 시작하여 xlist에 정의된 경유점 집합을 연결하는 스플라인
 # 곡선을 최대속도 100, 30(mm/sec, deg/sec), 최대가속도 200(mm/sec2),
# 60(deg/sec2)로 움직임
movesx(xlist, vel=[100, 30], acc=[200, 60], time=5, vel_opt=DR_MVS_VEL_CONST)
# 현재위치에서 시작하여 xlist에 정의된 경유점 집합을 연결하는 스플라인
# 곡선을 정속 100, 30(mm/sec, deg/sec)(가감속구간제외)로 움직임

#CASE 2) 상대좌표 입력 (mod= DR_MV_MOD_REL)
P0 = posj(0,0,90,0,90,0)
movej(P0)
x0 = posx(600, 43, 500, 0, 180, 0) # posx 변수(공간좌표/자세) x0 정의
movel(x0, vel=100, acc=200) # 초기위치 x0로 line모션
dx1 = posx(0, 557, 100, 0, -5, 0)
# x0에 대한 상대좌표 dx1 정의(x1=x0기준 dx1의 동차변환)
```

```
dx2 = posx(0, 150, 0, 0, 0, 0)
# x1에 대한 상대좌표 dx2 정의(x2=x1기준 dx2의 동차변환)
dx3 = posx(-450, -150, -150, 0, 0, 0)
 # x2에 대한 상대좌표 dx3 정의(x3=x2기준 dx3의 동차변환)
dx4 = posx(-450, -300, -150, 0, 0, 0)
 # x3에 대한 상대좌표 dx4 정의(x4=x3기준 dx4의 동차변환)
dx5 = posx(100, 400, 200, 0, 0, 0)
 # x4에 대한 상대좌표 dx5 정의(x5=x4기준 dx5의 동차변환)
dx6 = posx(800, -100, -100, 0, 0, 0)
 # x5에 대한 상대좌표 dx6 정의(x6=x5기준 dx6의 동차변환)

dxlist = [dx1, dx2, dx3, dx4, dx5, dx6]
# dx1~dx6를 경유점 집합으로 하는 리스트 dxlist 정의

movesx(dxlist, vel=[100, 30], acc=[200, 60], mod= DR_MV_MOD_REL,
vel_opt=DR_MVS_VEL_NONE)
# 현재위치에서 시작하여 dxlist에 정의된 상대 경유점 집합을 연결하는
# 스플라인 곡선을 최대속도 100(mm/sec), 30(deg/sec),
# 최대가속도 200(mm/sec2), 60(deg/sec2)로 움직임 (CASE-1과 동일한 모션)
```

- **관련 명령어**

posx()/set_velx()/set_accx()/set_tcp()/set_ref_coord()/amovesx()

8.21 moveb

- **기능**

하나 이상의 경로 세그먼트(line 또는 circle)를 인자로 갖는 list를 받아 각 세그먼트를 설정된 radius로 블렌딩하여 등속으로 이동합니다. 여기서 radius는 posb를 통해 설정 가능합니다.

- **인수**

인수명	자료형	기본값	설명
pos_list	list (posb)	-	posb list
vel (v)	float list (float[2])	None	velocity 또는 velocity1, velocity2
acc (a)	float list (float[2])	None	acceleration 또는 acceleration1, acceleration2
time (t)	float	None	도달 시간 [sec] * time 지정 시, vel, acc를 무시하고 time 기준으로 처리
ref	int	None	reference coordinate · DR_BASE: base coordinate · DR_TOOL: tool coordinate · user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 · DR_MV_MOD_ABS: 절대 · DR_MV_MOD_REL: 상대

알아두기

- 단축 인수를 지원합니다. (v:vel, a:acc, t:time)
- posb_list 는 최대 50 개까지 입력할 수 있습니다.
- vel 이 None 인 경우 _global_velx 가 적용됩니다.(_global_velx 초깃값은 0.0 이며, set_velx 에 의해 설정 가능)
- acc 이 None 인 경우 _global_accx 가 적용됩니다.(_global_accx 초깃값은 0.0 이며, set_accx 에 의해 설정 가능)
- vel 에 하나의 인자를 입력한 경우(예를들어, vel=30) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- acc 에 하나의 인자를 입력한 경우(예를들어, acc=60) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우 0 으로 처리됩니다.
- ref 가 None 인 경우 _g_coord 가 적용됩니다. (_g_coord 초기값은 DR_BASE 이며, set_ref_coord 명령에 의해 설정 가능)
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- mo 가 DR_MV_MOD_REL 인 경우 posb_list 의 각 pos 는 앞 선 pos 에 대한 상대좌표로 정의됩니다.

주의

- posb 에서 blending radius 가 0 인 경우, 사용자 입력 오류가 나타납니다.
- 연속된 Line-Line segment 가 같은 방향을 가질 경우 Line 의 중복입력으로 사용자 입력 오류가 나타납니다.
- 블렌딩 구간에서 조건에 따라 급격하게 방향전환이 발생하게 되는 경우 급가속을 방지하기 위해 사용자 입력오류가 나타납니다.
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
# Init Pose @ Jx1
Jx1 = posj(45,0,90,0,90,45)                    #초기 Joint위치
        X0 = posx(370, 420, 650, 0, 180, 0)#초기 Task위치
```

```
# CASE 1) ABSOLUTE
# Absolute Goal Poses
X1 =   posx(370, 670, 650, 0, 180, 0)
X1a = posx(370, 670, 400, 0, 180, 0)
X1a2= posx(370, 545, 400, 0, 180, 0)
X1b = posx(370, 595, 400, 0, 180, 0)
X1b2= posx(370, 670, 400, 0, 180, 0)
X1c = posx(370, 420, 150, 0, 180, 0)
X1c2= posx(370, 545, 150, 0, 180, 0)
X1d = posx(370, 670, 275, 0, 180, 0)
X1d2= posx(370, 795, 150, 0, 180, 0)

seg11 = posb(DR_LINE, X1, radius=20)
seg12 = posb(DR_CIRCLE, X1a, X1a2, radius=20)
seg14 = posb(DR_LINE, X1b2, radius=20)
seg15 = posb(DR_CIRCLE, X1c, X1c2, radius=20)
seg16 = posb(DR_CIRCLE, X1d, X1d2, radius=20)
b_list1 = [seg11, seg12, seg14, seg15, seg16]
        # 마지막경유점(seg16)의 blending radius는 무시됨

movej(Jx1, vel=30, acc=60, mod=DR_MV_MOD_ABS)
        # 초기각도(Jx1)로 Joint모션
movel(X0, vel=150, acc=250, ref=DR_BASE, mod=DR_MV_MOD_ABS)
        #초기위치(X0)로 line모션
```

```
moveb(b_list1, vel=150, acc=250, ref=DR_BASE, mod=DR_MV_MOD_ABS)
        # 현재위치에서 시작하여 seg11(LINE), seg12(CIRCLE), seg14(LINE),
        # seg15(CIRCLE), seg16(CIRCLE)으로 이루어진 궤적을 속도 150(mm/sec)를
        # 유지하며(가감속구간 제외) 움직임 (최종point는 X1d2). 각 segment의 끝점
        # (X1, X1a2, X1b2, X1c2, X1d2)에서 20mm 거리에 도달하면 다음 segment로
        # blending이 시작됨
```

```
# CASE 2) RELATIVE
# Relative Goal Poses
dX1 =   posx(0, 250, 0, 0, 0, 0)
dX1a = posx(0, 0, -150, 0, 0, 0)
dX1a2= posx(0, -125, 0, 0, 0, 0)
dX1b = posx(0, 50, 0, 0, 0, 0)
dX1b2= posx(0, 75, 0, 0, 0, 0)
dX1c = posx(0, -250, -250, 0, 0, 0)
dX1c2= posx(0, 125, 0, 0, 0, 0)
dX1d = posx(0, 125, 125, 0, 0, 0)
dX1d2= posx(0, 125, -125, 0, 0, 0)

dseg11 = posb(DR_LINE, dX1, radius=20)
dseg12 = posb(DR_CIRCLE, dX1a, dX1a2, radius=20)
dseg14 = posb(DR_LINE, dX1b2, radius=20)
dseg15 = posb(DR_CIRCLE, dX1c, dX1c2, radius=20)
dseg16 = posb(DR_CIRCLE, dX1d, dX1d2, radius=20)
db_list1 = [dseg11, dseg12, dseg14, dseg15, dseg16]
        # 마지막경유점(dseg16)의 blending radius는 무시됨

movej(Jx1, vel=30, acc=60, mod=DR_MV_MOD_ABS)
        # 초기각도(Jx1)로 Joint모션
movel(X0, vel=150, acc=250, ref=DR_BASE, mod=DR_MV_MOD_ABS)
        #초기위치(X0)로 line모션
moveb(b_list1, vel=150, acc=250, ref=DR_BASE, mod=DR_MV_MOD_ABS)
        # 현재위치에서 시작하여 상대위치로 정의된 dseg11(LINE), dseg12(CIRCLE),
        # dseg14(LINE), dseg15(CIRCLE), dseg16(CIRCLE)으로 이루어진 궤적을
```

```
# 속도 150(mm/sec)를 유지하며(가감속구간제외) 움직임 (최종point는 X1d2).
# 각 segment의 끝점(X1, X1a2, X1b2, X1c2, X1d2)에서 20mm 거리에
# 도달하면 다음 segment로 blending이 시작됨 (경로는 CASE#1과 동일)
```

- **관련 명령어**

posb()/set_velx()/set_accx()/set_tcp()/set_ref_coord()/amoveb()

8.22 move_spiral

▪ 기능

방사형 방향으로 반경이 증가하며 회전하는 Spiral motion과 축 방향으로 병행하며 이동합니다. 현재 위치에서 ref로 지정한 좌표계 상의 axis 방향에 수직인 평면에서의 나선궤적과 axis 방향으로의 직선궤적을 동시에 따라 이동합니다.

▪ 인수

인수명	자료형	기본값	범위	설명
rev	float	10	rev > 0	총 회전수 [revolution]
rmax	float	10	rmax > 0	spiral 최종 반경 [mm]
lmax	float	0		axis 방향으로 이동하는 거리 [mm]
vel (v)	float	None		velocity
acc (a)	float	None		acceleration
time (t)	float	None	time ≥ 0	총 수행시간 <sec>
axis	int	DR_AXIS_Z	-	axis ・ DR_AXIS_X: x축 ・ DR_AXIS_Y: y축 ・ DR_AXIS_Z: z축
ref	Int	DR_TOOL	-	reference coordinate ・ DR_BASE : base coordinate ・ DR_TOOL : tool coordinate ・ user coordinate : 사용자 정의

알아두기

- 단축 인수를 지원합니다. (v:vel, a:acc, t:time)
- rev 는 spiral 모션의 총 회전수를 의미합니다.
- rmax 는 spiral 모션의 최대 반경을 의미합니다.
- lmax 는 모션 동안 axis 방향으로 병진하는 거리를 의미합니다. 단, 음수인 경우 –axis 방향 병진합니다.
- vel 은 spiral 모션의 이동 속도를 의미합니다.
- vel 이 None 인 경우, _global_velx 의 첫째 값(병진 속도)이 적용됩니다. (_global_velx 초기값은 0.0 이며, set_velx 에 의해 설정 가능)
- acc 는 spiral 모션의 이동 가속도를 의미합니다.
- acc 가 None 인 경우, _global_accx 첫째 값(병진 가속도)이 적용됩니다. (_global_accx 초깃값은 0.0 이며, set_accx 에 의해 설정 가능)
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우 0 으로 처리됩니다.
- axis 는 Spiral 모션이 정의하는 평면에 수직인 축을 정의합니다.
- ref 는 spiral 모션이 정의하는 기준 좌표계를 의미합니다.
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

주의

- 경로 생성 시 Spiral 경로에 의한 회전각 가속도를 연산하여 값이 큰 경우 안정적인 모션을 위하여 에러가 발생나타날 수 있습니다.
  이 경우 vel, acc 또는 time 값을 작게 조정하는 것을 권장합니다.

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

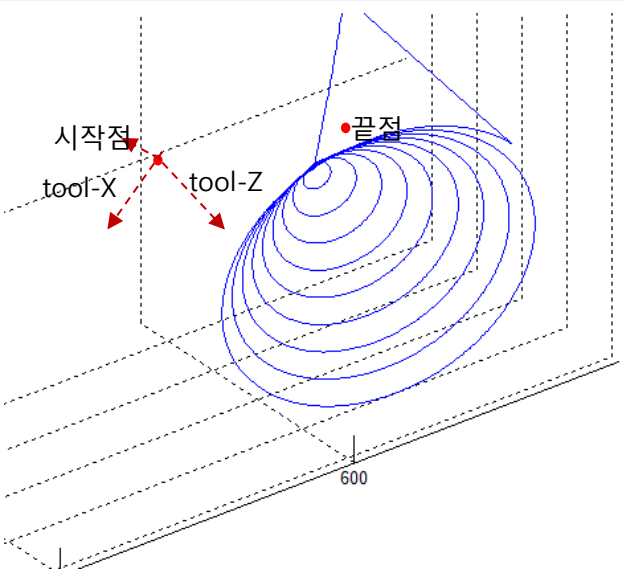
예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

예외	설명
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
# hole search
#(초기 위치로부터 Tool-Z 방향의 회전중심으로, Tool-X/Y 평면의 0에서 20mm 반경(rmax)으로 9.5회전(rev) 함과 동시에 Tool-Z 방향으로 50mm(lmax) 이동하는 spiral 궤적을 20초에 완료하는 모션)

J00 = posj(0,0,90,0,60,0)
movej(J00,vel=30,acc=30) # 초기 자세로 Joint 이동
move_spiral(rev=9.5,rmax=20.0,lmax=50.0,time=20.0,axis=DR_AXIS_Z,ref=DR_TOOL)
```



- **관련 명령어**

set_velx()/set_accx()/set_tcp()/set_ref_coord()/amove_spiral()

8.23 move_periodic

▪ 기능

현재 위치에서 시작하는 상대 모션으로 입력된 기준 좌표계(ref)의 각 축(병진 및 회전)에 대한 Sine 함수 기반으로 주기 모션을 수행합니다. 각 axis 별 모션의 특성은 amp(amplitude)와 period 에 의해 결정되고, 가감속 시간과 총 모션 시간은 주기, 반복, 횟수에 의해 설정됩니다.

▪ 인수

인수명	자료형	기본값	범위	설명
amp	list (float[6])	-	0≤amp	Amplitude(-amp에서 +amp사이 모션) [mm] or [deg]
period	float or list (float[6])		0≤period	period(1주기 소요 시간)[sec]
atime	float	0.0	0≤atime	Acc-, dec- time [sec]
repeat	int	1	> 0	반복 횟수
ref	int	DR_TOOL	-	reference coordinate ・ DR_BASE : base coordinate ・ DR_TOOL : tool coordinate ・ user coordinate : 사용자 정의

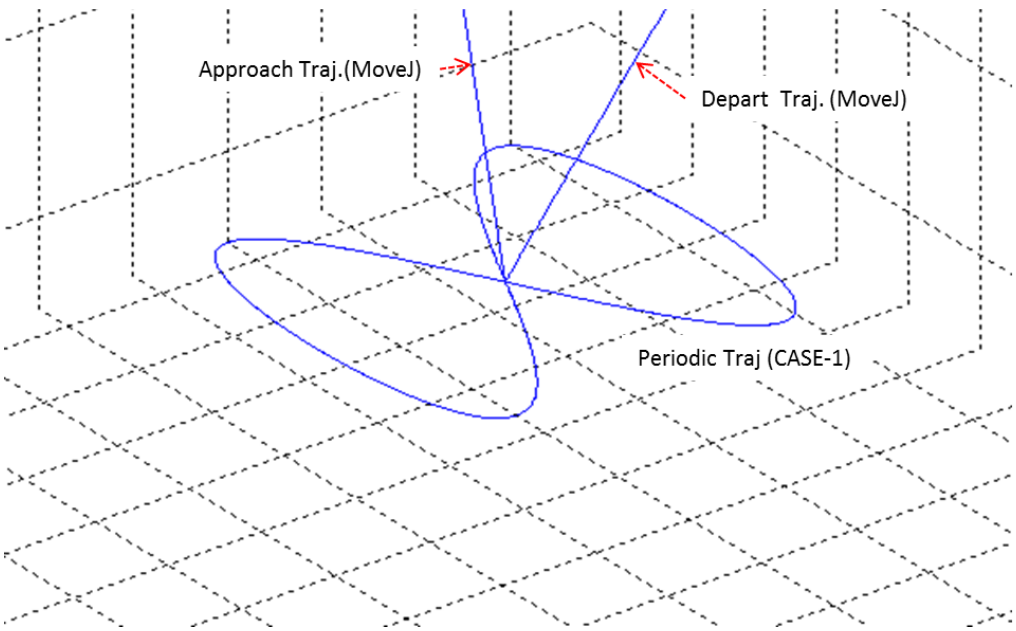
알아두기

- amp 는 진폭(amplitude)을 의미하며, 각 축(x, y, z, rx, ry, rz) 별로 amp 를 값으로 하는 6 개 원소의 list 형태로 입력해야 합니다. 단, 주기 모션을 진행하지 않는 축 방향은 amp 를 0 으로 입력해야 합니다.
- period 는 해당 방향 모션의 1 회 반복 시간을 의미하며, 각 축(x, y, z, rx, ry, rz) 별 period 를 값으로 하는 총 6 개 원소의 list 형태로 입력하거나 대표값을 입력해야합니다.
- atime 은 주기모션의 시작과 끝의 가속 및 감속 시간을 의미합니다. 입력된 가감속시간과 최대주기*1/4 중 큰 값이 적용됩니다. 입력된 가감속 시간이 전체모션시간의 1/2 을 초과하는 경우 에러가 발생합니다.
- repeat 은 가장 큰 period 값을 가지는 축(기준 축)의 반복 횟수를 정의하며, 이에 따라 총 모션 시간이 결정됩니다. 나머지 축의 반복 횟수는 모션 시간에 따라 자동 결정됩니다.

- 모션이 정상 종료되는 경우 종료 위치가 시작 위치와 일치하게 하도록 나머지 축 모션은 기준 축 모션이 종료되기 전에 먼저 종료될 수 있습니다. 모든 축의 모션이 동시 종료되지 않는 경우 감속구간에서의 경로는 이전 경로에서 벗어나게 됩니다. 관련한 사항은 아래 이미지를 참조하십시오

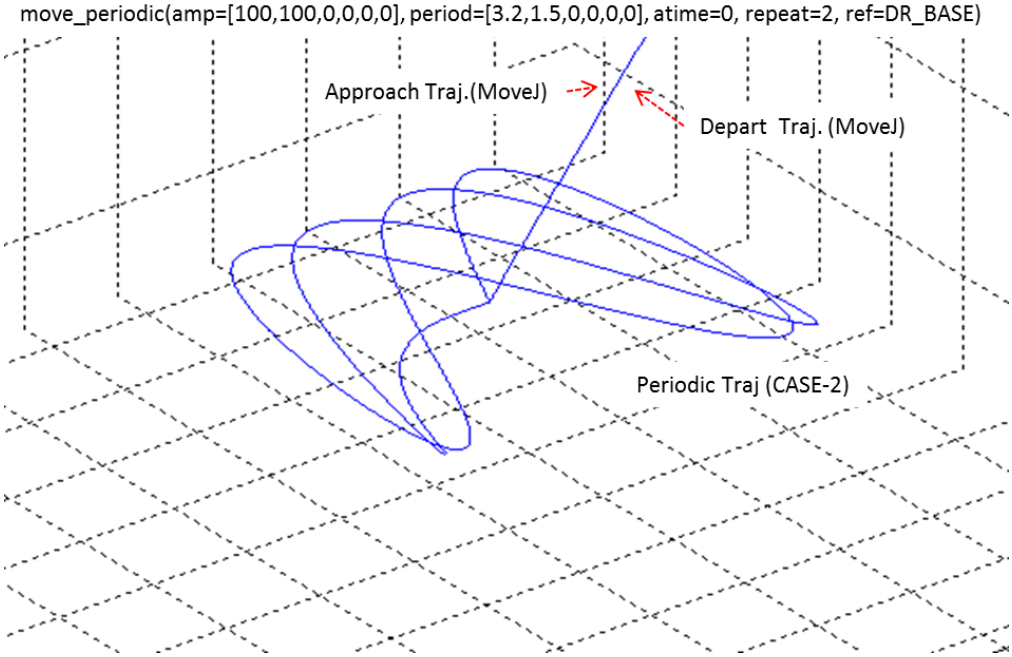
CASE-1) All-axis motions end at the same time

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 는 반복 모션의 기준 좌표계를 의미합니다.
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- 모션명령 수행 시 최대속도 에러가 발생하는 경우 다음의 식을 참조하여 진폭 및 주기를 조정할 것을 제안합니다.
  최대속도=진폭(amp)*2*pi(3.14)/주기(period)
  (예, 진폭=10mm, 주기=1 초인 경우 최대속도=62.83mm/sec)
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

▪ 예제

```
P0 = posj(0,0,90,0,90,0)
movej(P0)

#1
move_periodic(amp =[10,0,0,0,30,0], period=1.0, atime=0.2, repeat=5,
ref=DR_TOOL)
# Tool 좌표계 x축(10mm 진폭, 1초 주기) 모션과 y회전축(진폭 30deg, 1초 주
기)
# 모션이 총 5회 반복 수행

#2
move_periodic(amp =[10,0,20,0,0.5,0], period=[1,0,1.5,0,0,0], atime=0.5,
```

```
        repeat=3, ref=DR_BASE)
        # BASE 좌표계 x축(10mm 진폭, 1초 주기), z축(20mm 진폭, 1.5초 주기) 모션이
        # 총 3회 반복 수행됨, y회전축 모션은 period가 'zero(0)'이므로 미수행
        # z축 모션의 주기가 크므로 총 모션 시간은 약 5.5초(1.5초*3회 + 가감속 1초)
        # 이며, x축은 4.5회 반복 수행
```

- **관련 명령어**

set_ref_coord()/amove_periodic()

8.24 amovej

▪ 기능

비동기(async.)방식의 movej로 블렌딩을 위한 radius 인자를 갖지 않는 점을 제외하고 movej와 동일하게 작동합니다. 그러나 해당 명령어는 async 방식의 모션 명령어로 모션 시작과 동시에 다음 명령어를 수행합니다.

비교)

- movej(pos): 현재 위치에서 출발하여 pos에 도달(정지)한 후에 다음 명령 수행
- amovej(pos): 현재 위치에서 출발하여 pos 도달(정지) 여부와 관계없이 즉시 다음 명령 수행

▪ 인수

인수명	자료형	기본값	설명
pos	posj list (float[6])	-	posj 또는 joint angle list
vel (v)	float list (float[6])	None	velocity(모든 축에 동일) 또는 velocity(축별 velocity)
acc (a)	float list (float[6])	None	acceleration(모든 축에 동일) 또는 acceleration(축별 acceleration)
time (t)	float	None	도달 시간 [sec]
mod	int	DR_MV_MOD_ABS	이동 기준 · DR_MV_MOD_ABS: 절대 · DR_MV_MOD_REL: 상대
ra	int	DR_MV_RA_DUPLICATE	Reactive motion mode · DR_MV_RA_DUPLICATE: duplicate · DR_MV_RA_OVERRIDE: override

알아두기

- 단축 인수를 지원합니다. (v:vel, a:acc, t:time)
- vel 이 None 인 경우, _global_velj 가 적용됩니다. (_global_velj 초깃값은 0.0 이며, set_velj 에 의해 설정 가능)
- acc 이 None 인 경우, _global_accj 가 적용됩니다. (_global_accj 초깃값은 0.0 이며, set_accj 에 의해 설정 가능)
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우 0 으로 처리됩니다.
- 옵션 ra 및 vel/acc 에 따른 blending 시의 경로는 movej() 모션 설명을 참조할 것

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
#예제 1. q0로 모션 시작 3초 후에 q1으로 움직여 모션 정지 후 q99으로 이동
q0 = posj(0, 0, 90, 0, 90, 0)
amovej (q0, vel=10, acc=20)  # q0로 모션 및 즉시 다음 명령 수행
wait(3)      # 3초간 프로그램 일시 중지(모션은 진행 중)
q1 = posj(0, 0, 0, 0, 90, 0)
amovej (q1, vel=10, acc=20)
# q0 모션을 유지(ra 인자 생략 시 DUPLICATE blending)하며 q1으로 중첩
# blending하는 모션 및 즉시 다음 명령 수행
mwait(0)    # 모션이 종료할 때까지 프로그램 일시 중지
q99 = posj(0, 0, 0, 0, 0, 0)
movej (q99, vel=10, acc=20) # q99으로 조인트 모션
```

- **관련 명령어**

posj()/set_velj()/set_accj()/mwait()/movej()

8.25 amovel

▪ 기능

비동기(async.)방식의 movel모션으로 블렌딩을 위한 radius인자를 갖지 않는 점을 제외하고 movel와 동일하게 작동합니다. 그러나 해당 명령어는 async 방식의 모션명령어로 모션 종료를 기다리지 않고 다음 명령어를 수행합니다.

비교)

- movel(pos) : 현재위치에서 출발하여 pos에 도달(정지)한 후에 다음 명령 수행
- amovel(pos) : 현재위치에서 출발하여 pos 도달(정지)여부와 관계없이 즉시 다음 명령 수행

▪ 인수

인수명	자료형	기본값	설명
pos	posx list (float[6])	-	posx 또는 position list
vel (v)	float list (float[2])	None	velocity 또는 velocity1, velocity2
acc (a)	float list (float[2])	None	acceleration 또는 acceleration1, acceleration2
time (t)	float	None	도달 시간 [sec] * time 지정 시, vel, acc를 무시하고 time 기준으로 처리
ref	int	None	reference coordinate · DR_BASE : base coordinate · DR_TOOL : tool coordinate · user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 · DR_MV_MOD_ABS: 절대 · DR_MV_MOD_REL: 상대
ra	int	DR_MV_RA_DUPLICATE	Reactive motion mode · DR_MV_RA_DUPLICATE: duplicate · DR_MV_RA_OVERRIDE: override

알아두기

- 단축 인수 지원(v:vel, a:acc, t:time)
- vel 가 None 인 경우, _global_velx 적용(_global_velx 초깃값은 0.0 이며, set_velx 에 의해 설정 가능)
- acc 가 None 인 경우, _global_accx 적용(_global_accx 초깃값은 0.0 이며, set_accx 에 의해 설정 가능)
- vel 에 하나의 인자를 입력한 경우(예를들어, vel=30) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- acc 에 하나의 인자를 입력한 경우(예를들어, acc=60) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우 0 으로 처리됩니다.
- ref 가 None 인 경우, _g_coord 적용(_g_coord 초기값은 DR_BASE 이며, set_ref_coord 명령에 의해 설정 가능)
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- 옵션 ra 및 vel/acc 에 따른 blending 시의 경로는 movel() 모션 설명을 참조할 것

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

▪ 예제

```
#예제 1. x1으로 모션시작 후 2초 후에 D-Out
j0 = posj(-148,-33,-54,180,92,32)
movej(j0, v=30, a=30)
x1 = posx(784, 543, 570, 0, 180, 0)
amovel (x1, vel=100, acc=200) # x1으로 모션 및 즉시 다음명령 수행
```

```
wait(2)      # 2초간 프로그램 일시중지 (모션은 진행 중)
set_digital_output(1 , 1)   # D-Out(1번채널) ON
mwait(0)     # 모션이 종료할 때까지 프로그램 일시중지
```

- **관련 명령어**

posx()/set_velx()/set_accx()/set_tcp()/set_ref_coord()/mwait()movel()

8.26 amovejx

▪ 기능

비동기(async.)방식의 movejx모션으로 블렌딩을 위한 radius인자를 갖지 않는 점을 제외하고 movejx와 동일하게 작동합니다. 그러나 해당 명령어는 async 방식의 모션명령어로 모션 종료를 기다리지 않고 다음 명령어를 수행합니다.

비교)

- movejx(pos): 현재 위치에서 출발하여 pos에 도달(정지)한 후에 다음 명령 수행
- amovejx(pos): 현재 위치에서 출발하여 pos 도달(정지) 여부와 관계없이 즉시 다음 명령 수행

▪ 인수

인수명	자료형	기본값	설명
pos	posx list (float[6])	-	posx 또는 position list
vel (v)	float list (float[6])	None None	velocity(모든 축에 동일) 또는 velocity(축별 velocity)
acc (a)	float list (float[6])	None None	acceleration(모든 축에 동일) 또는 acceleration(축별 acceleration)
time (t)	float	None	도달 시간 [sec]
ref	int	None	reference coordinate · DR_BASE: base coordinate · DR_TOOL: tool coordinate · user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 · DR_MV_MOD_ABS: 절대 · DR_MV_MOD_REL: 상대
ra	int	DR_MV_RA_DUPLICATE	Reactive motion mode · DR_MV_RA_DUPLICATE: duplicate · DR_MV_RA_OVERRIDE: override
sol	int	0	Solution space

알아두기

- 단축 인수를 지원합니다. (v:vel, a:acc, t:time)
- vel 이 None 인 경우 _global_velj 가 적용됩니다. (_global_velj 초깃값은 0.0 이며, set_velj 에 의해 설정 가능합니다.)
- acc 가 None 인 경우 _global_accj 가 적용됩니다. (_global_accj 초깃값은 0.0 이며, set_accj 에 의해 설정 가능합니다.)
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우 0 으로 처리됩니다.
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우 0 으로 처리됩니다.
- ref 가 None 인 경우 _g_coord 가 적용됩니다. _g_coord 의 초기값은 DR_BASE 이며, set_ref_coord 명령에 의해 설정이 가능합니다.
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- 옵션 ra 와 vel/acc 에 따른 블렌딩 상태의 경로는 movej() 모션 설명을 참조하십시오.

⚠ 주의

상대모션으로 입력하는 경우(mod=DR_MV_MOD_REL), 진행중인 모션에 블렌딩 할 수 없으며 movej() 또는 movel()을 이용하여 블렌딩하는 것을 권장합니다.

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
#예제 1. x1으로 조인트모션 시작 후 2초 후에 D-Out
p0 = posj(-148,-33,-54,180,92,32)
movej(p0, v=30, a=30)
x1 = posx(784, 443, 770, 0, 180, 0)
amovejx (x1, vel=100, acc=200, sol=2)      # x1으로 조인트모션 및 즉시 다음명령 수행
wait(2)              # 2초간 프로그램 일시중지 (모션은 진행 중)
set_digital_output(1 , 1)            # D-Out(1번채널) ON
mwait(0)
```

- **관련 명령어**

posx()/set_velj()/set_accj()/get_current_posx()/mwait()/movejx()

8.27 amovec

▪ 기능

비동기(async.)방식의 movec모션으로 블렌딩을 위한 radius인자를 갖지 않는 점을 제외하고 movec와 동일하게 작동합니다. 그러나 해당 명령어는 async 방식의 모션명령어로서 모션 종료를 기다리지 않고 다음 명령어를 수행합니다.

비교)

- movec(pos1, pos2): 현재위치에서 출발하여 pos2에 도달(정지)한 후에 다음 명령 수행
- amovec(pos1, pos2): 현재위치에서 출발하여 pos2 도달(정지)여부와 관계없이 즉시 다음 명령 수행

▪ 인수

인수명	자료형	기본값	설명
pos	posx list (float[6])	-	posx 또는 position list
pos2	posx list (float[6]	-	posx 또는 position list
vel (v)	float list (float[2])	None	velocity 또는 velocity1, velocity2
acc (a)	float list (float[2])	None	acceleration 또는 acceleration1, acceleration2
time (t)	float	None	도달 시간 [sec]
ref	int	None	reference coordinate ・ DR_BASE: base coordinate ・ DR_TOOL: tool coordinate ・ user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 ・ DR_MV_MOD_ABS: 절대 ・ DR_MV_MOD_REL: 상대
angle (an)	float list (float[2])	None	angle 또는 angle1, angle2
ra	int	DR_MV_RA_DUPLICATE	Reactive motion mode

인수명	자료형	기본값	설명
			• DR_MV_RA_DUPLICATE: duplicate • DR_MV_RA_OVERRIDE: override

알아두기

- 단축 인수를 지원합니다. (v:vel, a:acc, t:time, angle:an)
- vel 이 None 인 경우 _global_velx 가 적용됩니다. (_global_velx 초깃값은 0.0 이며, set_velx 에 의해 설정 가능)
- acc 가 None 인 경우 _global_accx 가 적용됩니다. (_global_accx 초깃값은 0.0 이며, set_accx 에 의해 설정 가능)
- vel 에 하나의 인자를 입력한 경우(예를들어, vel=30) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- acc 에 하나의 인자를 입력한 경우(예를들어, acc=60) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우 0 으로 처리됩니다.
- ref 가 None 인 경우, _g_coord 가 적용됩니다. (_g_coord 초깃값은 DR_BASE 이며, set_ref_coord 명령에 의해 설정 가능)
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- mod 가 DR_MV_MOD_REL 인 경우 pos1 과 pos2 는 각각 앞 선 pos 에 대한 상대좌표로 정의된다. (pos1 은 시작점 대비 상대좌표, pos2 는 pos1 대비 상대좌표)
- angle 이 None 일 경우 0 으로 처리됩니다.
- angle 이 한 개만 입력된 경우, angle 은 Circular path 상의 총 회전각을 적용합니다.
- angle 이 두 개가 입력된 경우, angle1 은 circular path 상에서 정속으로 이동하는 총 회전각을, angle2 는 가속과 감속을 위한 회전 구간의 회전각을 의미합니다. 이때 총 이동각은 angle1+2xangle2 만큼 circular path 상을 움직입니다.
- 옵션 ra 와 vel/acc 에 따른 블렌딩 상태의 경로는 movej() 모션 설명을 참조하십시오.

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

▪ 예제

```
#예제 1. x1으로 조인트모션 시작 후 2초 후에 D-Out
p0 = posj(-148,-33,-54,180,92,32)
movej(p0, v=30, a=30)
x1 = posx(784, 443, 770, 0, 180, 0)
amovejx (x1, vel=100, acc=200, sol=2)      # x1으로 조인트모션 및 즉시 다음명령 수행
wait(2)              # 2초간 프로그램 일시중지 (모션은 진행 중)
set_digital_output(1 , 1)            # D-Out(1번채널) ON
mwait(0)
```

▪ 관련 명령어

posx()/set_velx()/set_accx()/set_tcp()/set_ref_coord()/mwait()/movec()

8.28 amovesj

▪ 기능

비동기(async.)방식의 movesj모션으로 async 처리 외에는 movesj()와 동일하게 동작합니다.

amovesj()에 의한 모션이 종료되기 전에 발생하는 새로운 모션지령은 안전상의 이유로 오류를 발생시킵니다. 따라서 amovej()와 이어지는 모션명령어 사이에는 mwait() 또는 check_motion() 등을 사용하여 amovesj() 모션 종료를 확인한 후 새로운 모션 명령어가 진행되도록 해야 합니다.

비교)

- movesj(pos_list): 현재위치에서 출발하여 pos_list의 끝점에 도달(정지)한 후에 다음 명령 수행합니다.
- amovesj(pos_list): 현재위치에서 출발하여 pos_list의 끝점 도달(정지)여부와 관계없이 즉시 다음 명령을 수행합니다.

▪ 인수

인수명	자료형	기본값	설명
pos_list	list (posj)	-	posj list
vel (v)	float list (float[6])	None	velocity(모든 축에 동일) 또는 velocity(축별 velocity)
acc (a)	float list (float[6])	None	acceleration(모든 축에 동일) 또는 acceleration(축별 acceleration)
time (t)	float	None	도달 시간 [sec]
mod	int	DR_MV_MOD_ABS	이동 기준 ・ DR_MV_MOD_ABS: 절대 ・ DR_MV_MOD_REL: 상대

알아두기

- 단축 인수을 지원합니다. (v:vel, a:acc, t:time)
- vel 이 None 인 경우 _global_velj 가 적용됩니다. (_global_velj 초기값은 0.0 이며, set_velj 에 의해 설정 가능)
- acc 이 None 인 경우 _global_accj 가 적용됩니다. (_global_accj 초기값은 0.0 이며, set_accj 에 의해 설정 가능)
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우 0 으로 처리됩니다.

- mod 가 DR_MV_MOD_REL 인 경우 pos_list 의 각 pos 는 앞 선 pos 에 대한 상대좌표로 정의됩니다. (pos_list=[q1, q2, ...,q(n-1), q(n)]로 이루어질 때 q1 은 시작점 대비 상대각도, q(n)은 q(n-1) 대비 상대좌표)
- 선행모션과 후행모션에 대한 온라인 블랜딩을 지원하지 않습니다.

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
# 예제 1. q1~q5 경유하는 스플라인모션 시작 후 3초 후에 D-Out
q0 = posj(0,0,0,0,0,0)
movej(q0, vel=30, acc=60)                    # 초기위치(q0)로 joint모션 이동
q1 = posj(10, -10, 20, -30, 10, 20)                    # posj 변수(관절각) q1 정의
q2 = posj(25, 0, 10, -50, 20, 40)
q3 = posj(50, 50, 50, 50, 50, 50)
q4 = posj(30, 10, 30, -20, 10, 60)
q5 = posj(20, 20, 40, 20, 0, 90)

qlist = [q1, q2, q3, q4, q5]
        # q1~q5를 경유점 집합으로 하는 리스트(qlist) 정의

amovesj(qlist, vel=30, acc=100)
        # qlist에 정의된 경유점 집합을 연결하는 스플라인 곡선을 최대속도
        # 30(mm/sec), 최대가속도 100(mm/sec2)로 움직임, 모션시작 즉시
        # 다음명령 수행
wait(3)                                      # 3초간 프로그램 일시중지 (모션은 진행
중)
set_digital_output(1 , 1)           # D-Out(1번채널) ON
mwait(0)                            # 모션이 종료할 때까지 프로그램 일시중지
```

- **관련 명령어**

posj()/set_velj()/set_accj()/mwait()/amovesj()

8.29 amovesx

▪ 기능

비동기(async.)방식의 movesx모션으로 async 처리 외에는 movesx()와 동일하게 동작합니다.

amovesx()에 의한 모션이 종료되기 전에 발생하는 새로운 모션지령은 안전상의 이유로 오류를 발생시킵니다. 따라서 amovesx()와 이어지는 모션명령어 사이에는 mwait() 또는 check_motion() 등을 사용하여 amovesx() 모션이 종료를 확인한 후 새로운 모션 명령어가 진행되어야 합니다.

비교)

- movesx(pos_list): 현재위치에서 출발하여 pos_list의 끝점에 도달(정지)한 후에 다음 명령 수행
- amovesx(pos_list): 현재위치에서 출발하여 pos_list의 끝점 도달(정지)여부와 관계없이 즉시 다음 명령 수행

▪ 인수

인수명	자료형	기본값	설명
pos_list	list (posx)	-	posx list
vel (v)	float list (float[2])	None	velocity 또는 velocity1, velocity2
acc (a)	float list (float[2])	None	acceleration 또는 acceleration1, acceleration2
time (t)	float	None	도달 시간 [sec]
ref	int	None	reference coordinate ・ DR_BASE: base coordinate ・ DR_TOOL: tool coordinate ・ user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 ・ DR_MV_MOD_ABS: 절대 ・ DR_MV_MOD_REL: 상대
vel_opt	int	DR_MVS_VEL_NONE	속도 옵션 ・ DR_MVS_VEL_NONE: 없음 ・ DR_MVS_VEL_CONST: 등속

알아두기

- 단축 인수를 지원합니다. (v:vel, a:acc, t:time)
- vel 이 None 인 경우 _global_velx 가 적용됩니다. (_global_velx 초깃값은 0.0 이며, set_velx 에 의해 설정 가능)
- acc 이 None 인 경우 _global_accx 가 적용됩니다. (_global_accx 초깃값은 0.0 이며, set_accx 에 의해 설정 가능)
- vel 에 하나의 인자를 입력한 경우(예를들어, vel=30) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- acc 에 하나의 인자를 입력한 경우(예를들어, acc=60) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우 0 으로 처리됩니다.
- ref 가 None 인 경우, _g_coord 가 적용됩니다. (_g_coord 초깃값은 DR_BASE 이며, set_ref_coord 명령에 의해 설정 가능)
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- mod 가 DR_MV_MOD_REL 인 경우 pos_list 의 각 pos 는 앞 선 pos 에 대한 상대좌표로 정의됩니다. (pos_list=[p1, p2, ...,p(n-1), p(n)]로 이루어질 때 p1 은 시작점 대비 상대각도, p(n)은 p(n-1) 대비 상대좌표)
- 선행모션과 후행모션에 대한 온라인 블랜딩을 지원하지 않습니다.

⚠ 주의

vel_opt= DR_MVS_VEL_CONST 옵션(등속모션) 선택 시, 입력된 경유점 간 거리 및 속도조건에 따라 등속모션이 불가능할 수 있으며 이 경우에 변속모션 (vel_opt= DR_MVS_VEL_NONE)으로 자동 전환됩니다.

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
#예제 1. x1~x6 경유하는 스플라인모션 시작 후 3초 후에 D-Out >
P0 = posj(0,0,90,0,90,0)
movej(P0)
x0 = posx(600, 43, 500, 0, 180, 0)            # posx 변수(공간좌표/자세) x0 정의
movel(x0, vel=100, acc=200)  # 초기위치 x0로 line모션
x1 = posx(600, 600, 600, 0, 175, 0)            # posx 변수(공간좌표/자세) x1 정의
x2 = posx(600, 750, 600, 0, 175, 0)
x3 = posx(150, 600, 450, 0, 175, 0)
x4 = posx(-300, 300, 300, 0, 175, 0)
x5 = posx(-200, 700, 500, 0, 175, 0)
x6 = posx(600, 600, 400, 0, 175, 0)

xlist = [x1, x2, x3, x4, x5, x6] # x1~x6를 경유점 집합으로 하는 리스트 xlist 정의

amovesx(xlist, vel=[100, 30], acc=[200, 60], vel_opt=DR_MVS_VEL_NONE)
 # 현재위치에서 시작하여 xlist에 정의된 경유점 집합을 연결하는 스플라인
 # 곡선을 최대속도 100, 30(mm/sec, deg/sec), 최대가속도 200(mm/sec2),
 # 60(deg/sec2)로 움직임, 모션시작 즉시 다음명령 수행
wait(3)      # 3초간 프로그램 일시중지 (모션은 진행 중)
set_digital_output(1 , 1)  # D-Out(1번채널) ON
mwait(0)    # 모션이 종료할 때까지 프로그램 일시중지
```

- **관련 명령어**

posx()set_velx()/set_accx()/set_tcp()/set_ref_coord()/mwait()/movesx()

8.30 amoveb

▪ 기능

비동기(async.)방식의 moveb모션으로 async 처리 외에는 moveb()와 동일하게 동작하며 명령어를 수행한 후 바로 다음 라인를 수행합니다.

amoveb()에 의한 모션이 종료되기 전에 발생하는 새로운 모션지령은 안전상의 이유로 오류를 발생시킵니다. 따라서 amoveb()와 이어지는 모션명령어 사이에는 mwait() 또는 check_motion() 등을 사용하여 amoveb() 모션 종료를 확인한 후 새로운 모션 명령어가 진행되도록 해야합니다

비교)

- moveb(seg_list): 현재위치에서 출발하여 seg_list의 끝점에 도달(정지)한 후에 다음 명령 수행
- amoveb(seg_list): 현재위치에서 출발하여 seg_list의 끝점 도달(정지)여부와 관계없이 즉시 다음 명령 수행

▪ 인수

인수명	자료형	기본값	설명
pos_list	list (posb)	-	posb list
vel (v)	float list (float[2])	None	velocity 또는 velocity1, velocity2
acc (a)	float list (float[2])	None	acceleration 또는 acceleration1, acceleration2
time (t)	float	None	도달 시간 [sec] * time 지정 시, vel, acc를 무시하고 time 기준으로 처리
ref	int	None	reference coordinate · DR_BASE: base coordinate · DR_TOOL: tool coordinate · user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 · DR_MV_MOD_ABS: 절대 · DR_MV_MOD_REL: 상대

알아두기

- 단축 인수를 지원합니다. (v:vel, a:acc, t:time)
- posb_list 는 최대 50 개까지 입력할 수 있습니다.
- vel 이 None 인 경우 _global_velx 가 적용됩니다. (_global_velx 초깃값은 0.0 이며, set_velx 에 의해 설정 가능)
- acc 이 None 인 경우, _global_accx 가 적용됩니다. (_global_accx 초깃값은 0.0 이며, set_accx 에 의해 설정 가능)
- vel 에 하나의 인자를 입력한 경우(예를들어, vel=30) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 결정됩니다.
- acc 에 하나의 인자를 입력한 경우(예를들어, acc=60) 입력된 인자는 모션의 선가속도에 대응되며, 각가속도는 선가속도에 비례하여 결정됩니다.
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우 0 으로 처리됩니다.
- ref 가 None 인 경우 _g_coord 가 적용됩니다. (_g_coord 초깃값은 DR_BASE 이며, set_ref_coord 명령에 의해 설정 가능)
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- mod 가 DR_MV_MOD_REL 인 경우 posb_list 의 각 pos 는 앞 선 pos 에 대한 상대좌표로 정의됩니다.
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

주의

- posb 에서 blending radius 가 0 인 경우, 사용자 입력 오류가 나타납니다.
- 연속된 Line-Line segment 가 같은 방향을 가질 경우 Line 의 중복입력으로 사용자 입력 오류가 나타납니다.
- 블렌딩 구간에서 조건에 따라 급격하게 방향전환이 발생하게 되는 경우 급가속을 방지하기 위해 사용자 입력오류가 나타납니다.
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
# 예제 1. seg11~seg16의 경로를 따르는 모션 시작 후 3초 후에 D-Out
# Init Pose @ Jx1
Jx1 = posj(45,0,90,0,90,45)                #초기 Joint위치
X0 = posx(370, 420, 650, 0, 180, 0)        #초기 Task위치

# CASE 1) ABSOLUTE
# Absolute Goal Poses
X1 =  posx(370, 670, 650, 0, 180, 0)
X1a = posx(370, 670, 400, 0, 180, 0)
X1a2= posx(370, 545, 400, 0, 180, 0)
X1b = posx(370, 595, 400, 0, 180, 0)
X1b2= posx(370, 670, 400, 0, 180, 0)
X1c = posx(370, 420, 150, 0, 180, 0)
X1c2= posx(370, 545, 150, 0, 180, 0)
X1d = posx(370, 670, 275, 0, 180, 0)
X1d2= posx(370, 795, 150, 0, 180, 0)

seg11 = posb(DR_LINE, X1, radius=20)
seg12 = posb(DR_CIRCLE, X1a, X1a2, radius=20)
seg14 = posb(DR_LINE, X1b2, radius=20)
seg15 = posb(DR_CIRCLE, X1c, X1c2, radius=20)
seg16 = posb(DR_CIRCLE, X1d, X1d2, radius=20)
b_list1 = [seg11, seg12, seg14, seg15, seg16]
      # 마지막경유점(seg16)의 blending radius는 무시됨
movej(Jx1, vel=30, acc=60, mod=DR_MV_MOD_ABS)
      # 초기각도(Jx1)로 Joint모션
movel(X0, vel=150, acc=250, ref=DR_BASE, mod=DR_MV_MOD_ABS)
      #초기위치(X0)로 line모션
amoveb(b_list1, vel=150, acc=250, ref=DR_BASE, mod=DR_MV_MOD_ABS)
```

```
        # 현재위치에서 시작하여 seg11(LINE), seg12(CIRCLE), seg14(LINE),
        # seg15(CIRCLE), seg16(CIRCLE)으로 이루어진 궤적을 속도 150(mm/sec)를
        # 유지하며(가감속구간 제외) 움직임 (최종point는 X1d2).
        # 각 segment의 끝점(X1, X1a2, X1b2, X1c2, X1d2)에서 20mm 거리에 도달하면
        # 다음 segment로 blending이 시작됨
wait(3)                                 # 3초간 프로그램 일시중지 (모션은 진행중)
set_digital_output(1 , 1)           # D-Out(1번채널) ON
mwait(0)                            # 모션이 종료할 때까지 프로그램 일시중지
```

- **관련 명령어**

posb()/set_velx()/set_accx()/set_tcp()/set_ref_coord()/mwait()/moveb()

8.31 amove_spiral

▪ 기능

비동기(async.)방식의 move_spiral모션으로 async 처리 외에는 move_spiral()와 동일하게 동작하며 명령어를 수행한 후 바로 다음 라인을 수행합니다.

amove_spiral()에 의한 모션이 종료되기 전에 발생하는 새로운 모션지령은 안전상의 이유로 오류를 발생시킵니다. 따라서 amove_spiral()과 이어지는 모션명령어 사이에는 mwait() 또는 check_motion() 등을 사용하여 amove_spiral() 모션 종료를 확인한 후 새로운 모션명령이 진행되도록 해야합니다.

본 명령은 방사형 방향으로 반경이 증가하며 회전하는 Spiral motion과 축방향으로 병행하는 모션을 수행합니다. 현재위치에서 기준 좌표계(ref) 지정한 좌표계 상의 axis 방향에 수직인 평면에서의 나선궤적과 axis방향으로의 직선궤적을 동시에 따라 이동합니다.

비교)

- move_spiral: 현재위치에서 출발하여 spiral궤적의 끝에 도달(정지)한 후에 다음 명령 수행
- amove_spiral: 현재위치에서 출발하여 spiral궤적의 끝에 도달(정지)여부와 관계없이 즉시 다음 명령 수행

▪ 인수

인수명	자료형	기본값	범위	설명
rev	float	10	rev > 0	총 회전수 [revolution]
rmax	float	10	rmax > 0	spiral 최종 반경 [mm]
lmax	float	0		axis 방향으로 이동하는 거리 [mm]
vel (v)	float	None		velocity
acc (a)	float	None		acceleration
time (t)	float	None	time ≥ 0	총 수행시간 <sec>
axis	int	DR_AXIS_Z	-	axis ・ DR_AXIS_X: x축 ・ DR_AXIS_Y: y축 ・ DR_AXIS_Z: z축
ref	Int	DR_TOOL	-	reference coordinate ・ DR_BASE : base coordinate ・ DR_TOOL : tool coordinate ・ user coordinate : 사용자 정의

알아두기

- 단축 인수를 지원합니다. (v:vel, a:acc, t:time)
- rev 는 spiral 모션의 총 회전수를 의미합니다.
- rmax 는 spiral 모션의 최대 반경을 의미합니다.
- lmax 는 모션 동안 axis 방향으로 병진하는 거리를 의미합니다. 단, 음수인 경우 –axis 방향 병진합니다.
- vel 은 spiral 모션의 이동 속도를 의미합니다.
- vel 이 None 인 경우, _global_velx 의 첫째 값(병진 속도)이 적용됩니다. (_global_velx 초기값은 0.0 이며, set_velx 에 의해 설정 가능)
- acc 는 spiral 모션의 이동 가속도를 의미합니다.
- acc 가 None 인 경우, _global_accx 첫째 값(병진 가속도)이 적용됩니다. (_global_accx 초깃값은 0.0 이며, set_accx 에 의해 설정 가능)
- time 을 지정할 경우 vel, acc 를 무시하고 time 기준으로 처리됩니다.
- time 이 None 인 경우 0 으로 처리됩니다.
- axis 는 Spiral 모션이 정의하는 평면에 수직인 축을 정의합니다.
- ref 는 spiral 모션이 정의하는 기준 좌표계를 의미합니다.
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

⚠ 주의

- 경로 생성 시 Spiral 경로에 의한 회전각 가속도를 연산하여 값이 큰 경우 안정적인 모션을 위하여 에러가 발생나타날 수 있습니다.
  이 경우 vel, acc 또는 time 값을 작게 조정하는 것을 권장합니다.

▪ 리턴

값	설명
0	성공
음수값	오류

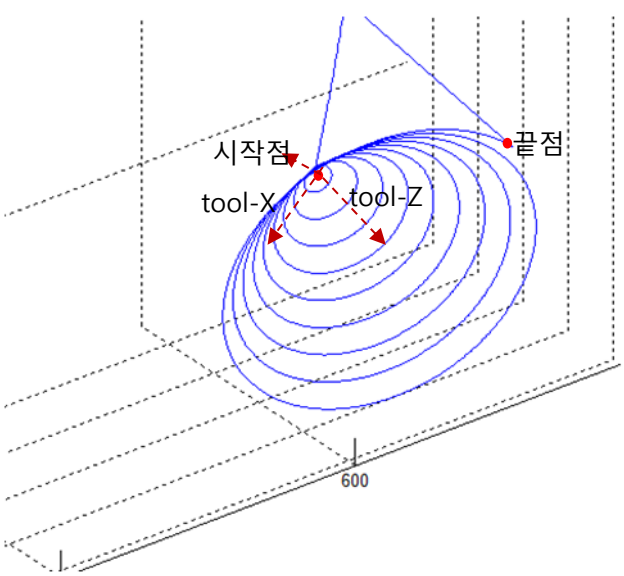
▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
## hole search
# (초기위치로 부터 Tool-Z방향의 회전중심으로 Tool-X/Y평면에서 0에서
# 30mm반경(rmax) 으로 9.5회전(revmax)을 하며 동시에 Tool-Z방향으로
50mm(lmax)
# 이동하는 spiral 궤적을 20초에 완료하는 모션, 모션시작 후 3초 후에
# D-Out(1번채널))

J00 = posj(0,0,90,0,60,0)
movej(J00,vel=30,acc=30)              #초기자세로 Joint이동
amove_spiral(rev=9.5,rmax=30.0,lmax=50.0,time=20.0,axis=DR_AXIS_Z,ref=DR_TOOL)
wait(3)
set_digital_output(1 , 1)   # D-Out(1번채널) ON
mwait(0)    #모션이 멈출때까지 대기
```



- **관련 명령어**

set_velx()/set_accx()/set_tcp()/set_ref_coord()/mwait()/move_spiral()

8.32 amove_periodic

▪ 기능

비동기(async.)방식의 move_periodic모션으로 async 처리 외에 move_periodic()와 동일하게 동작하며 명령어를 수행한 후 바로 다음 라인을 수행합니다.

amove_periodic()에 의한 모션이 종료되기 전에 발생하는 새로운 모션지령은 안전상의 이유로 오류를 발생시킵니다. 따라서 amove_periodic()과 이어지는 모션명령어 사이에는 mwait() 또는 check_motion() 등을 사용하여 amove_periodic() 모션 종료를 확인한 후 새로운 모션명령이 진행되도록 해야합니다.

이 명령어는 현재위치에서 시작하는 상대모션어로 입력된 기준 좌표계(ref)의 각 축(병진 및 회전)에 대한 Sine함수 기반으로 주기모션을 수행합니다. 각 axis별 모션의 특성은 amp(amplitude)와 주기에 의해 결정되고 가감속 시간과 총 모션 시간은 주기, 반복 및 횟수에 의해 설정됩니다.

비교)

- move_ periodic: 현재위치에서 출발하여 periodic 궤적의 끝에 도달(정지)한 후에 다음 명령 수행
- amove_ periodic: 현재위치에서 출발하여 periodic 궤적의 끝에 도달(정지)여부와 관계없이 즉시 다음 명령 수행

▪ 인수

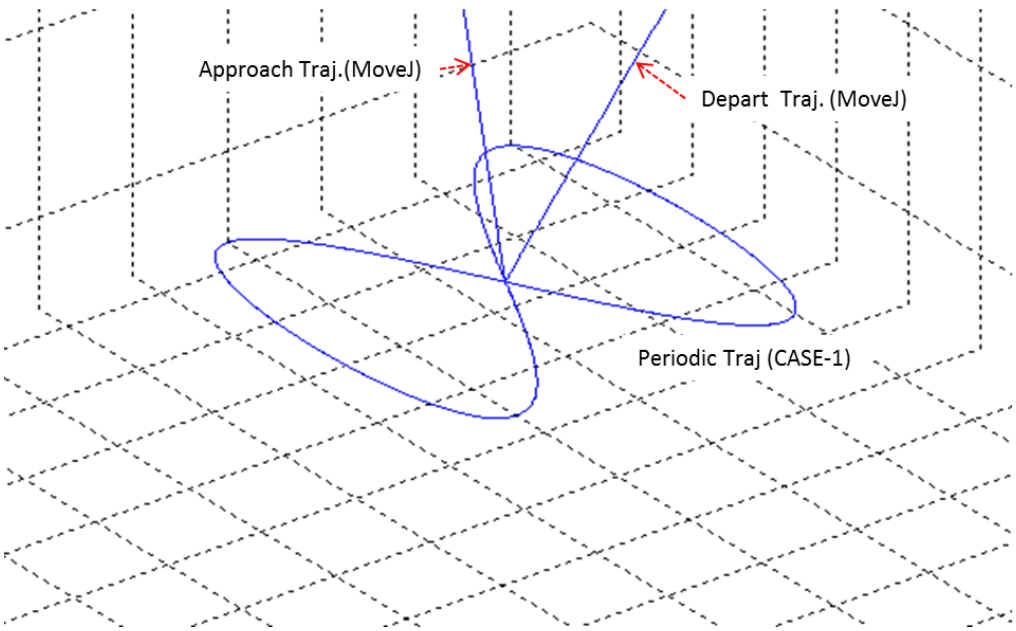
인수명	자료형	기본값	범위	설명
amp	list (float[6])	-	0≤amp	Amplitude(-amp에서 +amp사이 모션) [mm] or [deg]
period	float or list (float[6])		0≤period	period(1주기 소요 시간)[sec]
atime	float	0.0	0≤atime	Acc-, dec- time [sec]
repeat	int	1	> 0	반복 횟수
ref	int	DR_TOOL	-	reference coordinate ・ DR_BASE : base coordinate ・ DR_TOOL : tool coordinate ・ user coordinate : 사용자 정의

알아두기

- amp 는 진폭(amplitude)을 의미하며, 각 축(x, y, z, rx, ry, rz) 별로 amp 를 값으로 하는 6 개 원소의 list 형태로 입력해야 합니다. 단, 주기 모션을 진행하지 않는 축 방향은 amp 를 0 으로 입력해야 합니다.
- period 는 해당 방향 모션의 1 회 반복 시간을 의미하며, 각 축(x, y, z, rx, ry, rz) 별 period 를 값으로 하는 총 6 개 원소의 list 형태로 입력하거나 대표값을 입력해야합니다.
- atime 은 주기모션의 시작과 끝의 가속 및 감속 시간을 의미합니다. 입력된 가감속시간과 최대주기*1/4 중 큰 값이 적용됩니다. 입력된 가감속 시간이 전체모션시간의 1/2 을 초과하는 경우 에러가 발생합니다.
- repeat 은 가장 큰 period 값을 가지는 축(기준 축)의 반복 횟수를 정의하며, 이에 따라 총 모션 시간이 결정됩니다. 나머지 축의 반복 횟수는 모션 시간에 따라 자동 결정됩니다.
- 모션이 정상 종료되는 경우 종료 위치가 시작 위치와 일치하게 하도록 나머지 축 모션은 기준 축 모션이 종료되기 전에 먼저 종료될 수 있습니다. 모든 축의 모션이 동시 종료되지 않는 경우 감속구간에서의 경로는 이전 경로에서 벗어나게 됩니다. 관련한 사항은 아래 이미지를 참조하십시오

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)



- ref 는 반복 모션의 기준 좌표계를 의미합니다.
- **ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.**
- 모션명령 수행 시 최대속도 에러가 발생하는 경우 다음의 식을 참조하여 진폭 및 주기를 조정할 것을 제안합니다.
  최대속도=진폭(amp)*2*pi(3.14)/주기(period)
  (예, 진폭=10mm, 주기=1 초인 경우 최대속도=62.83mm/sec)
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
P0 = posj(0,0,90,0,90,0)
movej(P0)
amove_periodic(amp =[10,0,0,0,0.5,0], period=1, atime=0.5, repeat=5, ref=DR_TOOL)
wait(1)
set_digital_output(1, 1)
mwait(0)
# Tool좌표계 x축(10mm진폭, 1초 주기)모션과 y회전축(진폭 0.5deg, 1초 주기)
# 모션을 총 5회 반복 수행
# periodic 모션을 시작하고 1초 후에 Digital_Output 채널1번을 SET(1) 한다.
```

- **관련 명령어**

set_ref_coord()/move_periodic()

8.33 mwait(time=0)

- **기능**

선행된 모션 명령어와 다음 라인의 모션 명령어 사이의 대기 시간을 설정합니다. 대기 시간은 time[sec]에 입력한 시간에 따라 달라집니다.

- **인수**

인수명	자료형	기본값	설명
time	float	0	모션 끝난 후 대기 시간 [sec]

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
#예제 1. q0로 모션시작 후 3초 후에 q1으로 움직여 모션정지까지 대기하였다가 q99로 이동
q0 = posj(0, 0, 90, 0, 90, 0)
amovej (q0, vel=10, acc=20)     # q0로 모션 및 즉시 다음 명령 수행
wait(3)                                 # 3초간 프로그램 일시 중지(모션은 진행 중)
q1 = posj(0, 0, 0, 0, 90, 0)
amovej (q1, vel=10, acc=20)
```

```
        # q0 모션을 유지(ra 인자 생략 시 DUPLICATE blending)하며 q1으로 중첩
        # blending하는 모션 및 즉시 다음 명령 수행
mwait(0)                        # 모션이 종료할 때까지 프로그램 일시 중지
q99 = posj(0, 0, 0, 0, 0, 0)
movej (q99, vel=10, acc=20)     # q99으로 조인트 모션
```

- **관련 명령어**

wait()amovej()/amovel()/amovejx()/amovec()/amovesj()/amovesx()/amoveb()/amove_spiral()/amove_periodic()

8.34 check_motion()

- **기능**

현재 진행 중인 모션의 상태를 확인합니다.

- **인수**

해당 사항 없음

- **리턴 (TBD)**

값	설명
0	DR_STATE_IDLE (수행중인 모션이 없음)
1	DR_STATE_INIT (모션 연산 중)
2	DR_STATE_BUSY (모션이 수행 중)

- **예외**

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
#1. q0로의 비동기모션 중 모션이 감속을 시작하면 다음모션(q99) 명령 수행
q0 = posj(0, 0, 90, 0, 90, 0)
q99 = posj(0, 0, 0, 0, 0, 0)
amovej (q0, vel=10, acc=20)  # q0로 모션 및 즉시 다음명령 수행
while True:
 if check_motion()==0:  # 모션이 완료 된 경우
  amovej (q99, vel=10, acc=20)  # q99로 조인트 모션
  break
 if check_motion()==2:  # 모션 중인 경우
  pass
mwait(0)   # 모션이 종료할 때까지 프로그램 일시중지
```

- **관련 명령어**

movej()/movel()/movejx()/movec()/movesj()/movesx()/moveb()/move_spiral()/move_periodic()/amovej()/amovel()/amovejx()/amovec()/amovesj()/amovesx()/amoveb()/amove_spiral()/amove_periodic()

8.35 stop(st_mode)

- **기능**

수행 중인 모션을 정지합니다. 인자로 받는 st_mode에 따라 다르게 정지하며 Estop을 제외한 모든 Stop 모드는 현재 수행하고 있는 구간의 모션을 정지합니다.

- **인수**

인수명	자료형	기본값	설명
st_mode	int	-	stop mode • DR_QSTOP_STO: Stop Category 1 • DR_QSTOP: Stop Category 2 • DR_SSTO: Stop Category 2 • DR_HOLD: emergency stop

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
#1. x1으로 이동 시작 2초 후에 Soft Stop으로 모션 종료
p0 = posj(-148,-33,-54,180,92,32)
movej(p0, v=30, a=30)
x1 = posx(784, 543, 570, 0, 180, 0)
amovel (x1, vel=100, acc=200) # x1으로 모션 및 즉시 다음 명령 수행
wait(2)      # 2초간 프로그램 일시 중지
stop(DR_SSTOP)    # Soft Stop하여 모션 정지
```

- **관련 명령어**

movej()/movel()/movejx()/movec()/movesj()/movesx()/moveb()/move_spiral()/move_periodic()/amovej()/amovel()/amovejx()/amovec()/amovesj()/amovesx()/amoveb()/amove_spiral()/amove_periodic()

8.36 change_operation_speed(speed)

- **기능**

작동 속도를 조정합니다. 인자는 현재 설정된 속도에 대한 상대적인 비율을 백분율로 환산한 값으로 1에서 100까지의 값을 갖습니다. 따라서 50은 현재 설정된 속도의 50 Percent로 속도를 줄인다는 의미입니다.

- **인수**

인수명	자료형	기본값	설명
speed	int	-	operation speed(1~100)

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
change_operation_speed(10)
change_operation_speed(100)
#1. q0로 지정 속도 모션 및 지정 속도의 20%로 모션
q0 = posj(0, 0, 90, 0, 90, 0)
movej (q0, vel=10, acc=20)          # q0로 10mm/sec 속도로 이동
change_operation_speed(10)      # 이후 실행되는 모든 모션의 속도는 지정 속도의 10%
q1 = posj(0, 0, 0, 0, 90, 0)
movej (q1, vel=10, acc=20)          # q1으로 1mm/sec 속도(10mm/sec의 10%)로 이동
change_operation_speed(100)    # 이후 실행되는 모든 모션의 속도는 지정속도의 100%
movej (q0, vel=10, acc=20)          # q0로 10mm/sec 속도로 이동(10mm/sec의 100%)
```

- **관련 명령어**

movej()/movel()/movejx()/movec()/movesj()/movesx()/moveb()/move_spiral()/move_periodic()/amovej()/amovel()/amovejx()/amovec()/amovesj()/amovesx()/amoveb()/amove_spiral()/amove_periodic()

8.37 enable_alter_motion(n,mode,ref,limit_dPOS,limit_dPOS_per)

▪ 기능

본 함수는 M2.40 이상의 버전에서만 사용 가능합니다.

경로 수정 기능을 활성화 합니다. 경로 생성의 단위 주기는 100msec이며 입력 인자 n을 설정하여 경로 생성 주기(n*100msec)를 변경 할 수 있습니다. 입력 인자 mode를 통해 alter_motion()의 입력값의 의미를 2가지 모드(누적량 모드, 증분량 모드) 중 하나로 선택하여 사용 할 수 있습니다. 누적량 모드의 경우 현재의 모션경로에 대한 절대적 증분위치/자세만큼 경로 수정량이 반영 됩니다. 증분량 모드의 경우 바로 현재의 절대적 증분위치/자세에 입력된 증분위치/자세만큼 경로 수정량이 추가되어 반영 됩니다. 입력 인자 ref를 통해 기준 좌표계를 설정할 수 있습니다. 입력 인자 limit_dPOS, limit_dPOS_per를 통해 각 각 누적량, 증분량의 한계치를 설정 할 수 있습니다. 한계치를 벗어나는 위치 값의 한계치에 수렴한 값으로 경로 수정량이 재 조정 됩니다.

▪ 인수

인수명	자료형	기본값	설명
n	int	None	경로 생성 주기
mode	Int	None	경로 수정 모드 · DR_DPOS : 누적량 · DR_DVEL : 증분량
ref	int	None	reference coordinate · DR_BASE: base coordinate · DR_WORLD: world coordinate · DR_TOOL: tool coordinate · user coordinate: 사용자 정의
limit_dPOS	list(float[2])	None	첫번째 값 : 이동량 제한 값[mm] 두번째 값 : 회전량 제한 값[deg]
limit_dPOS_per	list(float[2])	None	첫번째 값 : 이동량 제한 값[mm] 두번째 값 : 회전량 제한 값[deg]

알아두기

· alter_motion()은 사용자 thread 내에서만 동작합니다.

- ref 가 None 인 경우, _g_coord 적용(_g_coord 초기값은 DR_BASE 이며, set_ref_coord 명령에 의해 설정 가능)
- limit_dPOS, limit_dPOS_per 를 설정하지 않을 시 누적량, 증분량의 한계를 제한하지 않습니다.

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

▪ 예제

```
def alter_thread():
        alter_motion(dX)

dX = [10,0,0,10,0,0]

J00 = posj(0,0,90,0,90)
X1 = posx(559.0, 200, 651.5, 180, -180.0, 180)
X2 = posx(559.0, 200, 400, 180, -180.0, 180)

movej(J00,vel=50,acc=100)

enable_alter_motion(n=5,mode=DR_DPOS, ref=DR_BASE, limit_dPOS=[50,90],
limit_dPOS_per=[50,50])
# 경로 수정 기능 활성화
# 생성주기:(5*100)msec, 모드:누적량, 기준좌표계:베이스
# 누적량 제한:50mm,90deg, 증분량 제한:50mm, 50deg
```

```
th_id = thread_run(alter_thread, loop=True)

movel(X1,v=50,a=100,r=30)
movel(X2,v=50,a=100)

thread_stop(th_id)
disable_alter_motion() # 경로 수정 기능 비활성화
```

- **관련 명령어**

alter_motion(pos), disable_alter_motion()

8.38 alter_motion([x,y,z,rx,ry,rz])

▪ 기능

본 함수는 M2.40 이상의 버전에서만 사용 가능합니다.

입력 인자 pos에 해당하는 양만큼 경로 수정을 진행합니다.

⚠ 주의

- alter_motion()은 사용자 thread 내에서만 동작합니다.

알아두기

- alter_motion()은 enable_alter_motion()을 통해 경로보정기능이 활성화 된 경우에만 유효합니다.
- enable_alter_motion 의 설정 값 limit_dPOS, limit_dPOS_per 에 따라 경로 수정량은 조정될 수 있습니다.
- pos 의 방향값은 Fixed XYZ 로 설정하여야 됩니다.

▪ 인수

인수명	자료형	기본값	설명
pos	list (float[6])	-	position list

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
def alter_thread():
        alter_motion(dX)

dX = [10,0,0,10,0,0]

J00 = posj(0,0,90,0,90)
X1 = posx(559.0, 200, 651.5, 180, -180.0, 180)
X2 = posx(559.0, 200, 400, 180, -180.0, 180)

movej(J00,vel=50,acc=100)

enable_alter_motion(n=10,mode=DR_DPOS, ref=DR_BASE, limit_dPOS=[50,90],
limit_dPOS_per=[10,10])
# 경로 수정 기능 활성화
# 생성주기:(5*100)msec, 모드:누적량, 기준좌표계:베이스
# 누적량 제한:50mm,50deg, 증분량 제한:10mm, 10deg

th_id = thread_run(alter_thread, loop=True)

movel(X1,v=50,a=100,r=30)
movel(X2,v=50,a=100)

thread_stop(th_id)
disable_alter_motion() # 경로 수정 기능 비활성화
```

- **관련 명령어**

enable_alter_motion(n,mode,ref,limit_dPOS,limit_dPOS_per), disable_alter_motion()

8.39 disable_alter_motion()

- **기능**

본 함수는 M2.40 이상의 버전에서만 사용 가능합니다.

경로 수정 기능을 비활성화 합니다.

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
def alter_thread():
        alter_motion(dX)

dX = [10,0,0,10,0,0]

J00 = posj(0,0,90,0,90)
X1 = posx(559.0, 200, 651.5, 180, -180.0, 180)
X2 = posx(559.0, 200, 400, 180, -180.0, 180)

movej(J00,vel=50,acc=100)

enable_alter_motion(n=10,mode=DR_DPOS, ref=DR_BASE, limit_dPOS=[50,90],
limit_dPOS_per=[50,50])
# 경로 수정 기능 활성화
```

```
# 생성주기:(5*100)msec, 모드:누적량, 기준좌표계:베이스
# 누적량 제한:50mm,50deg, 증분량 제한:10mm, 10deg

th_id = thread_run(alter_thread, loop=True)

movel(X1,v=50,a=100,r=30)
movel(X2,v=50,a=100)

thread_stop(th_id)
disable_alter_motion() # 경로 수정 기능 비활성화
```

- **관련 명령어**

enable_alter_motion(n,mode,ref,limit_dPOS,limit_dPOS_per), alter_motion(pos)

9. 제어 보조 함수

9.1 get_control_mode()

- **기능**

현재 제어 모드를 리턴합니다.

- **인수**

해당 사항 없음

- **리턴**

값	설명
int	제어모드 3 : Position control mode 4 : Torque control mode

- **예외**

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

- **예제**

```
mode = get_control_mode()
```

- **관련 명령어**

해당 사항 없음

9.2 get_control_space()

- **기능**

현재 제어 공간을 리턴합니다.

- **인수**

해당 사항 없음

- **리턴**

값	설명
int	제어모드 1 : Joint space control 2 : Task space control

- **예외**

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

- **예제**

```
x1 = get_control_space()
```

- **관련 명령어**

해당 사항 없음

9.3 get_current_posj()

- **기능**

현재 관절 각도를 리턴합니다.

- **인수**

해당 사항 없음

- **리턴**

값	설명
posj	관절각

- **예외**

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

- **예제**

```
q1 = get_current_posj()
```

9.4 get_current_velj()

- **기능**

현재 관절 속도를 리턴합니다.

- **인수**

해당 사항 없음

- **리턴**

값	설명
float[6]	Joint speed

- **예외**

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

- **예제**

```
velj1 = get_current_velj()
```

- **관련 명령어**

get_desired_velj()

9.5 get_desired_posj()

- **기능**

현재의 목표(target) 관절각을 리턴합니다. 단, movel, movec, movesx, moveb, move_spiral, move_periodic 명령어에서는 사용할 수 없습니다.

- **인수**

해당 사항 없음

- **리턴**

값	설명
posj	관절각

- **예외**

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_INVALID)	유효하지 않은 명령어

- **예제**

```
jp1 = get_desired_posj()
```

- **관련 명령어**

get_current_posj()

9.6 get_desired_velj()

- **기능**

현재의 목표(target) 관절 속도를 리턴합니다. 단, movel, movec, movesx, moveb, move_spiral, move_periodic 명령어에서는 사용할 수 없습니다.

- **인수**

해당 사항 없음

- **리턴**

값	설명
float[6]	목표 관절 속도

- **예외**

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_INVALID)	유효하지 않은 명령어

- **예제**

```
velj1 = get_desired_velj()
```

- **관련 명령어**

get_current_velj()

9.7 get_current_posx(ref)

▪ 기능

현재 태스크 좌표계의 자세와 solution space를 리턴합니다. 이 때 자세는 base coordinate 를 기준으로 합니다.

▪ 인수

인수명	자료형	기본값	설명
Ref	Int	DR_BASE	reference coordinate ・ DR_BASE : base coordinate ・ DR_WORLD : world coordinate ・ user coordinate: 사용자 정의

알아두기

・ ref 생략시 DR_BASE 로 적용됩니다.
・ ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
Posx	Task space point
Int	Solution space (0 ~ 7)

▪ 예외

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

▪ 예제

```
x1, sol = get_current_posx()   #x1 w.r.t. DR_BASE
```

9.8 get_current_tool_flange_posx(ref)

▪ 기능

입력된 기준좌표계(ref)에 해당하는 현재 툴 플랜지 포즈를 리턴합니다. 즉, tcp=(0,0,0,0,0,0)인 위치로 리턴하는 것을 의미합니다.

▪ 인수

인수명	자료형	기본값	설명
ref	Int	DR_BASE	reference coordinate · DR_BASE : base coordinate · DR_WORLD : world coordinate · user coordinate: 사용자 정의

알아두기

- ref 생략시 DR_BASE 로 적용됩니다.
- ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
posx	Tool flange의 pose

▪ 예외

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

▪ 예제

```
x1 = get_current_tool_flange_posx()  #x1 : BASE 좌표계(기본값)에서의 flange포즈
x2 = get_current_tool_flange_posx(DR_BASE)  #x2: BASE 좌표계에서의 flange포즈
x3 = get_current_tool_flange_posx(DR_WORLD) #x3: WORLD좌표계에서의 flange포즈
```

▪ 관련 명령어

해당 사항 없음

9.9 get_current_velx(ref)

▪ 기능

입력된 기준좌표계(ref)에 해당하는 현재 툴 속도를 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
ref	Int	DR_BASE	reference coordinate · DR_BASE : base coordinate · DR_WORLD : world coordinate

알아두기

- ref 생략시 DR_BASE 로 적용됩니다.
- ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
float[6]	Tool velocity

▪ 예외

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

▪ 예제

```
velx1 = get_current_velx() # velx1 : BASE좌표계(기본값)의 속도
velx2 = get_current_velx(DR_BASE) # velx2 (=velx1): BASE좌표계의 속도
velx3 = get_current_velx(DR_WORLD) #velx3 : WORLD좌표계의 속도
```

▪ 관련 명령어

get_desired_velx()

9.10 get_desired_posx(ref)

▪ 기능

현재의 툴의 목표(target) 자세를 리턴합니다. 이때 자세는 ref coordinate 기준으로 합니다.

▪ 인수

인수명	자료형	기본값	설명
ref	Int	DR_BASE	reference coordinate · DR_BASE : base coordinate · DR_WORLD : world coordinate · user coordinate: 사용자 정의

알아두기

- ref 생략시 DR_BASE 로 적용됩니다.
- ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
float[6]	Tool velocity

▪ 예외

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

▪ 예제

```
x1 = get_desired_posx()  #x1 w.r.t. DR_BASE
x2 = posx(100, 0, 0, 0, 0, 0)
x3 = posx(0, 0, 20, 20, 20, 20)
pos = x3
DR_USR1=set_user_cart_coord(x1, x2, x3, pos)
set_ref_coord(DR_USR1)

xa = get_desired_posx(DR_USR1)  #xa w.r.t. DR_USR1
```

```
xb = get_desired_posx(DR_WORLD)  #xb w.r.t. DR_WORLD
```

- **관련 명령어**

get_desired_posx()

9.11 get_desired_velx(ref)

▪ 기능

입력된 기준좌표계(ref)에 해당하는 현재 툴의 목표(target) 속도를 리턴합니다. 단, movej, movejx, movesj 명령어에서는 사용할 수 없습니다.

▪ 인수

인수명	자료형	기본값	설명
ref	Int	DR_BASE	reference coordinate ・ DR_BASE : base coordinate ・ DR_WORLD : world coordinate

알아두기

・ ref 생략시 DR_BASE 로 적용됩니다.
・ ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
float[6]	Tool velocity

▪ 예외

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_INVALID)	유효하지 않은 명령어

▪ 예제

```
vel_x1 = get_desired_velx()  #vel_x1 : BASE좌표계(기본값)에서의 툴의 목표속도
vel_x2 = get_desired_velx(DR_BASE) #vel_x2 : BASE좌표계에서의 툴의 목표속도
vel_x3 = get_desired_velx(DR_WORLD) #vel_x3 : WORLD좌표계에서의 툴의 목표속도
```

▪ 관련 명령어

get_current_velx()

9.12 get_current_solution_space()

- **기능**

현재 solution space 값을 리턴합니다.

- **인수**

해당 사항 없음

- **리턴**

값	설명
int	Solution space (0 ~ 7)

- **예외**

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

- **예제**

```
sol = get_current_solution_space()
```

9.13 get_current_rotm(ref)

▪ 기능

입력된 기준좌표계(ref)에 해당하는 현재 툴의 회전행렬을 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
ref	Int	DR_BASE	reference coordinate · DR_BASE : base coordinate · DR_WORLD : world coordinate

알아두기

- ref 생략시 DR_BASE 로 적용됩니다.
- ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
float[3][3]	Rotation matrix

▪ 예외

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

▪ 예제

```
rotm1 = get_current_rotm(DR_WORLD) #rotm1 : WORLD좌표계 기준 회전행렬(3x3)
        # 결과값은 3X3 matrix로 저장됩니다.
```

$$rotm1 = \begin{bmatrix} rotm1[0][0] & rotm1[0][1] & rotm1[0][2] \\ rotm1[1][0] & rotm1[1][1] & rotm1[1][2] \\ rotm1[2][0] & rotm1[2][1] & rotm1[2][2] \end{bmatrix}$$

9.14 get_joint_torque()

- **기능**

현재 조인트의 센서 토크 값을 리턴합니다.

- **인수**

해당 사항 없음

- **리턴**

값	설명
float[6]	JTS 토크값

- **예외**

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

- **예제**

```
j_trq1 = get_joint_torque()
```

- **관련 명령어**

get_external_torque()/get_tool_force()

9.15 get_external_torque()

- **기능**

현재 각 관절에서 외력에 의해 발생하는 토크 값을 리턴합니다.

- **인수**

해당 사항 없음

- **리턴**

값	설명
float[6]	외력에 의해 발생하는 토크값

- **예외**

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

- **예제**

```
trq_ext=get_external_torque()
```

- **관련 명령어**

get_joint_torque()/get_tool_force()

9.16 get_tool_force(ref)

▪ 기능

입력된 기준좌표계(ref)에서의 현재 툴에 작용하는 외력 값을 리턴합니다. 출력 값의 힘(Force)은 기준좌표계(ref), 모멘트(Moment)는 Tool 좌표계를 기준으로 합니다.

▪ 인수

인수명	자료형	기본값	설명
ref	Int	DR_BASE	reference coordinate ・ DR_BASE : base coordinate ・ DR_WORLD : world coordinate

알아두기

- ref 생략시 DR_BASE 로 적용됩니다.
- ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
float[6]	Tool에 작용하는 외력

▪ 예외

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

▪ 예제

```
force_ext = get_tool_force()
```

▪ 관련 명령어

get_joint_torque()/get_external_torque()

9.17 get_solution_space(pos)

- **기능**

Solution space 값을 구합니다.

- **인수**

인수명	자료형	기본값	설명
pos	posj list (float[6])	-	posj 또는 position list

- **리턴**

값	설명
0 ~ 7	Solution space

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
q1 = posj(0, 0, 0, 0, 0, 0)
sol1 = get_solution_space(q1)
sol2 = get_solution_space([10, 20, 30, 40, 50, 60])
```

- **관련 명령어**

get_current_solution_space()

10. 기타 설정 및 안전 관련 함수

10.1 get_workpiece_weight()

- **기능**

작업물의 무게를 측정하여 리턴합니다.

- **인수**

해당 사항 없음

- **리턴**

값	설명
0 이상의 값	측정 무게 값
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
weight = get_workpiece_weight()
```

- **관련 명령어**

set_workpiece_weight()/reset_workpiece_weight()

10.2 reset_workpiece_weight()

- **기능**

소재의 무게를 측정하기 전 알고리즘의 초기화를 위해 소재의 무게정보를 초기화합니다.

- **인수**

해당 사항 없음

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
reset_workpiece_weight()
```

- **관련 명령어**

set_workpiece_weight()/get_workpiece_weight()

10.3 set_tool(name)

- **기능**

티치팬던트에 등록된 툴 정보를 이름으로 가져옵니다.

- **인수**

인수명	자료형	기본값	설명
name	string	-	티치 팬던트에 등록된 툴 이름

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
set_tool ("tool1")  # TP에서 등록한 "tool1" 의 정보를 호출하여 Tool로 설정한다.
```

- **관련 명령어**

set_tcp()

10.4 set_tool_shape(name)

- **기능**

티치팬던트에 등록된 툴 형상 정보 중에서 입력된 name의 tool 형상을 활성화 합니다.

- **인수**

인수명	자료형	기본값	설명
name	string	-	티치 팬던트에 등록된 툴 shape 이름

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
set_tool_shape("tool_shape1") #TP에서 등록된 "tool_shape1"의 정보를 활성화 한다.
```

- **관련 명령어**

set_tcp()

10.5 set_singularity_handling(mode)

- **기능**

task motion에서 특이점의 영향으로 path deviation이 발생할 경우 대응 정책을 사용자가 선택 할 수 있도록 합니다. mode의 설정은 아래와 같은 설정이 가능 합니다.

- 자동회피 모드(Default) : DR_AVOID
- 경로 우선 : DR_TASK_STOP
- 속도 가변 : DR_VAR_VEL

기본 설정은 자동회피 모드이며, 이 설정의 경우 특이점으로 인한 불안정성을 감소시키지만 path tracking 정확도가 감소합니다. 경로 우선 설정의 경우 singularity 의 영향으로 불안정성이 발생할 가능성이 있는 경우, 감속 후 warning 메시지를 출력하고 해당 Task를 종료합니다. 속도 가변 설정의 경우 특이점으로 인한 불안정을 감소시키면서 path tracking 정확도를 높입니다. 하지만 특이점 구간에서 TCP 속도 변경이 발생합니다.

- **인수**

인수명	자료형	기본값	설명
mode	int	DR_AVOID	DR_AVOID : 자동 회피 모드 DR__TASK_STOP : 감속/ Warning/ Task 종료 DR_VAR_VEL : 속도 가변

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
.P1 = posx(400,500,800,0,180,0)
P2 = posx(400,500,500,0,180,0)
P3 = posx(400,500,200,0,180,0)
set_singularity_handling (DR_AVOID) # 특이점 자동회피 모드
movel(P1, vel=10, acc=20)
set_velx(30)
set_accx(60)
set_singularity_handling(DR_TASK_STOP) # Task 모션 경로 우선
movel(P2)
set_singularity_handling(DR_VAR_VEL) # 특이점 속도 가변 모드
movel(P3)
```

- **관련 명령어**

movel()/movec()/movesx()/moveb()/move_spiral()/amovel()/amovec()/ amovesx()/amoveb()/amove_spiral()

11. 힘/강성 제어 및 기타 사용자 편의 기능

11.1 parallel_axis(x1, x2, x3, axis, ref)

▪ 기능

입력된 기준좌표계(ref) 기준의 3개의 포즈(x1,x2,x3)가 이루는 평면의 normal vector(get_normal(x1, x2, x3) 참조)방향에 Tool좌표계의 지정축(axis)의 방향을 일치시킵니다. 이때 로봇 TCP 위치는 현재 위치를 유지합니다.

▪ 인수

인수명	자료형	기본값	설명
x1	posx list (float[6])	-	posx 또는 position list
x2	posx list (float[6])	-	posx 또는 position list
x3	posx list (float[6])	-	posx 또는 position list
axis	int	-	axis ・ DR_AXIS_X: x축 ・ DR_AXIS_Y: y축 ・ DR_AXIS_Z: z축
ref	int	DR_BASE	reference coordinate ・ DR_BASE: base coordinate ・ DR_WORLD: world coordinate ・ user coordinate: 사용자 정의

알아두기

・ ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
0	성공

값	설명
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
x0 = posx(0, 0, 90, 0, 90, 0)
movej(x0)
x1 = posx(0, 500, 700, 30, 0, 90)
x2 = posx(500, 0, 700, 0, 0, 45)
x3 = posx(300, 100, 500, 45, 0, 45)
parallel_axis(x1, x2, x3, DR_AXIS_X, DR_WORLD)
#WORLD좌표계 기준 x1,x2,x3로 이루어진 평면에 수직인 방향에 툴 X축을 일치
```

- **관련 명령어**

get_normal()/parallel_axis()/align_axis()/align_axis()

11.2 parallel_axis(vect, axis, ref)

▪ 기능

입력된 기준좌표계(ref) 기준의 벡터(vect) 방향에 Tool좌표계의 지정축(axis)의 방향을 일치시킵니다. 이때 로봇 TCP 위치는 현재 위치를 유지합니다.

▪ 인수

인수명	자료형	기본값	설명
vect	list (float[3])	-	vector
axis	int	-	axis ・ DR_AXIS_X: x축 ・ DR_AXIS_Y: y축 ・ DR_AXIS_Z: z축
ref	int	DR_BASE	reference coordinate ・ DR_BASE: base coordinate ・ DR_WORLD: world coordinate ・ user coordinate: 사용자 정의

알아두기

・ ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
x0 = posx(0, 0, 90, 0, 90, 0)
movej(x0)
parallel_axis([1000, 700, 300], DR_AXIS_X, DR_WORLD)
  # WORLD 좌표계 기준 [1000,700,300] vector방향으로 툴의 X축을 일치
```

- **관련 명령어**

parallel_axis()/align_axis()/align_axis()

11.3 align_axis(x1, x2, x3, pos, axis, ref)

▪ 기능

입력된 기준좌표계(ref) 기준의 3개의 포즈(x1,x2,x3)가 이루는 평면의 normal vector(get_normal(x1, x2, x3) 참조)방향에 Tool좌표계의 지정축(axis)의 방향을 일치시킵니다. 이때 로봇 TCP 위치는 pos 위치로 이동합니다.

▪ 인수

인수명	자료형	기본값	설명
x1	posx list (float[6])	-	posx 또는 position list
x2	posx list (float[6])	-	posx 또는 position list
x3	posx list (float[6])	-	posx 또는 position list
pos	posx list (float[6])	-	posx 또는 position list
axis	int	-	axis ・ DR_AXIS_X: x축 ・ DR_AXIS_Y: y축 ・ DR_AXIS_Z: z축
ref	int	DR_BASE	reference coordinate ・ DR_BASE: base coordinate ・ DR_WORLD: world coordinate ・ user coordinate: 사용자 정의

알아두기

・ ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
0	성공

값	설명
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
p0 = posj(0,0,45,0,90,0)
movej(p0, v=30, a=30)

x1 = posx(0, 500, 700, 30, 0, 0)
x2 = posx(500, 0, 700, 0, 0, 0)
x3 = posx(300, 100, 500, 0, 0, 0)
pos = posx(400, 400, 500, 0, 0, 0)
align_axis(x1, x2, x3, pos, DR_AXIS_X, DR_BASE)
#BASE좌표계 기준 x1,x2,x3로 이루어진 평면에 수직인 방향에 툴 X축 방향을, pos에
#위치를 일치
```

- **관련 명령어**

get_normal()/align_axis()/parallel_axis()/parallel_axis()

11.4 align_axis(vect, pos, axis, ref)

▪ 기능

입력된 기준좌표계(ref) 기준의 벡터(vect) 방향에 Tool좌표계의 지정축(axis)의 방향을 일치시킵니다. 이때 로봇 TCP 위치는 pos 위치로 이동시킵니다.

▪ 인수

인수명	자료형	기본값	설명
vect	list (float[3])	-	vector
pos	posx list (float[6])	-	posx 또는 position list
axis	int	-	axis · DR_AXIS_X: x축 · DR_AXIS_Y: y축 · DR_AXIS_Z: z축
Ref	int	DR_BASE	reference coordinate · DR_BASE: base coordinate · DR_WORLD: world coordinate user coordinate: 사용자 정의

알아두기

· ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

예외	설명
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
p0 = posj(0,0,45,0,90,0)
movej(p0, v=30, a=30)

vect = [10,20,30]
pos = posx(100, 500, 700, 45, 0, 0)
align_axis(vect, pos, DR_AXIS_X)
align_axis(vect, pos, DR_AXIS_X, DR_WORLD)
#WORLD좌표계 기준 [10,20,30]벡터 방향에 툴 X축 방향을, pos에 위치를 일치
```

- **관련 명령어**

align_axis()/parallel_axis()/parallel_axis()

11.5 is_done_bolt_tightening(m=0, timeout=0, axis=None)

- **기능**

  툴의 조임 토크를 모니터링하여 주어진 시간 내에 설정된 토크(m)에 도달한 경우는 True를 리턴하고, 주어진 시간을 초과한 경우에는 False를 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
m	float	0	Target torque
timeout	float	0	Monitoring duration [sec]
axis	int	-	axis ・ DR_AXIS_X: x축 ・ DR_AXIS_Y: y축 ・ DR_AXIS_Z: z축

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
p0 = posj(0,0,90,0,90,0)
movej(p0, v=30, a=30)

task_compliance_ctrl()
xd = posx(559, 34.5, 651.5, 0, 180.0, 60)
amovel(xd, vel=50, acc=50) # 볼트 조이는 모션

res = is_done_bolt_tightening(10, 5, DR_AXIS_Z)
    # 5초 내에 10Nm을 조임 토크에 도달한 경우는 True,
    # 그렇지 않은 경우는 False를 Return 하십시오.
if res==True:
  # some action comes here for the case that bolt tightening is done
  x=1
else:
  # some action comes here for the case that it fails
  x=2
```

- **관련 명령어**

amovel()

11.6 release_compliance_ctrl()

- **기능**

Compliance control을 종료하고 현재 위치에서 위치 제어를 시작합니다.

- **인수**

해당 사항 없음

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
P0 = posj(0,0,90,0,90,0)
movej(P0)
task_compliance_ctrl()
set_stiffnessx([100, 100, 300, 100, 100, 100])
release_compliance_ctrl()
```

- **관련 명령어**

task_compliance_ctrl()/set_stiffnessx()

11.7 task_compliance_ctrl(stx, time)

- **기능**

기존에 설정한 기준 좌표계를 기준으로 태스크 Compliance control을 시작합니다.

- **인수(강성값 TBD)**

인수명	자료형	기본값	설명
stx	float[6]	[3000, 3000, 3000, 200, 200, 200]	Translational 강성 3개, 회전강성 3개
time	float	0	강성변화 시간 [sec] 범위 0~1.0 * 주어진 시간 동안 linear transition

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
P0 = posj(0,0,90,0,90,0)
movej(P0)
task_compliance_ctrl()              # default 강성으로 시작
set_stiffnessx([500, 500, 500, 100, 100, 100], time=0.5)
# 사용자 정의 강성으로 0.5초 동안 전환
release_compliance_ctrl()

task_compliance_ctrl([500, 500, 500, 100, 100, 100])
# 사용자 정의 강성으로 시작
release_compliance_ctrl()
```

- **관련 명령어**

set_stiffnessx()/elease_compliance_ctrl()

11.8 set_stiffnessx(stx, time)

▪ 기능

전역으로 설정된 좌표계(set_ref_coord() 참조) 기준으로 강성값을 설정합니다. 현재 강성 또는 기본값으로부터 STX로 주어진 time값 동안 linear transition 합니다. Translation 강성의 사용자 범위는 0~20000N/m, Rotational 강성의 사용자 범위는 0~400Nm/rad입니다.

▪ 인수

인수명	자료형	기본값	설명
stx	float[6]	[500, 500, 500, 100, 100, 100]	Translational 강성3개, 회전강성 3개
time	float	0	강성변화 시간 [sec] 범위 0~1.0 * 주어진 시간 동안 linear transition

▪ 리턴

값	설명
0	성공
음수값	오류

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

▪ 예제

```
set_ref_coord(DR_WORLD)      # 전역좌표계를 World로 설정
x0 = posx(0, 0, 90, 0, 90, 0)
movej(x0)
task_compliance_ctrl()
```

```
stx = [1, 2, 3, 4, 5, 6]
set_stiffnessx(stx)          # 현재의 전역좌표계(World) 기준 강성 적용
release_compliance_ctrl()
```

- **관련 명령어**

task_compliance_ctrl()/release_compliance_ctrl()

11.9 calc_coord(x1, x2, x3, x4, ref, mod)

▪ 기능

지정한 좌표계(ref) 기준의 최대 4개의 입력점(x1~x4) 및 입력 모드(mod)를 기반으로 새로운 직교 좌표계를 계산할 수 있습니다. 여기서 입력 모드는 입력점의 개수가 2개인 경우에만 유효합니다.
입력점의 개수가 1개인 경우, x1의 위치와 회전으로 좌표계가 계산됩니다.
입력점의 개수가 2개인 경우 입력모드가 0일 때, x1에서 x2로 향하는 벡터가 x방향, x축에 직교하는 평면에 투영된 현재의 Tool-z방향이 사용자 좌표계의 z방향으로 정의되며 x1의 위치가 원점이 되도록 좌표계가 계산됩니다.
입력점의 개수가 2개인 경우 입력모드가 1일 때, x1에서 x2로 향하는 벡터가 x방향, x축에 직교하는 평면에 투영된 x1의 z축방향이 사용자 좌표계의 z방향으로 정의되며 x1의 위치가 원점이 되도록 좌표계가 계산됩니다.
입력점의 개수가 3개인 경우, x1에서 x2로 향하는 벡터가 x방향으로 정의되며, x1에서 x3으로 향하는 벡터를 v라고 하였을 경우 z방향은 오른손법칙에 따라 x방향 벡터 곱하기 v로 정의되며, x1의 위치가 원점이 되도록 좌표계가 계산됩니다.
입력점의 개수가 4개인 경우, 입력점의 개수가 3개인 경우와 축의 방향은 동일하며 원점의 위치가 x4의 위치가 되도록 좌표계가 계산됩니다.

▪ 인수

인수명	자료형	기본값	설명
x1, x2, x3, x4	posx list (float[6])	-	posx 또는 position list
ref	int	-	reference coordinate · DR_BASE: base coordinate · DR_WORLD: world coordinate
mod	int	-	입력 모드 (입력점개수가 2개인 경우에만 유효함) · 0: 현재 Tool-z방향 기준으로 사용자 좌표계의 z방향 정의 · 1: x1의 z방향 기준으로 사용자 좌표계의 z방향 정의

- **리턴**

값	설명
posx	Coordinate 계산 성공 설정된 Coordinate의 위치정보

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
pos1 = posx(500, 30, 500, 0, 0, 0)
pos2 = posx(400, 30, 500, 0, 0, 0)
pos3 = posx(500, 30, 600, 45, 180, 45)
pos4 = posx(500, -30, 600, 0, 180, 0)
pose_user1 = calc_coord(pos1, ref=DR_BASE, mod=0)
pose_user21 = calc_coord(pos1, pos2, ref=DR_WORLD, mod=0)
%% 현재 Tool-z방향 기준으로 사용자 좌표계의 z방향 정의
pose_user22 = calc_coord(pos1, pos2, ref=DR_BASE, mod=1)
%% pos1의 z방향 기준으로 사용자 좌표계의 z방향 정의
pose_user3 = calc_coord(pos1, pos2, pos3, ref=DR_BASE, mod=0)
pose_user4 = calc_coord(pos1, pos2, pos3, pos4, ref=DR_WORLD, mod=0)
ucart1 = set_user_cart_coord(pose_user1, ref=DR_BASE)
ucart2 = set_user_cart_coord(pose_user2, ref=DR_WORLD)
```

- **관련 명령어**

set_user_cart_coord()

11.10 set_user_cart_coord(pos, ref)

▪ 기능

기준 좌표계(ref) 기반의 새로운 사용자좌표계를 설정할 수 있습니다. Workcell Item에서 설정한 좌표계를 포함하여 총 20개의 사용자좌표계를 설정할 수 있으며, 20개가 넘어가면 새로운 직교 좌표계를 설정할 수 없습니다. 명령어를 통해 설정한 사용자좌표계는 프로그램 실행 종료 시 삭제되므로, 사용자좌표계 정보를 유지하려면 Workcell Item에서 사용자좌표계를 설정하세요.

▪ 인수

인수명	자료형	기본값	설명
pos	posx list (float[6])	-	사용자좌표계 정보 (위치 및 방향)
ref	int	-	reference coordinate · DR_BASE: base coordinate · DR_WORLD: world coordinate

알아두기

· ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
양의 정수	Coordinate 설정 성공 설정된 Coordinate ID (101 ~ 120)
-1	Coordinate 설정 실패

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
pos1 = posx(10, 20, 30, 0, 0, 0)
pos2 = posx(30, 50, 70, 45, 180, 45)
user_id1 = set_user_cart_coord(pos1, ref=DR_BASE)
user_id2 = set_user_cart_coord(pos2, ref=DR_WORLD)
```

- **관련 명령어**

set_ref_coord()

11.11 set_user_cart_coord(x1, x2, x3, pos, ref)

▪ 기능

사용자가 입력좌표계(ref) 기준의 포즈 x1, x2, x3를 사용하여 새로운 직교 좌표계를 설정할 수 있습니다. [1]x1x2의 단위 벡터를 ux, x1x2로부터 x3까지 최단거리로 잇는 vector의 단위벡터를 uy로 하여, ux, uy, uz를 각 축의 방향 벡터, 원점은 입력좌표계(ref) 기준의 pos에 위치한 직교 좌표계를 생성합니다. Workcell Item에서 설정한 좌표계를 포함하여 총 20개의 사용자좌표계를 설정할 수 있으며, 20개가 넘어가면 새로운 직교 좌표계를 설정할 수 없습니다. 명령어를 통해 설정한 사용자좌표계는 프로그램 실행 종료 시 삭제되므로, 사용자좌표계 정보를 유지하려면 Workcell Item에서 사용자좌표계를 설정하세요.

[1]M2.0.2 이전 버전에서는 x2x1의 단위 벡터를 ux로 사용

▪ 인수

인수명	자료형	기본값	설명
x1	Posx list (float[6])	-	posx 또는 position list
x2	Posx list (float[6])	-	posx 또는 position list
x3	Posx list (float[6])	-	posx 또는 position list
pos	Posx list (float[6])	-	posx 또는 position list
ref	int	DR_BASE	reference coordinate ・ DR_BASE: base coordinate ・ DR_WORLD: world coordinate

알아두기

・ ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
양의 정수	Coordinate 설정 성공 설정된 Coordinate ID (101 ~ 200)

값	설명
-1	Coordinate 설정 실패

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
x1 = posx(0, 500, 700, 0, 0, 0) # Euler angle은 계산시 무시
x2 = posx(500, 0, 700, 0, 0, 0)
x3 = posx(300, 100, 500, 0, 0)
x4 = posx(300, 110, 510, 0, 0)
pos = posx(10, 20, 30, 0, 0, 0)
user_tc1 = set_user_cart_coord(x1, x2, x3, pos, DR_BASE)
user_tc2 = set_user_cart_coord(x2, x3, x4, pos, DR_WORLD)
```

- **관련 명령어**

set_ref_coord()

11.12 set_user_cart_coord(u1, v1, pos, ref)

▪ 기능

사용자가 입력좌표계(ref) 기준의 벡터 u1과 v1를 사용하여 새로운 직교 좌표계를 설정할 수 있습니다. 직교 좌표계의 원점은 입력좌표계(ref) 기준의 pos에 위치하고, x축/y축 basis는 vector u1과 v1에 주어집니다. 나머지 방향은 u1 x v1에 의해 정해집니다. u1과 v1이 orthogonal 하지 않은 경우, u1과 v1이 span 하는 평면상에 u1과 수직인 v1'를 y축의 방향 vector로 설정합니다. Workcell Item에서 설정한 좌표계를 포함하여 총 20개의 사용자좌표계를 설정할 수 있으며, 20개가 넘어가면 새로운 직교 좌표계를 설정할 수 없습니다. 명령어를 통해 설정한 사용자좌표계는 프로그램 실행 종료 시 삭제되므로, 사용자좌표계 정보를 유지하려면 Workcell Item에서 사용자좌표계를 설정하세요.

▪ 인수

인수명	자료형	기본값	설명
u1	float[3]	-	X축 단위벡터
v1	float[3]	-	y축 단위벡터
pos	posx list (float[6])	-	posx 또는 position list
ref	int	DR_BASE	reference coordinate ・ DR_BASE: base coordinate ・ DR_WORLD: world coordinate

알아두기

・ ref 의 인자에서 DR_WORLD 는 M2.40 이상의 버전에서만 사용 가능합니다.

▪ 리턴

값	설명
양의 정수	Coordinate 설정 성공 설정된 Coordinate ID (101 ~ 200)
-1	Coordinate 설정 실패

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
u1 = [1, 1, 0]
v1 = [-1, 1, 0]
pos = posx(10, 20, 30, 0, 0, 0)
user_tc1 = set_user_cart_coord(u1, v1, pos)
user_tc1 = set_user_cart_coord(u1, v1, pos, DR_WORLD)
```

- **관련 명령어**

set_ref_coord()

11.13 overwrite_user_cart_coord(id, pos, ref)

- **기능**

요청하는 ID(id)의 사용자 좌표계의 좌표계 위치(pos), 기준 좌표계(ref) 정보를 변경합니다.

인수명	자료형	기본값	설명
id	int	-	사용자 좌표계 ID
pos	posx list (float[6])	-	사용자좌표계 정보 (위치 및 방향)
ref	int	DR_BASE	reference coordinate · DR_BASE: base coordinate · DR_WORLD: world coordinate

- **리턴**

값	설명
양의 정수	변경된 Coordinate의 ID, 참조 기준 및 위치정보
-1	Coordinate 계산 실패

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
pose_user1 = posx(30, 40, 50, 0, 0, 0)
id_user = set_user_coord(pose_user1, ref=DR_BASE)
pose_user2 = posx(100, 150, 200, 45, 180, 0)
overwrite_user_coord(id_user, pose_user2, ref=DR_BASE)
```

- **관련 명령어**

set_user_cart_coord()/overwrite_user_cart_coord()

11.14 get_user_cart_coord(id)

- **기능**

해당하는 ID(id)의 사용자 좌표계의 정보인 참조 기준 및 위치정보를 조회합니다.

- **인수**

인수명	자료형	기본값	설명
id	int	-	사용자 좌표계 ID

- **리턴**

값	설명
posx	조회하고자 하는 Coordinate의 위치정보
ref	조회하고자 하는 Coordinate의 부모 좌표계

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
pose_user1 = posx(10, 20, 30, 0, 0, 0)
id_user = set_user_coord(pose_user1, ref=DR_BASE)
pose, ref = get_user_cart_coord(id_user)
```

- **관련 명령어**

set_user_cart_coord()

11.15 set_desired_force(fd, dir, time, mod)

- **기능**

전역으로 설정된 좌표계(set_ref_coord() 참조) 기준으로 힘 제어 목표값, 힘 제어 방향, 힘 목표값, 외력참조모드를 설정합니다.

- **인수**

인수명	자료형	기본값	설명
Fd	float[6]	[0, 0, 0, 0, 0, 0]	(Translational) 힘 성분 3개, (Rotational) 모멘트 성분 3개
dir	int[6]	[0, 0, 0, 0, 0, 0]	1이면 해당 방향 힘 제어 0이면 해당 방향 compliance 제어
time	float	0	힘을 증가시키는데 소요되는 시간 [sec] 범위 0~1.0
mod	int	DR_FC_MOD_ABS	DR_FC_MOD_ABS(0): 힘제어 시 외력을 센서값 그대로(절대적) 참조 DR_FC_MOD_REL(1): 힘제어 초기의 센서값을 기준으로 상대적인 외력만 참조

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

알아두기

- release_force() 명령을 통한 힘제어 종료(강성제어 전환)시 외력값은 센서값을 참조합니다. 따라서 mod=DR_FC_MOD_REL 옵션을 선택한 경우에 참조하는 외력값의 변화가 발생합니다.
- mod 를 DR_FC_MOD_REL 로 설정하더라도 충돌 감지나 툴 무게 추정에서는 센서값을 참조합니다.

⚠ **주의**

- 힘제어 시의 정확도를 확보하기 위해서는 접촉할 대상물에 근접하여 mod=DR_FC_MOD_REL 옵션을 설정하고 힘제어를 시작하며, 또한 힘제어 중 제한된 영역에서 위치/자세를 변화시키는 것을 권장합니다.

▪ 예제

```
set_ref_coord(DR_TOOL)
x0 = posx(0, 0, 90, 0, 90, 0)
movej(x0)
task_compliance_ctrl(stx=[500, 500, 500, 100, 100, 100])
fd = [0, 0, 0, 0, 0, 10]
fctrl_dir= [0, 0, 1, 0, 0, 1]
set_desired_force(fd, dir=fctrl_dir, mod=DR_FC_MOD_REL)
# 전역좌표계(Tool) 기준으로 힘제어 수행
# Tool-z축방향 zero force제어, Tool-z축 모멘트 제어, 나머지축 Compliance 제어
# Relative Force Mode 활성화 (초기외력 기준의 상대적 외력을 참조하여 힘제어)
```

▪ 관련 명령어

release_force()/task_compliance_ctrl()/set_stiffnessx()/release_compliance_ctrl()

11.16 release_force(time=0)

- **기능**

힘 제어 목표값을 time 값 동안 0으로 줄이고 작업 공간을 순응 제어로 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
time	float	0	힘을 감소시키는데 소요되는 시간 범위 0~1.0

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
j0 = posj(0, 0, 90, 0, 90, 0)
x0 = posx(0, 0, 0, 0, 0, 0)
x1 = posx(0, 500, 700, 0, 180, 0)
x2 = posx(300, 100, 700, 0, 180, 0)
x3 = posx(300, 100, 500, 0, 180, 0)
set_velx(100,20)
set_accx(100,20)
movej(j0, vel=10, acc=10)
movel(x2)
task_compliance_ctrl(stx = [500, 500, 500, 100, 100, 100])
fd = [0, 0, 0, 0, 0, 10]
fctrl_dir= [0, 0, 1, 0, 0, 1]
```

```
set_desired_force(fd, dir=fctrl_dir, time=1.0)
movel(x3, v=10)
release_force(0.5)
release_compliance_ctrl()
```

- **관련 명령어**

set_desired_force()/task_compliance_ctrl()/set_stiffnessx()/release_compliance_ctrl()

11.17 check_position_condition(axis, min, max, ref, mod, pos)

- **기능**

주어진 위치 상태를 확인합니다. while 또는 if 조건과 함께 사용하여 해당 조건을 반복 확인할 수 있습니다. axis, pos는 입력좌표계(ref) 기준의 축방향 및 포즈입니다.

- **인수**

인수명	자료형	기본값	설명
axis	int	-	axis ・ DR_AXIS_X: x축 ・ DR_AXIS_Y: y축 ・ DR_AXIS_Z: z축
min	float	DR_COND_NONE	최소값
max	float	DR_COND_NONE	최대값
ref	int	None	reference coordinate ・ DR_BASE : base coordinate ・ DR_WORLD : world coordinate ・ DR_TOOL : tool coordinate ・ user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 ・ DR_MV_MOD_ABS: 절대 ・ DR_MV_MOD_REL: 상대
pos	posx list (float[6])	-	posx 또는 position list

알아두기

・ mod 가 DR_MV_MOD_ABS 인 경우는 절대 위치 기준으로 확인합니다.
・ mod 가 DR_MV_MOD_REL 인 경우는 pos 위치 기준으로 확인합니다.
・ pos 는 mod 가 DR_MV_MOD_REL 인 경우에만 의미가 있습니다.

- **리턴**

값	설명
True	조건이 참
False	조건이 거짓

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
CON1= check_position_condition(DR_AXIS_X, min=-5, max=0, ref=DR_WORLD)
CON2= check_position_condition(DR_AXIS_Y, max=700)
CON3= check_position_condition(DR_AXIS_Z, min=-10, max=-5)     # −10 ≤ z ≤ −5
CON4= check_position_condition(DR_AXIS_Z, min=30)              # 30 ≤ z

CON5= check_position_condition(DR_AXIS_Z,min=-10,max=-5, ref=DR_BASE)
                                                               # −10 ≤ z ≤ −5


  CON6= check_position_condition(DR_AXIS_Z,min=-10,max=-5,
  mod=DR_MV_MOD_ABS)
      # −10 ≤ z ≤ −5

posx1 = posx(400, 500, 800, 0, 180,0)
CON7= check_position_condition(DR_AXIS_Z,min=-10,max=-5,mod =
DR_MV_MOD_REL, pos=posx1)                                      # posx1_(z) − 10 ≤ z ≤
posx1_(z) − 5
```

- **관련 명령어**

check_force_condition()/check_orientation_condition()/set_ref_coord()

11.18 check_force_condition(axis, min, max, ref)

- **기능**

주어진 힘 상태를 확인합니다. 단, 힘의 방향은 고려하지 않고 크기로만 비교합니다. while 또는 if 조건과 함께 사용하여 해당 조건을 반복 확인할 수 있습니다. 힘 측정 시 axis는 입력좌표계(ref) 기준의 축방향 이고 모멘트 측정 시 axis는 툴 좌표계 기준의 축방향 입니다.

- **인수**

인수명	자료형	기본값	설명
axis	int	-	axis · DR_AXIS_X: x축 · DR_AXIS_Y: y축 · DR_AXIS_Z: z축 · DR_AXIS_A: x축 회전 · DR_AXIS_B: y축 회전 · DR_AXIS_C: z축 회전
min	float	DR_COND_NONE	최소값 (min ≥ 0)
max	float	DR_COND_NONE	최대값 (max ≥ 0)
ref	int	None	reference coordinate · DR_BASE : base coordinate · DR_WORLD : world coordinate · DR_TOOL : tool coordinate · user coordinate: 사용자 정의

- **리턴**

값	설명
True	조건이 참
False	조건이 거짓

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
fcon1 = check_force_condition(DR_AXIS_Z, min=5, max=10, DR_WORLD)
# 5 ≤ f_z ≤ 10

while (fcon1):
    fcon2 = check_force_condition(DR_AXIS_C, min=30)                  # 30 ≤ m_z
    pcon1 = check_position_condition(DR_AXIS_X, min=0, max=0.1)   # 0 ≤ x ≤ 0.1

    if (fcon2 and pcon1):
        break
```

- **관련 명령어**

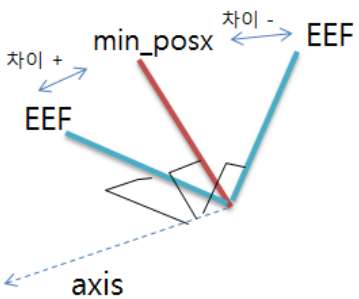
check_position_condition()/check_orientation_condition()/set_ref_coord()

11.19 check_orientation_condition(axis, min, max, ref, mod)

▪ 기능

현재 로봇 엔드이펙터의 자세 정보와 주어진 위치 자세 간 차이의 상태를 확인합니다. 현재 자세와 주어진 자세 간의 차이는 알고리즘 내부에서 회전행렬로 변환되어 “AngleAxis” 기법으로 차이 값(rad 단위)을 리턴합니다. 차이가 + 값이면 true를, - 값이면 false를 리턴합니다. 현재 자세를 기준으로, 주어진 position보다 차이가 +인지 –인지 확인할 때 사용합니다. 사용 예로, 직접교시 position을 이용하여 현재 위치와 차이가 + 방향인지, - 방향인지를 판단한 후 orientation limit에 대한 조건을 만들 수 있습니다. 또한 while 또는 if 조건과 함께 사용하여 해당 조건을 반복 확인할 수 있습니다.

- Min만 설정 시: 차이가 +이면 True, -이면 False
- Min, Max 설정 시: min과의 차이가 -이고, max 차이가 +이면 True, 반대면 False
- Max만 설정 시: max 차이가 -이면 True, +이면 False



▪ 인수

인수명	자료형	기본값	설명
axis	int	-	axis • DR_AXIS_A: x축 회전 • DR_AXIS_B: y축 회전 • DR_AXIS_C: z축 회전
min	posx list (float[6])	-	posx 또는 position list
max	posx list (float[6])	-	posx 또는 position list

인수명	자료형	기본값	설명
ref	int	None	reference coordinate • DR_BASE : base coordinate • DR_WORLD : world coordinate • DR_TOOL : tool coordinate • user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 • DR_MV_MOD_ABS: 절대

- **리턴**

값	설명
True	조건이 참
False	조건이 거짓

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
posx1 = posx(400,500,800,0,180,30)
posx2 = posx(400,500,500,0,180,60)

CON1= check_orientation_condition(DR_AXIS_C, min=posx1, max= posx2)
# 현재 Task 좌표가 posxc = posx(400,500,500,0,180,40)인 경우
# posx1 Rz=30 < posxc Rz=40 < posx2 Rz=60이므로 CON1=True 값이 됩니다.

CON2= check_orientation_condition(DR_AXIS_C, min=posx1)
# 현재 Task 좌표가 posxc = posx(400,500,500,0,180,15)인 경우
# posx1 Rz= 30 > posxc Rz=15이므로 CON2=False 값이 됩니다.
```

```
CON3= check_orientation_condition(DR_AXIS_C, max= posx2)
# 현재 Task 좌표가 posxc = posx(400,500,500,0,180,75)인 경우
# posx1 Rz= 75 > posxc Rz = 60이므로 CON2=False 값이 됩니다.
```

- **관련 명령어**

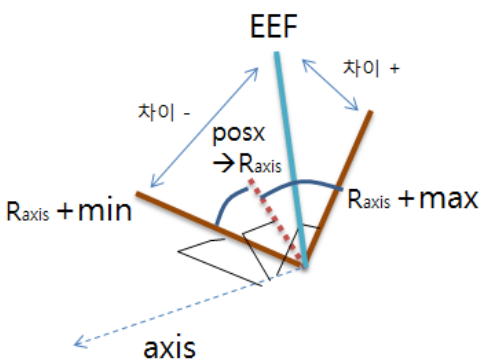
check_position_condition()/check_force_condition()/check_orientation_condition()/set_ref_coord()

11.20 check_orientation_condition(axis, min, max, ref, mod, pos)

▪ 기능

현재 로봇 엔드이펙터의 자세와 회전각 범위 차이에 대한 상태를 확인합니다. 현재 자세와 회전각 범위에 대한 차이는 알고리즘 내부에서 회전행렬로 변환되어 “AngleAxis” 기법으로 차이 값(rad 단위)을 리턴합니다. 차이가 + 값이면 true를, -값이면 false를 리턴합니다. 현재 자세를 기준으로, 주어진 position과 회전각 범위 차이가 +인지 –인지 확인할 때 사용합니다. 사용 예로, 어떤 기준이 되는 position에서 min, max로 회전각 범위를 설정하여, 현재 위치와 차이가 + 방향인지, - 방향인지 판단한 후 orientation limit에 대한 조건을 만들 수 있습니다. 또한 while 또는 if 조건과 함께 사용하여 해당 조건을 반복 확인할 수 있습니다.

- Min만 설정 시: 차이가 +이면 True, -이면 False
- Min, Max 설정 시: min과의 차이가 -이고, max 차이가 +이면 True, 그 반대이면 False
- Max만 설정 시: max 차이가 -이면 True, +이면 False



알아두기

회전각 범위: 주어진 position 에서 설정된 axis 를 기준으로, 상대적인 각도 범위(min, max)를 말합니다. 인자 ref 에 따라 주어진 position 의 기준 좌표계가 정해집니다.

▪ 인수

인수명	자료형	기본값	설명
axis	int	-	axis ・ DR_AXIS_X: x축 회전 ・ DR_AXIS_Y: y축 회전 ・ DR_AXIS_Z: z축 회전

인수명	자료형	기본값	설명
min	float	DR_COND_NONE	최솟값
max	float	DR_COND_NONE	최댓값
ref	int	None	reference coordinate • DR_BASE : base coordinate • DR_WORLD : world coordinate • DR_TOOL : tool coordinate • user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 • DR_MV_MOD_REL: 상대
pos	posx list (float[6])	-	posx 또는 position list

- **리턴**

값	설명
True	조건이 참
False	조건이 거짓

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
posx1 = posx(400,500,800,0,180,15)
CON1= check_orientation_condition(DR_AXIS_C, min=-5, mod=DR_MV_MOD_REL,
pos=posx1, DR_WORLD)
# 현재 Task 좌표가 posxc = posx(400,500,500,0,180,40) 인 경우
# posx1 Rz=15 – (min=5) < posxc Rz=40 이므로 CON1=True 값이 됨.
```

```
CON1= check_orientation_condition(DR_AXIS_C, max=5, mod=DR_MV_MOD_REL,
pos=posx1)
# 현재 Task 좌표가 posxc = posx(400,500,500,0,180,40) 인 경우
# posxc Rz=40 > posx1 Rz=15 + (max=5) 이므로 CON1=False 값이 됨.
```

- **관련 명령어**

check_position_condition()/check_force_condition()/check_orientation_condition()

/set_ref_coord()

11.21 coord_transform(pose_in, ref_in, ref_out)

▪ 기능

'ref_in' 기준 좌표계에서 표현되는 'pose_in' Task 좌표를 'ref_out' 기준 좌표계에서 표현되는 Task 좌표로 변환하여, 출력합니다. 아래의 경우에 대한 좌표변환 계산을 지원 합니다.

- (ref_in) 월드 기준 좌표계 → (ref_out) 월드 기준 좌표계
- (ref_in) 월드 기준 좌표계 → (ref_out) 베이스 기준 좌표계
- (ref_in) 월드 기준 좌표계 → (ref_out) 툴 기준 좌표계
- (ref_in) 월드 기준 좌표계 → (ref_out) 사용자 기준 좌표계
- (ref_in) 베이스 기준 좌표계 → (ref_out) 월드 기준 좌표계
- (ref_in) 베이스 기준 좌표계 → (ref_out) 베이스 기준 좌표계
- (ref_in) 베이스 기준 좌표계 → (ref_out) 툴 기준 좌표계
- (ref_in) 베이스 기준 좌표계 → (ref_out) 사용자 기준 좌표계
- (ref_in) 툴 기준 좌표계 → (ref_out) 월드 기준 좌표계
- (ref_in) 툴 기준 좌표계 → (ref_out) 베이스 기준 좌표계
- (ref_in) 툴 기준 좌표계 → (ref_out) 툴 기준 좌표계
- (ref_in) 툴 기준 좌표계 → (ref_out) 사용자 기준 좌표계
- (ref_in) 사용자 기준 좌표계 → (ref_out) 월드 기준 좌표계
- (ref_in) 사용자 기준 좌표계 → (ref_out) 베이스 기준 좌표계
- (ref_in) 사용자 기준 좌표계 → (ref_out) 툴 기준 좌표계
- (ref_in) 사용자 기준 좌표계 → (ref_out) 사용자 기준 좌표계

▪ 인수

인수명	자료형	기본값	설명
pose_in	posx	-	posx
ref_in	float	DR_COND_NONE	변환 전 reference coordinate · DR_BASE : base coordinate · DR_WORLD : world coordinate · DR_TOOL : tool coordinate · user coordinate: 사용자 정의
ref_out	float	DR_COND_NONE	변환 후 reference coordinate · DR_BASE : base coordinate

인수명	자료형	기본값	설명
			• DR_WORLD : world coordinate • DR_TOOL : tool coordinate • user coordinate: 사용자 정의

- **리턴**

값	설명
pos	posx

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
base_pos = posx(400,500,800,0,180,15)
# 베이스 기준의 Task 좌표가 base_pos = posx(400,500,800,0,180,15) 인 경우

tool_pos = coord_transform(base_pos, DR_BASE, DR_TOOL)
# 베이스 기준의 Task 좌표인 base_pos 를 툴 기준의 Task 좌표로 변환
# 상기 명령어는 변환된 툴 기준의 Task 좌표를 리턴 하며 tool_pos 에 저장
```

- **관련 명령어**

set_user_cart_coord()/get_current_posx()/get_desired_posx()/set_ref_coord()

12. 시스템 함수

12.1 로봇 모드

12.1.1 set_robot_mode(robot_mode)

- **기능**

로봇 제어기의 운용 모드(수동/자동)를 설정하기 위한 함수입니다. 두 모드에서 모두 모션 명령어가 사용 가능하지만, 수동모드에서는 직접교시가 가능하며 안전을 위해 감속 모드로 동작합니다.

자동모드에서는 직접교시가 불가하며 정상 속도 모드로 동작합니다.

수동 모드: 로봇 LED가 푸른 색으로 점등

자동 모드: 로봇 LED가 하얀 색으로 점등

- **인수**

인수명	자료형	기본값	설명
robot_mode	int	-	• ROBOT_MODE_MANUAL : 0 • ROBOT_MODE_AUTONOMOUS : 1

- **리턴**

값	설명
0	성공
음수값	실패

- **예제**

```
#
set_robot_mode(ROBOT_MODE_MANUAL)  #로봇 모드를 수동모드로 전환합니다.
#...
#...
set_robot_mode(ROBOT_MODE_AUTONOMOUS) #로봇 모드를 자동모드로 전환
#...
```

12.1.2 get_robot_mode()

- **기능**

로봇 제어기의 현재 운용 모드를 상태를 읽어오는 함수입니다.

- **인수**

인수명	자료형	기본값	설명
none	-	-	-

- **리턴**

값	설명
ROBOT_MODE_MANUAL (0)	현재 로봇 모드가 수동입니다.
ROBOT_MODE_AUTONOMOUS (1)	현재 로봇 모드가 자동입니다.

- **예제**

```
#...
if get_robot_mode() != ROBOT_MODE_AUTONOMOUS:
    set_robot_mode(ROBOT_MODE_AUTONOMOUS) #로봇 모드를 자동모드로 전환
#...
```

12.1.3 get_last_alarm()

- **기능**

로봇 제어기의 최근 알람 로그를 읽어오는 함수입니다.

- **인수**

없음

- **리턴**

값	설명
level	로그 메시지 레벨 0 : Info 1 : Warning 2 : Alarm
group	로그 메시지 그룹
index	알람 로그 번호
param	알람 로그 파라미터

- **예제**

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

12.1.4 set_safe_stop_reset_type(reset_type)

- **기능**

로봇 제어기의 운용 상태 정보가 SAFE_STOP 일 경우, set_robot_control 함수를 이용하여 상태 전환 후 이후 자동으로 실행되는 일련의 동작을 정의하기 위한 함수입니다. 로봇 운용 모드가 자동일 경우, 프로그램의 재실행 여부를 정의 및 설정할 수 있으며, 수동모드일 경우에는 이 설정은 무시됩니다.

- **인수**

인수명	자료형	기본값	설명
reset_type	int8	0	로봇 리셋 타입 STOP : 1 RESUME : 2

- **리턴**

값	설명
0	성공
1	오류

- **예제**

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

12.1.5 set_robot_speed_mode(speed_mode)

- **기능**

로봇 제어기에서 현재 운용 중인 속도 모드를 설정 및 변경하기 위한 함수입니다.

- **인수**

인수명	자료형	기본값	설명
speed_mode	int8	-	로봇 속도 모드 0 : 정상 모드 1 : 감속 모드

- **리턴**

값	설명
0	오류
1	성공

- **예제**

```
if get_robot_speed_mode() == SPEED_REDUCED_MODE:
    #감속모드 이면 정속 속도 모드로 변경
    set_robot_speed_mode(SPEED_NORMAL_MODE)
```

12.1.6 get_robot_speed_mode()

- **기능**

  로봇 제어기에서 현재 속도 모드(정상 모드, 감속 모드) 정보를 확인하기 위한 함수입니다.

- **인수**

  없음

- **리턴**

값	설명
robot_speed	로봇 속도 모드 0 : 정상 모드 1 : 감속 모드

- **예제**

```
if get_robot_speed_mode() == SPEED_REDUCED_MODE:
   #감속모드 이면 정속 속도 모드로 변경
   set_robot_speed_mode(SPEED_NORMAL_MODE)
```

12.1.7 set_robot_system(robot_system)

- **기능**

로봇 제어기에서 현재 운용 로봇 시스템을 설정 및 변경하기 위한 함수입니다.

- **인수**

인수명	자료형	기본값	설명
robot_system	int8	-	로봇 시스템 정보 0 : Real 1 : Virtual

- **리턴**

값	설명
0	오류
1	성공

- **예제**

```
if(get_robot_system() != ROBOT_SYSTEM_REAL):
    //자동모드 전환
    set_robot_system(ROBOT_SYSTEM_REAL)
else :
  //do somting …
```

12.1.8 get_robot_system()

- **기능**

로봇 제어기에서 현재 운용 로봇 시스템을 읽어오는 함수입니다.

- **인수**

없음

- **리턴**

값	설명
0	Real
1	Virtual

- **예제**

```
if get_robot_system() != ROBOT_SYSTEM_REAL:
    //자동모드 전환
    set_robot_system(ROBOT_SYSTEM_REAL)
else:
  //do somting …
```

12.1.9 get_robot_state()

- **기능**

  로봇 제어기의 현재 운용 상태 정보를 확인하기 위한 입니다.

- **인수**

  없음

- **리턴**

값	설명
robot_state	로봇 상태 정보 0 : 초기화 1 : Standby 2 : 동작 중 3 : SAFE OFF 4 : 직접교시 5 : SAFE STOP 6 : 비상 정지 7 : 호밍 8 : 복구

- **예제**

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

12.2 IO 관련

12.2.1 set_digital_output(index, val =None)

- **기능**

컨트롤러의 디지털 접점에서 신호를 내보내기 위한 명령문입니다. 디지털 출력 레지스터에 저장한 값을 디지털 신호로 출력합니다.

- **인수**

인수명	자료형	기본값	설명
index	int	-	컨트롤러에 장착된 I/O 접점 번호 • val 인자가 있을 경우: 1 ~ 16까지의 숫자 • val 인자가 없을 경우: 1 ~ 16 , -1 ~ -16 (양수는 ON, 음수는 OFF)
val	int	-	I/O value • ON: 1 • OFF: 0

알아두기

val 값을 생략하면, index 인자의 부호에 따라 양수는 ON, 음수는 OFF 가 됩니다.

- **리턴**

값	설명
0	성공
음수값	실패

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
set_digital_output(1, ON)                #1번 접점 ON
set_digital_output(16, OFF)      #16번 접점 OFF
set_digital_output(3)                #3번 접점 ON (val 인자가 생략된 경우 양수 ON)
set_digital_output(-3)               #3번 접점 OFF (val 인자가 생략된 경우 음수 OFF)
```

12.2.2 get_digital_input(index)

- **기능**

컨트롤러의 디지털 입력 점접에서 신호를 불러오기 위한 명령문으로 디지털 입력 점접 값을 읽습니다.

- **인수**

인수명	자료형	기본값	설명
index	int	-	1 ~ 16까지의 숫자이며, 컨트롤러에 장착된 I/O 의 접점 번호

- **리턴**

값	설명
1	ON
0	OFF
음수값	실패

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
in1 = get_digital_input(1)      #1번 접점 읽기
in8 = get_digital_input(8)      #8번 접점 읽기
```

12.2.3 set_tool_digital_output(index, val=None)

- **기능**

로봇 툴의 신호를 디지털 접점에서 내보내기 위한 명령문입니다.

- **인수**

인수명	자료형	기본값	설명
index	int	-	로봇 암에 장착된 I/O 접점 번호 • val 인자가 있을 경우: 1 ~ 6까지의 숫자 • val 인자가 없을 경우: 1 ~ 6 , -1 ~ -6 (양수는 ON, 음수는 OFF)
val	int	-	I/O value : 출력하고자 하는 값

알아두기

val 값을 생략하면, index 인자의 부호에 따라 양수는 ON, 음수는 OFF 가 됩니다.

- **리턴**

값	설명
0	성공
음수값	오류

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
set_tool_digital_outputs(1, ON)   #로봇 암의 1번 점점 ON
set_tool_digital_output(6, OFF)   #로봇 암의 6번 점점 OFF
set_tool_digital_output(3)        #3번 접점 ON, val 인자가 생략된 경우 양수 ON
```

```
set_tool_digital_output(-3)          #3번 접점 OFF, val 인자가 생략된 경우 음수 OFF
```

12.2.4 get_tool_digital_input(index)

- **기능**

로봇 툴의 신호를 디지털 접점에서 불러오기 위한 명령문입니다.

- **인수**

인수명	자료형	기본값	설명
index	int	-	로봇 Tool I/O 접점 번호 (1~6)

- **리턴**

값	설명
1	ON
0	OFF
음수값	실패

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
get_tool_digital_input(1)        #Tool IO 입력 1번 접점 읽기
get_tool_digital_input(6)        #Tool IO 입력 6번 접점 읽기
```

12.2.5 set_mode_analog_output(ch, mod)

- **기능**

컨트롤러 아날로그 출력에 대한 채널 모드를 설정합니다.

- **인수**

인수명	자료형	기본값	설명
ch	int	-	• 1 : channel 1 • 2 : channel 2
mod	int	-	analog io mode • DR_ANALOG_CURRENT: 전류 모드 • DR_ANALOG_VOLTAGE: 전압 모드

- **리턴**

값	설명
0	성공
음수값	실패

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
# analog_output channel 1을 전류 모드로 설정함
set_mode_analog_output(ch=1, mod=DR_ANALOG_CURRENT)

# analog_output channel 2를 전압 모드로 설정함
set_mode_analog_output(ch=2, mod=DR_ANALOG_VOLTAGE)
```

12.2.6 set_mode_analog_input(ch, mod)

- **기능**

컨트롤러 아날로그 입력 대한 채널 모드를 설정합니다.

- **인수**

인수명	자료형	기본값	설명
ch	int	-	• 1 : channel 1 • 2 : channel 2
mod	int	-	analog io mode • DR_ANALOG_CURRENT: 전류 모드 • DR_ANALOG_VOLTAGE: 전압 모드

- **리턴**

값	설명
0	성공
음수값	실패

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
# analog input channel 1을 전류 모드로 설정함
set_mode_analog_input(ch=1, mod=DR_ANALOG_CURRENT)

# analog input channel 2를 전압 모드로 설정함.
set_mode_analog_input(ch=2, mod=DR_ANALOG_VOLTAGE)
```

12.2.7 set_analog_output(ch, val)

- **기능**

컨트롤러 아날로그 출력에 해당하는 채널의 값을 출력합니다.

- **인수**

인수명	자료형	기본값	설명
ch	int	-	• 1 : channel 1 • 2 : channel 2
val	float	-	analog 출력 값 • 전류 모드인 경우: 4.0~20.0 [mA] • 전압 모드인 경우: 0~10.0 [V]

- **리턴**

값	설명
0	성공
음수값	실패

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
set_mode_analog_output(ch=1, mod=DR_ANALOG_CURRENT) #out ch1=current mode
set_mode_analog_output(ch=2, mod=DR_ANALOG_VOLTAGE) #out ch1=voltage mode
set_analog_output(ch=1, val=5.2)     #channel 1에 5.2 mA 출력
set_analog_output(ch=2, val=10.0)    #channel 2에 10V 출력
```

12.2.8 get_analog_input(ch)

- **기능**

컨트롤러 아날로그 입력에 해당하는 채널의 값을 불러옵니다.

- **인수**

인수명	자료형	기본값	설명
ch	int	-	• 1 : channel 1 • 2 : channel 2

- **리턴**

값	설명
float	해당 channel 의 analog input 값 • 전류 모드인 경우: 4.0~20.0 [mA] • 전압 모드인 경우: 0~10.0 [V]

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
set_mode_analog_input(ch=1, mod=DR_ANALOG_CURRENT) #input ch1=current mode
set_mode_analog_input(ch=2, mod=DR_ANALOG_VOLTAGE)   #input ch2=voltage mode

Cur = get_analog_input(1)   # channel 1의 analog input 전류 값 읽기
Vol = get_analog_input(2)   # channel 2의 analog input 전압 값 읽기.
```

13. 외부 통신 함수

13.1 Modbus

13.1.1 add_modbus_signal (ip, port, name, reg_type, index, value=0)

- **기능**

ModbusTCP의 신호를 등록합니다. Modbus I/O 설정의 경우 티치팬던트 I/O set-up 메뉴에서 설정해야하지만 티치 팬던트 사용이 어려운 경우에 테스트를 위해서만 본 명령어를 사용하시기 바랍니다. 이 명령어를 사용하여 셋팅한 경우 티치 팬던트에서 Modbus 관련 메뉴가 동작하지 않습니다.

- **인수**

인수명	자료형	기본값	설명
ip	string	-	modbusTCP 모듈 ip 주소
port	int	-	modbusTCP 모듈 port
name	string	-	modbus signal 이름
reg_type	int	-	Modbus의 신호 타입 • DR_MODBUS_DIG_INPUT • DR_MODBUS_DIG_OUTPUT • DR_MODBUS_REG_INPUT • DR_MODBUS_REG_OUTPUT
index	int	-	Modbus signal의 index
value	int	0	type이 DR_MODBUS_DIG_OUTPUT 또는 DR_MODBUS_REG_OUTPUT일 때 출력값 (그 외 경우에는 무시됩니다.)

- **리턴**

값	설명
0	성공
음수값	실패

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
#Modbus IO 2개를 연결하고 접점을 할당하는 예제
#Modbus IO 1번 : IP 192.168.127.254, input 8점: "di1"~"di8", output 8점:
"do1"~"do8"
#Modbus IO 2번 : IP 192.168.127.253, input 8점: "di9"~"di16", output 8점:
"do9"~"do16"

# set <modbus 1> input : di1~di8
add_modbus_signal(ip="192.168.127.254",port=502, name="di1",
reg_type=DR_MODBUS_REG_INPUT, index=0)
add_modbus_signal(ip="192.168.127.254",port=502, name="di2",
reg_type=DR_MODBUS_REG_INPUT, index=1)
add_modbus_signal(ip="192.168.127.254",port=502, name="di3",
reg_type=DR_MODBUS_REG_INPUT, index=2)
add_modbus_signal(ip="192.168.127.254",port=502, name="di4",
reg_type=DR_MODBUS_REG_INPUT, index=3)
add_modbus_signal(ip="192.168.127.254",port=502, name="di5",
reg_type=DR_MODBUS_REG_INPUT, index=4)
add_modbus_signal(ip="192.168.127.254",port=502, name="di6",
reg_type=DR_MODBUS_REG_INPUT, index=5)
add_modbus_signal(ip="192.168.127.254",port=502, name="di7",
reg_type=DR_MODBUS_REG_INPUT, index=6)
add_modbus_signal(ip="192.168.127.254",port=502, name="di8",
reg_type=DR_MODBUS_REG_INPUT, index=7)

# set <modbus 1> output : do1~do8
```

```
add_modbus_signal(ip="192.168.127.254",port=502, name="do1",
reg_type=DR_MODBUS_REG_OUTPUT, index=0, value=0)
add_modbus_signal(ip="192.168.127.254",port=502, name="do2",
reg_type=DR_MODBUS_REG_OUTPUT, index=1, value=0)
add_modbus_signal(ip="192.168.127.254",port=502, name="do3",
reg_type=DR_MODBUS_REG_OUTPUT, index=2, value=0)
add_modbus_signal(ip="192.168.127.254",port=502, name="do4",
reg_type=DR_MODBUS_REG_OUTPUT, index=3, value=0)
add_modbus_signal(ip="192.168.127.254",port=502, name="do5",
reg_type=DR_MODBUS_REG_OUTPUT, index=4, value=0)
add_modbus_signal(ip="192.168.127.254",port=502, name="do6",
reg_type=DR_MODBUS_REG_OUTPUT, index=5, value=0)
add_modbus_signal(ip="192.168.127.254",port=502, name="do7",
reg_type=DR_MODBUS_REG_OUTPUT, index=6, value=0)
add_modbus_signal(ip="192.168.127.254",port=502, name="do8",
reg_type=DR_MODBUS_REG_OUTPUT, index=7, value=0)

#=======================================================
========
# set <modbus 2> input : di9~di16
add_modbus_signal(ip="192.168.127.253",port=502, name="di9",
reg_type=DR_MODBUS_REG_INPUT, index=0)
add_modbus_signal(ip="192.168.127.253",port=502, name="di10",
reg_type=DR_MODBUS_REG_INPUT, index=1)
add_modbus_signal(ip="192.168.127.253",port=502, name="di11",
reg_type=DR_MODBUS_REG_INPUT, index=2)
add_modbus_signal(ip="192.168.127.253",port=502, name="di12",
reg_type=DR_MODBUS_REG_INPUT, index=3)
add_modbus_signal(ip="192.168.127.253",port=502, name="di13",
reg_type=DR_MODBUS_REG_INPUT, index=4)
add_modbus_signal(ip="192.168.127.253",port=502, name="di14",
reg_type=DR_MODBUS_REG_INPUT, index=5)
add_modbus_signal(ip="192.168.127.253",port=502, name="di15",
reg_type=DR_MODBUS_REG_INPUT, index=6)
add_modbus_signal(ip="192.168.127.253",port=502, name="di16",
```

```
reg_type=DR_MODBUS_REG_INPUT, index=7)

# set <modbus 2> output : do9~do16
add_modbus_signal(ip="192.168.127.253",port=502, name="do9",
reg_type=DR_MODBUS_REG_OUTPUT, index=0, value=0)
add_modbus_signal(ip="192.168.127.253",port=502, name="do10",
reg_type=DR_MODBUS_REG_OUTPUT, index=1, value=0)
add_modbus_signal(ip="192.168.127.253",port=502, name="do11",
reg_type=DR_MODBUS_REG_OUTPUT, index=2, value=0)
add_modbus_signal(ip="192.168.127.253",port=502, name="do12",
reg_type=DR_MODBUS_REG_OUTPUT, index=3, value=0)
add_modbus_signal(ip="192.168.127.253",port=502, name="do13",
reg_type=DR_MODBUS_REG_OUTPUT, index=4, value=0)
add_modbus_signal(ip="192.168.127.253",port=502, name="do14",
reg_type=DR_MODBUS_REG_OUTPUT, index=5, value=0)
add_modbus_signal(ip="192.168.127.253",port=502, name="do15",
reg_type=DR_MODBUS_REG_OUTPUT, index=6, value=0)
add_modbus_signal(ip="192.168.127.253",port=502, name="do16",
reg_type=DR_MODBUS_REG_OUTPUT, index=7, value=0)
```

13.1.2 del_modbus_signal (name)

- **기능**

등록된 Modbus의 신호를 삭제합니다. Modbus I/O 설정의 경우 티치 팬던트 I/O set-up 메뉴에서 설정해야 하지만 티치 팬던트 사용이 어려운 경우에 테스트를 위해서만 본 명령어를 사용하시기 바랍니다 이 명령어를 사용하여 셋팅한 경우 티치 팬던트에서 Modbus 관련 메뉴가 동작하지 않습니다.

- **인수**

인수명	자료형	기본값	설명
name	string	-	등록된 modbus 신호의 이름

- **리턴**

값	설명
0	성공
음수값	실패

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
# Modbus IO 신호가 "di1", "do1" 로 등록되어 있는데,
# 이 신호 등록을 삭제하고자 할 때 하기 명령을 사용합니다. .
del_modbus_signal("di1")        # "di1" 접점 등록 삭제
del_modbus_signal("do1")        # "do1" 접점 등록 삭제
```

13.1.3 set_modbus_output(iobus, val)

- **기능**

외부 Modbus 장치에 신호를 내보내기 위한 명령어입니다.

- **인수**

인수명	자료형	기본값	설명
iobus	string	-	modbus 이름(TP에서 설정)
value	int	-	Modbus digital I/O 인 경우 • ON : 1 • OFF : 0
			Modbus analog I/O 인 경우 value

- **리턴**

값	설명
0	성공
음수값	실패

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
#Modbus digital I/O가 연결되어 있고, 신호가 "do1", "do2" 로 등록되어 있는 경우
set_modbus_output("do1", ON)
set_modbus_output("do2", OFF)

#Modbus analog I/O가 연결되어 있고, 신호가 "reg1", "reg2" 로 등록되어 있는 경
```

```
우
set_modbus_output("reg1", 10)
set_modbus_output("reg2", 24)
```

13.1.4 get_modbus_input(iobus)

- **기능**

Modbus 장치에서 신호를 읽어오기 위한 명령어입니다.

- **인수**

인수명	자료형	기본값	설명
iobus	string	-	modbus 이름(TP에서 설정)

- **리턴**

값	설명
0 or 1	Modbus Digital I/O 인 경우 ON or OFF
value	Modbus Analog 모듈인 경우 해당 레지스터 값

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- **예제**

```
#Modbus digital I/O가 연결되어 있고, 신호가 "di1", "di2" 로 등록되어 있는 경우
get_modbus_input("di1")
get_modbus_input("di2")
#Modbus analog I/O가 연결되어 있고, 신호가 "reg1", "reg2" 로 등록되어 있는 경우
get_modbus_input("reg1")
get_modbus_input("reg2")
```


R