1.1 Real-Time External Control (Temporary Version_1)

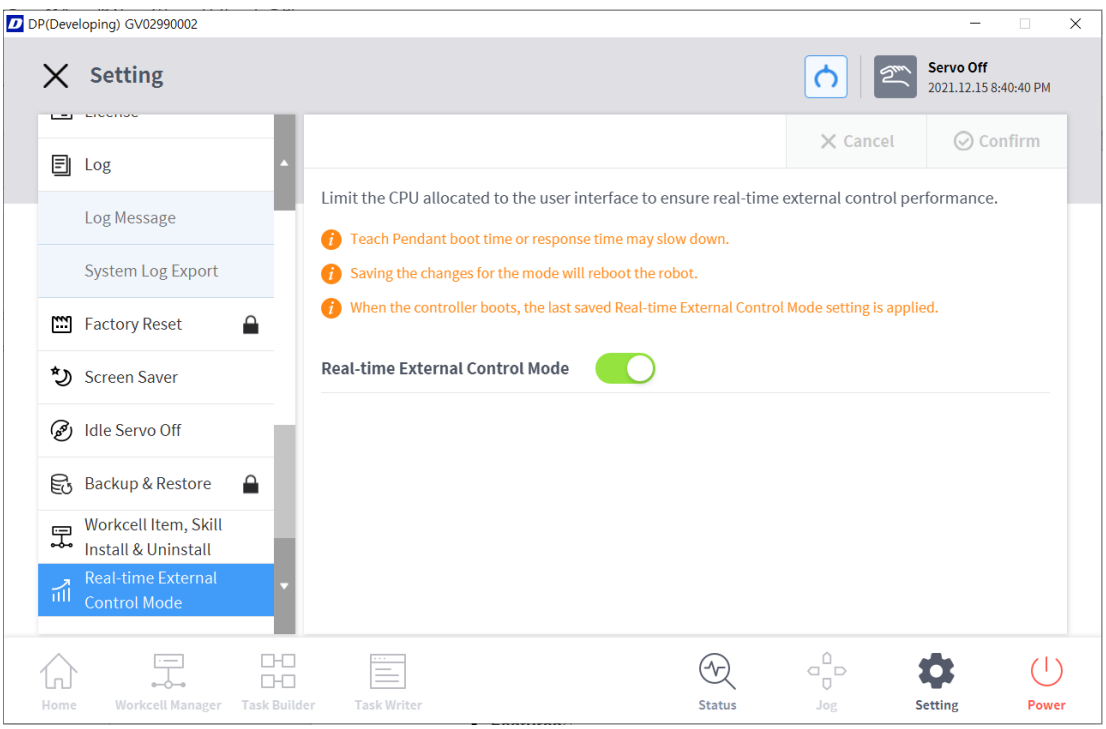
Note

- This is development version.
- Stable version will be released v2.9
- Currently, servoj/l has some minor bug. Please, test monitoring_rt_data callback, read_rt_data, speedj/l, torque first.
- Windows example is not real-time, but you can test simple function test.
- Refer to xenomai example, configure your own real-time system.

1.1.1 Real-Time External Control Mode

- **Features**

You have to turn on real-time external control mode in TP. If you don't turn on this mode, controller doesn't allow real-time external control api.
Go to Setting > Real-time External Control Mode, and enable this mode. Reboot. After rebooting, check if this toggle still enabled.



1.1.2 CDRFLEx.connect_rt_control

- **Features**

This function connects to real-time external control.

- **Parameter**

Parameter Name	Data Type	Default Value	Description
strIPAddr	string	192.168.137.100	IP Address
usPort	unsigned int	12345	Port Number

Note

- Real-time external control uses udp/ip communication.
- This channel is independent from the original tcp/ip api. It doesn't care control authority.

1.1.3 CDRFLEx.disconnect_rt_control

- **Features**

This function disconnects real-time external control.

1.1.4 CDRFLEx.set_rt_control_output

- **Features**

This function sets output data (External Controller → Robot Controller) properties.

- **Parameter**

Parameter Name	Data Type	Default Value	Description
strVersion	string	-	Protocol version (for now, just use v1.0)
fPeriod	float	-	Period (sec)
nLossCnt	int	-	Loss count

Note

- You can set fPeriod 0.001~1 sec.
- nLossCnt is unused for now.

- **Output Data List**

Name	Type	Description
time_stamp	double	timestamp at the data of data acquisition [s]
actual_joint_position	float[6]	actual joint position from incremental encoder at motor side(used for control) [deg]
actual_joint_position_abs	float[6]	actual joint position from absolute encoder at link side (used for exact link position) [deg]
actual_joint_velocity	float[6]	actual joint velocity from incremental encoder at motor side [deg/s]
actual_joint_velocity_abs	float[6]	actual joint velocity from absolute encoder at link side [deg/s]
actual_tcp_position	float[6]	actual robot tcp position w.r.t. base coordinates: (x, y, z, a, b, c), where (a, b, c) follows Euler ZYZ notation [mm, deg]
actual_tcp_velocity	float[6]	actual robot tcp velocity w.r.t. base coordinates [mm, deg/s]
actual_flange_position	float[6]	actual robot flange position w.r.t. base coordinates: (x, y, z, a, b, c), where (a, b, c) follows Euler ZYZ notation [mm, deg]
actual_flange_velocity	float[6]	robot flange velocity w.r.t. base coordinates [mm, deg/s]

actual_motor_torque	float[6]	actual motor torque applying gear ratio = gear_ratio * current2torque_constant * motor current [Nm]
actual_joint_torque	float[6]	estimated joint torque by robot controller [Nm]
raw_joint_torque	float[6]	calibrated joint torque sensor data
raw_force_torque	float[6]	calibrated force torque sensor data w.r.t. flange coordinates [N, Nm]
external_joint_torque	float[6]	estimated joint torque [Nm]
external_tcp_force	float[6]	estimated tcp force w.r.t. base coordinates [N, Nm]
target_joint_position	float[6]	target joint position [deg]
target_joint_velocity	float[6]	target joint velocity [deg/s]
target_joint_acceleration	float[6]	target joint acceleration [deg/s^2]
target_motor_torque	float[6]	target motor torque [Nm]
target_tcp_position	float[6]	target tcp position w.r.t. base coordinates: (x, y, z, a, b, c), where (a, b, c) follows Euler ZYZ notation [mm, deg]
target_tcp_velocity	float[6]	target tcp velocity w.r.t. base coordinates [mm, deg/s]

jacobian_matrix	float[6][6]	jacobian matrix=J(q) w.r.t. base coordinates
gravity_torque	float[6]	gravity torque=g(q) [Nm]
coriolis_matrix	float[6][6]	coriolis matrix=C(q) [Nm.s]
mass_matrix	float[6][6]	mass matrix=M(q)+B [Nm.s^2]
solution_space	uint8	robot configuration
singularity	float	minimum singular value
operation_speed_rate	float	current operation speed rate(1~100 %)
joint_temperature	float[6]	joint temperature(celsius)
controller_digital_input	uint16	controller digital input(16 channel)
controller_digital_output	uint16	controller digital output(16 channel)
controller_analog_input_type	uint8	controller analog input type(2 channel)
controller_analog_input	float[2]	controller analog input(2 channel)
controller_analog_output_type	uint8	controller analog output type(2 channel)
controller_analog_output	float[2]	controller analog output(2 channel)

flange_digital_input	uint8	flange digital input(A-Series: 2 channel, M/H-Series: 6 channel)
flange_digital_output	uint8	flange digital input(A-Series: 2 channel, M/H-Series: 6 channel)
flange_analog_input	float[4]	flange analog input(A-Series: 2 channel, M/H-Series: 4 channel)
external_encoder_strobe_count	uint8[2]	strobe count(increased by 1 when detecting setting edge)
external_encoder_count	uint32[2]	external encoder count
goal_joint_position	float[6]	final goal joint position (reserved)
goal_tcp_position	float[6]	final goal tcp position (reserved)
robot_mode	uint8	ROBOT_MODE_MANUAL(0), ROBOT_MODE_AUTONOMOUS(1), ROBOT_MODE_MEASURE(2)
robot_state	uint8	STATE_INITIALIZING(0), STATE_STANDBY(1), STATE_MOVING(2), STATE_SAFE_OFF(3), STATE_TEACHING(4), STATE_SAFE_STOP(5), STATE_EMERGENCY_STOP(6), STATE_HOMMING(7), STATE_RECOVERY(8), STATE_SAFE_STOP2(9), STATE_SAFE_OFF2(10)
control_mode	Uint16	position control mode, torque mode

- **Example**

```
// Connect and configure output data
CDRFLEx drfl;
drfl.connect_rt_control();
string version = “v1.0”;
float period = 0.001; // 1 msec
int losscount = 4;
drfl.set_rt_control_output(version, period, losscount);
```

1.1.5 CDRFLEx.start_rt_control

- **Features**

This function starts to send and receive real-time external control data.

- **Example**

```
// Connect and start
CDRFLEx drfl;
drfl.connect_rt_control();
string version = “v1.0”;
float period = 0.001; // 1 msec
int losscount = 4;
drfl.set_rt_control_output(version, period, losscount);
drfl.start_rt_control();
```

1.1.6 CDRFLEx.stop_rt_control

- **Features**

This function stops to send and receive real-time external control data.

- **Example**

```
// Connect and start/stop
CDRFLEx drfl;
drfl.connect_rt_control();
string version = “v1.0”;
float period = 0.001; // 1 msec
int losscount = 4;
drfl.set_rt_control_output(version, period, losscount);
drfl.start_rt_control();
// do something
drfl.stop_rt_control();
```

1.1.7 CDRFLEx. set_on_rt_monitoring_data

- **Features**

  This function is callback function of receiving controller's real-time output data.

Note

- You don't have to call non real-time function such as printf over 30Hz.

- **Example**

```
// callback function
void OnRTMonitoringData(LPRT_OUTPUT_DATA_LIST tData)
{
	return;
	static int td = 0;
	if (td++==1000) {
		td = 0;
		printf("timestamp : %.3f\n", tData->time_stamp);
		printf("actual_joint_position : %f %f %f %f %f %f\n", tData-
>actual_joint_position[0], tData->actual_joint_position[1], tData->actual_joint_position[2], tData-
>actual_joint_position[3], tData->actual_joint_position[4], tData->actual_joint_position[5]);
		printf("actual_motor_torque : %f %f %f %f %f %f\n", tData-
>actual_motor_torque[0], tData->actual_motor_torque[1], tData->actual_motor_torque[2], tData-
>actual_motor_torque[3], tData->actual_motor_torque[4], tData->actual_motor_torque[5]);
		printf("actual_grav_torque : %f %f %f %f %f %f\n", tData->gravity_torque[0],
tData->gravity_torque[1], tData->gravity_torque[2], tData->gravity_torque[3], tData-
>gravity_torque[4], tData->gravity_torque[5]);
		printf("target torque : %f %f %f %f %f %f\n", tData->target_motor_torque[0],
tData->target_motor_torque[1], tData->target_motor_torque[2], tData->target_motor_torque[3],
tData->target_motor_torque[4], tData->target_motor_torque[5]);
	}
}

// main.cpp
CDRFLEx drfl;
drfl.connect_rt_control();
string version = "v1.0";
float period = 0.001; // 1 msec
int losscount = 4;
Drfl.set_on_rt_monitoring_data(OnRTMonitoringData);
drfl.set_rt_control_output(version, period, losscount);
drfl.start_rt_control();
```

1.1.8 CDRFLEx.read_rt_data

- **Features**

  This function reads the real-time output data of controller.

- **Example**

```
// main.cpp
CDRFLEx drfl;
drfl.connect_rt_control();
string version = “v1.0”;
float period = 0.001; // 1 msec
int losscount = 4;
Drfl.set_on_rt_monitoring_data(OnRTMonitoringData);
drfl.set_rt_control_output(version, period, losscount);
drfl.start_rt_control();

float q[NUMBER_OF_JOINT] = {0.0, };
float q_dot[NUMBER_OF_JOINT] = {0.0, };
float trq_g[NUMBER_OF_JOINT] = {0.0, };
while (1)
{
	time=(++count)*st;

	memcpy(q, drfl.read_data_rt()->actual_joint_position, sizeof(float)*6);
	memcpy(q_dot, drfl.read_data_rt()->actual_joint_velocity, sizeof(float)*6;
	memcpy(trq_g, drfl.read_data_rt()->gravity_torque, sizeof(float)*6);
	memcpy(trq_d, trq_g, sizeof(float)*6);

	drfl.torque_rt(trq_d, 0);

	if(time > plan1.time)
	{
		time=0;
		 Drfl.stop(STOP_TYPE_SLOW);
		 break;
	}

	rt_task_wait_period(NULL);
}
```

1.1.9 CDRFLEx.set_velj_rt

- **Features**

  This function sets the global joint velocity limit.

- **Parameter**

Parameter Name	Data Type	Default Value	Description
vel	float[6]	-	Joint velocity limit

Note

・ If you plan over this velocity limit, controller just pops info message.

1.1.10 CDRFLEx.set_accj_rt

- **Features**

  This function sets the global joint acceleration limit.

- **Parameter**

Parameter Name	Data Type	Default Value	Description
vel	float[6]	-	Joint acceleration limit

Note

・ If you plan over this acceleration limit, controller just pops info message.

1.1.11 CDRFLEx.set_velx_rt

- **Features**

  This function sets the global task velocity limit.

- **Parameter**

Parameter Name	Data Type	Default Value	Description
fTransVel	float	-	Task translation velocity limit
fRotationVel	float		Task rotation velocity limit

Note

- If you plan over this velocity limit, controller pops info.

1.1.12 CDRFLEx.set_accx_rt

- **Features**

  This function reads the global task acceleration limit.

- **Parameter**

Parameter Name	Data Type	Default Value	Description
fTransVel	float	-	Task translation acceleration limit
fRotationVel	float		Task rotation acceleration limit

Note

- If you plan over this acceleration limit, controller pops info.

1.1.13 CDRFLEx.servoj_rt

- **Features**

  This is the low-level joint position control function.

- **Parameter**

Parameter Name	Data Type	Default Value	Description
fTargetPos	float[6]	-	target joint position
fTargetVel	float[6]	-	target joint velocity. If it is null, it will be auto-generated.
fTargetAcc	float[6]	-	target joint acceleration, if it is null, it will be auto-generated
fTargetTime	float	-	target time

Note

- Asnyc command

- interpolates (fTargetPos, fTargetVel, fTargetAcc) @ fTargetTime
- if fTargetTime <= controller's control period(=1ms), it directly pass command
- if intermediate fTargetVel, fTargetAcc is over globally-setting maximum velocity/acceleration, alarm Info message.

1.1.14 CDRFLEx.servol_rt

- **Features**

This is the low-level task position control function.

- **Parameter**

Parameter Name	Data Type	Default Value	Description
fTargetPos	float[6]	-	target task position [mm, deg] (rotation: euler zyz)
fTargetVel	float[6]	-	target task velocity. If it is null, it will be auto-generated.
fTargetAcc	float[6]	-	target task acceleration, if it is null, it will be auto-generated
fTargetTime	float	-	target time

Note

- Asnyc command
- interpolates (fTargetPos, fTargetVel, fTargetAcc) @ fTargetTime
- if fTargetTime <= controller's control period(=1ms), it directly pass command
- if intermediate fTargetVel, fTargetAcc is over globally-setting maximum velocity/acceleration, alarm Info message.

1.1.15 CDRFLEx.speedj_rt

- **Features**

This is the low-level joint velocity control function.

- **Parameter**

Parameter Name	Data Type	Default Value	Description
fTargetVel	float[6]	-	target joint velocity. If it is null, it will be auto-generated.
fTargetAcc	float[6]	-	target joint acceleration, if it is null, it will be auto-generated
fTargetTime	float	-	target time

Note

- Asnyc command
- interpolates (fTargetVel, fTargetAcc) @ fTargetTime
- if fTargetTime <= controller's control period(=1ms), it directly pass command
- if intermediate fTargetAcc is over globally-setting maximum velocity/acceleration, alarm Info message.

1.1.16 CDRFLEx.speedl_rt

- **Features**

  This is the low-level task velocity control function.

- **Parameter**

Parameter Name	Data Type	Default Value	Description
fTargetPos	float[6]	-	target joint position
fTargetVel	float[6]	-	target joint velocity. If it is null, it will be auto-generated.
fTargetAcc	float[6]	-	target joint acceleration, if it is null, it will be auto-generated
fTargetTime	float	-	target time

Note

- Asnyc command
- interpolates (fTargetPos, fTargetVel, fTargetAcc) @ fTargetTime
- if fTargetTime <= controller's control period(=1ms), it directly pass command.

- if intermediate v, a is over globally-setting maximum velocity/acceleration, alarm Info message.

1.1.17 CDRFLEx.torque_rt

- **Features**

  This is the low-level torque control function.

- **Parameter**

Parameter Name	Data Type	Default Value	Description
fMotorTor	float[6]	-	target motor torque.
fTargetTime	float	-	target time

Note

- Asnyc command
- interpolates (fMotorTor) @ fTargetTime
- if fTargetTime <= controller's control period(=1ms), it directly pass command
- In the case of H Series, gravity compensation mechanism helps this motor torque at the 2nd joint. You can get gravity_torque in real-time output data (=link gravity torque – gravity compensation mechanism compensation torque)

API

Doosan Robotics

www.doosanrobotics.com