

Manual Version 1.6

Software Version 2.5

Doosan Robot

M0609 | M1509 | M1013 | M0617

Programming Guide



DOOSAN

머리말	29
저작권	29
1. DRL Basic Syntax	30
1.1 들여쓰기.....	30
1.2 주석.....	31
1.3 변수명.....	32
1.4 수치.....	33
1.5 문자.....	34
1.5.1 String.....	34
1.5.2 +, *	35
1.5.3 인덱싱 & 슬라이싱	35
1.6 list	36
1.7 tuple	37
1.8 dictionary	37
1.9 함수.....	38
1.10 스코핑 룸.....	39
1.11 인수 모드.....	40
1.12 pass.....	41
1.13 if	42
1.14 while.....	43
1.15 for	44
1.16 break	44
1.17 continue.....	45
1.18 반복문의 else	46

2. 모션 관련 명령어 47

2.1 posj(q1=0, q2=0, q3=0, q4=0, q5=0, q6=0).....	47
2.2 posx(x=0, y=0, z=0, w=0, p=0, r=0).....	49
2.3 trans(pos, delta, ref, ref_out)	50
2.4 posb(seg_type, posx1, posx2=None, radius=0).....	52
2.5 fkin(pos, ref).....	54
2.6 ikin(pos, sol_space, ref)	56
2.7 addto(pos, add_val=None)	58
2.8 set_velj(vel).....	59
2.9 set_accj(acc).....	61
2.10 set_velx(vel1, vel2).....	63
2.11 set_velx(vel).....	65
2.12 set_accx(acc1, acc2).....	67
2.13 set_accx(acc)	69
2.14 set_tcp(name).....	71
2.15 set_ref_coord(coord)	72
2.16 movej	74
2.17 movel	77
2.18 movejx	81
2.19 movec	84
2.20 movesj	88
2.21 movesx	91
2.22 moveb	95
2.23 move_spiral	100
2.24 move_periodic	103
2.25 move_home.....	107
2.26 amovej	109
2.27 amovel	112
2.28 amovejx	115

2.29 amovec	118
2.30 amovesj	121
2.31 amovesx	124
2.32 amoveb	127
2.33 amove_spiral	131
2.34 amove_periodic	134
2.35 mwait(time=0)	137
2.36 begin_blend(radius=0)	139
2.37 end_blend()	141
2.38 check_motion()	143
2.39 stop(st_mode)	145
2.40 change_operation_speed(speed)	147
2.41 wait_manual_guide()	149
2.42 wait_nudge()	151
2.43 enable.Alter_motion(n, mode, ref, limit_dPOS, limit_dPOS_per)	153
2.44 alter_motion([x,y,z,rx,ry,rz])	156
2.45 disable.Alter_motion()	158

3. 제어 보조 명령어 160

3.1 get_control_mode()	160
3.2 get_control_space()	161
3.3 get_current_posj()	162
3.4 get_current_velj()	163
3.5 get_desired_posj()	164
3.6 get_desired_velj()	165
3.7 get_current_posx(ref)	166
3.8 get_current_tool_flange_posx(ref)	168
3.9 get_current_velx(ref)	169
3.10 get_desired_posx(ref)	170

3.11	get_desired_velx(ref)	171
3.12	get_current_solution_space().....	172
3.13	get_current_rotm(ref).....	173
3.14	get_joint_torque()	174
3.15	get_external_torque().....	175
3.16	get_tool_force(ref).....	176
3.17	get_solution_space(pos)	177
3.18	get_orientation_error(xd, xc, axis)	178

4. 기타 설정 및 안전 관련 명령어 180

4.1	get_workpiece_weight()	180
4.2	reset_workpiece_weight().....	181
4.3	set_tool(name)	182
4.4	set_tool_shape(name)	183
4.5	set_singularity_handling(mode).....	184

5. 힘/강성 제어 및 기타 사용자 편의 기능 186

5.1	parallel_axis(x1, x2, x3, axis, ref)	186
5.2	parallel_axis(vect, axis, ref).....	188
5.3	align_axis(x1, x2, x3, pos, axis, ref).....	190
5.4	align_axis(vect, pos, axis, ref)	192
5.5	is_done_bolt_tightening(m=0, timeout=0, axis=None)	194
5.6	release_compliance_ctrl().....	196
5.7	task_compliance_ctrl(stx, time).....	197
5.8	set_stiffnessx(stx, time)	199
5.9	calc_coord(x1, x2, x3, x4, ref, mod).....	201
5.10	set_user_cart_coord(pos, ref).....	203
5.11	set_user_cart_coord(x1, x2, x3, pos, ref).....	205
5.12	set_user_cart_coord(u1, v1, pos, ref).....	207

5.13	overwrite_user_cart_coord(id, pos, ref).....	209
5.14	get_user_cart_coord(id).....	211
5.15	set_desired_force(fd, dir, time, mod).....	212
5.16	release_force(time=0)	214
5.17	check_position_condition(axis, min, max, ref, mod, pos).....	216
5.18	check_force_condition(axis, min, max, ref).....	219
5.19	check_orientation_condition(axis, min, max, ref, mod).....	221
5.20	check_orientation_condition(axis, min, max, ref, mod, pos).....	224
5.21	coord_transform(pose_in, ref_in, ref_out).....	227

6. 시스템 명령어 229

6.1	IO 관련.....	229
6.1.1	set_digital_output(index, val=None)	229
6.1.2	set_digital_outputs(bit_list).....	231
6.1.3	set_digital_outputs(bit_start, bit_end, val).....	232
6.1.4	get_digital_input(index)	234
6.1.5	get_digital_inputs(bit_list)	235
6.1.6	get_digital_inputs(bit_start, bit_end).....	236
6.1.7	wait_digital_input(index, val, timeout=None).....	237
6.1.8	set_tool_digital_output(index, val=None)	239
6.1.9	set_tool_digital_outputs(bit_list).....	240
6.1.10	set_tool_digital_outputs(bit_start, bit_end, val).....	241
6.1.11	get_tool_digital_input(index)	243
6.1.12	get_tool_digital_inputs(bit_list)	244
6.1.13	get_tool_digital_inputs(bit_start, bit_end).....	245
6.1.14	wait_tool_digital_input(index, val, timeout=None).....	246
6.1.15	set_mode_analog_output(ch, mod)	248
6.1.16	set_mode_analog_input(ch, mod)	249
6.1.17	set_analog_output(ch, val)	250
6.1.18	get_analog_input(ch)	251

6.2 TP 연동	252
6.2.1 tp_popup(message, pm_type=DR_PM_MESSAGE, button_type=0)	252
6.2.2 tp_log(message)	254
6.2.3 tp_progress(cur_progress, total_progress)	255
6.2.4 tp_get_user_input(message, input_type)	256
6.3 Thread	258
6.3.1 thread_run(th_func_name, loop=False)	258
6.3.2 thread_stop(th_id)	260
6.3.3 thread_pause(th_id)	261
6.3.4 thread_resume(th_id)	262
6.3.5 thread_state(th_id)	264
6.3.6 통합 예제	265
6.4 기타	267
6.4.1 wait(time)	267
6.4.2 exit()	268
6.4.3 sub_program_run(name)	269
6.4.4 drl_report_line(option)	271
6.4.5 set_fm(key, value)	272

7. 수학 함수	273
7.1 sin(x)	273
7.2 cos(x)	274
7.3 tan(x)	275
7.4 asin(x)	276
7.5 acos(x)	277
7.6 atan(x)	278
7.7 atan2(y, x)	279
7.8 ceil(x)	280
7.9 floor(x)	281

7.10 pow(x, y)	282
7.11 sqrt(x).....	283
7.12 log(x, b).....	284
7.13 d2r(x).....	285
7.14 r2d(x).....	286
7.15 norm(x)	287
7.16 random().....	288
7.17 rotx(angle).....	289
7.18 roty(angle).....	290
7.19 rotz(angle).....	291
7.20 rotm2eul(rotm)	292
7.21 rotm2rotvec(rotm)	293
7.22 eul2rotm([alpha,beta,gamma]).....	294
7.23 eul2rotvec([alpha,beta,gamma])	295
7.24 rotvec2eul([rx,ry,rz])	296
7.25 rotvec2rotm([rx,ry,rz])	297
7.26 htrans(posx1,posx2)	298
7.27 get_intermediate_pose(posx1,posx2,alpha).....	299
7.28 get_distance(posx1, posx2)	300
7.29 get_normal(x1, x2, x3).....	301
7.30 add_pose(posx1,posx2).....	302
7.31 subtract_pose(posx1,posx2).....	303
7.32 inverse_pose(posx1)	304
7.33 dot_pose(posx1, posx2).....	305
7.34 cross_pose(posx1, posx2)	306
7.35 unit_pose(posx1).....	307

8. 외부 통신 명령어 308

8.1 Serial	308
------------------	-----

8.1.1	serial_open(port=None, baudrate=115200, bytesize=DR_EIGHTBITS, parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE).....	308
8.1.2	serial_close(ser).....	310
8.1.3	serial_state(ser).....	311
8.1.4	serial_set_inter_byte_timeout(ser, timeout=None).....	312
8.1.5	serial_write(ser, tx_data).....	313
8.1.6	serial_read(ser, length=1, timeout=-1).....	314
8.1.7	serial_get_count().....	316
8.1.8	serial_get_info(id).....	317
8.1.9	통합 예제.....	318
8.2	Tcp/Client.....	321
8.2.1	client_socket_open(ip, port).....	321
8.2.2	client_socket_close(sock).....	322
8.2.3	client_socket_state(sock).....	323
8.2.4	client_socket_write(sock, tx_data).....	324
8.2.5	client_socket_read(sock, length=1, timeout=-1).....	325
8.2.6	통합 예제.....	327
8.3	Tcp/Server.....	330
8.3.1	server_socket_open(port).....	330
8.3.2	server_socket_close(sock).....	331
8.3.3	server_socket_state(sock).....	332
8.3.4	server_socket_write(sock, tx_data).....	333
8.3.5	server_socket_read(sock, length=1, timeout=-1).....	334
8.3.6	통합 예제.....	336
8.4	Modbus.....	339
8.4.1	add_modbus_signal(ip, port, name, reg_type, index, value=0, slaveid=255).....	339
8.4.2	add_modbus_rtu_signal(slaveid=1, port=None, baudrate=115200, bytesize=DR_EIGHTBITS, parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE, name, reg_type, index, value=0).....	343
8.4.3	del_modbus_signal(name)	345
8.4.4	set_modbus_output(iobus, val).....	346
8.4.5	set_modbus_outputs(iobus_list, val_list).....	348

8.4.6	get_modbus_input(iobus)	349
8.4.7	get_modbus_inputs(iobus_list)	350
8.4.8	get_modbus_inputs_list(iobus_list)	352
8.4.9	wait_modbus_input(iobus, val, timeout=None)	353
8.4.10	set_modbus_slave(address, val)	355
8.4.11	get_modbus_slave(address)	356
8.4.12	modbus_crc16(data)	357
8.4.13	modbus_send_make(send_data)	358
8.4.14	modbus_recv_check(recv_data)	359
8.5	Industrial Ethernet (EtherNet/IP,PROFINET)	360
8.5.1	set_output_register_bit(address, val)	360
8.5.2	set_output_register_int(address, val)	361
8.5.3	set_output_register_float(address, val)	362
8.5.4	get_output_register_bit(address)	363
8.5.5	get_output_register_int(address)	364
8.5.6	get_output_register_float(address)	365
8.5.7	get_input_register_bit(address)	366
8.5.8	get_input_register_int(address)	367
8.5.9	get_input_register_float(address)	368

9.	외부 비전 명령어	369
	개 요	369
9.1	vs_set_info(type)	370
9.2	vs_connect(ip_addr, port_num=9999)	371
9.3	vs_disconnect()	372
9.4	vs_get_job()	373
9.5	vs_set_job(job_name)	374
9.6	vs_trigger()	375
9.7	vs_set_init_pos(vision_posx_init, robot_posx_init, vs_pos=1)	376

9.8	vs_get_offset_pos(vision_posx_meas, vs_pos=1)	377
9.9	vs_request(cmd).....	378
9.10	vs_result()	379
9.11	통합예제 (DR_VS_COGNEX, DR_VS_SICK).....	380
9.12	통합예제 (DR_VS_CUSTOM)	381

10. 두산비전(SVM) 명령어 383

10.1	svm_connect(ip="192.168.137.5", port=20).....	383
10.2	svm_disconnect().....	384
10.3	svm_set_job(job_id).....	385
10.4	svm_get_robot_pose(job_id).....	386
10.5	svm_get_vision_info(job_id).....	387
10.6	svm_get_variable(tool_id, var_type).....	388
10.7	svm_get_offset_pos(posx_robot_init, job_id, tool_id).....	390
10.8	svm_set_init_pos_data(Id_list, Pos_list)	392
10.9	svm_set_tp_popup (svm_flag)	394
10.10	svm_set_led_brightness(value)	395
10.11	svm_get_led_brightness()	396
10.12	svm_set_camera_exp_val(value).....	397
10.13	svm_set_camera_gain_val(value).....	398
10.14	svm_set_camera_load(job_id).....	399
10.15	통합예제 (SVM).....	400

11. Application 명령어 402

11.1	Conveyor Tracking.....	402
11.1.1	set_conveyor(name).....	402

11.1.2	set_conveyor_ex(name="", conv_type=0, encoder_channel=1, triggering_mute_time=0.0, count_per_dist=5000, conv_coord=posx(0,0,0,0,0), ref=DR_BASE, conv_speed=100.0, speed_filter_size=500, min_dist=0.0, max_dist=1000.0, watch_window=100.0, out_tracking_dist=10.0).....	403
11.1.3	get_conveyor_obj(conv_id, timeout=None, container_type=DR_FIFO, obj_offset_coord=None).....	408
11.1.4	tracking_conveyor(conv_id).....	413
11.1.5	untracking_conveyor(conv_id, time=None).....	415

11.2 External Encoder 설정 명령어..... 417

11.2.1	set_extenc_polarity(channel, polarity_A, polarity_B, polarity_Z, polarity_S).....	417
11.2.2	set_extenc_mode(channel, mode_AB, pulse_AZ, mode_Z, mode_S, inverse_cnt).....	419
11.2.3	get_extenc_count(channel).....	421
11.2.4	clear_extenc_count(channel).....	422

12. A-Series 전용 명령어 423

12.1	get_function_input(index).....	423
12.2	flange_serial_open(baudrate=115200, bytesize=DR_EIGHTBITS, parity=DR_PARITY_NONE, stopbits = DR_STOPBITS_ONE).....	424
12.3	flange_serial_close().....	426
12.4	flange_serial_write(tx_data)	427
12.5	flange_serial_read().....	428

• DRL Basic Syntax

번호	함수명	설명
1.1	들여쓰기	각 코드 블록과의 구분을 위해 사용합니다.
1.2	주석	해당 코드를 부가적으로 설명하기 위해 사용합니다. 주석은 실제 코드 처리 과정에서 제외되기 때문에 소스 코드에 영향을 주지 않습니다.
1.3	변수명	자료의 값을 나타내기 위해 사용하며 문자, 숫자, 밑줄(_)로 구성할 수 있습니다. (첫 글자는 숫자 불가).
1.4	수치	int, float, complex
1.5	문자	String
1.6	list	결과 값을 나열한 것으로 순서를 인식합니다.
1.7	tuple	list와 동일하게 결과 값을 나열하지만 순서를 인식하지 않습니다.
1.8	tuple	list와 비슷하나, 읽기 전용으로 속도가 빠릅니다.
1.9	dictionary	Key와 Value를 지정하여 값을 나열합니다.
1.10	함수	함수명을 입력하여 값을 구할 때 사용합니다.
1.11	스코핑 룸	함수내에 지역변수에 해당하는 이름의 값이 없을 경우 LGB 규칙에 기반하여 이름을 찾을 수 있습니다.
1.12	인수 모드	기본 인수 값, 키워드 인수, 가변 인수를 사용합니다.
1.13	pass	어떠한 동작을 실행하지 않을 때 사용합니다.
1.14	if	조건문 함수로 if 문의 조건식에 참인가 거짓인가에 따라 elif와 else를 사용할 수 있습니다.
1.15	while	조건문 함수로 참과 거짓에 따라 반복작업을 실행합니다.
1.16	for	range로 반복 범위를 설정하여 반복작업을 실행합니다.
1.17	break	반복문 내부 블록을 빠져나올 수 있습니다.
1.18	continue	반복문 내부 블록을 더 이상 수행하지 않고 반복문 시작 지점으로 이동합니다.
1.19	반복문의 else	반복문 수행 도중 break 함수로 인하여 반복문이 종료되지 않고 끝까지 수행되었을 때 else 블록이 실행됩니다.

• 모션 관련 명령어

번호	함수명	설명
2.1	posj (q1=0, q2=0, q3=0, q4=0, q5=0, q6=0)	조인트 공간 각도를 좌표값으로 지정합니다.
2.2	posx (x=0, y=0, z=0, w=0, p=0, r=0)	작업 공간을 좌표값으로 지정합니다.
2.3	trans (pos, delta, ref, ref_out)	작업 공간에서의 addto에 해당하며 작업 공간 내의 한 객체의 delta만큼 동종 변환 후 값을 리턴합니다.
2.4	posb (seg_type, posx1, posx2=None, radius=0)	정속 블렌딩 모션(moveb, amoveb)의 입력 인자로, 각 경유점의 좌표와 단위 경로 형태(라인 또는 원호)의 정보를 갖는 posb는 블렌딩되는 trajectory의 단위 세그먼트 객체를 정의합니다.
2.5	fkin (pos, ref)	조인트 공간에서 포인트 또는 이에 상응하는 자료형(float[6])의 data를 입력받아 TCP의 좌표(task space 상의 객체)를 리턴합니다.
2.6	ikin (pos, sol_space, ref)	작업 공간에서 해당 로봇 자세에 상응하는 8개의 관절 중 sol_space에 해당하는 관절 위치로 리턴합니다.
2.7	addto (pos, add_val=None)	posj의 각 관절 값에 add_val 만큼 추가하여 새로운 posj 객체를 생성합니다.
2.8	set_velj (vel)	본 명령어를 사용한 이후 조인트 모션(movej, movejx, amovej, amovejx)에서의 전역 속도를 설정합니다.
2.9	set_accj (acc)	본 명령어를 사용한 이후의 조인트 모션(movej, movejx, amovej, amovejx)에서의 전역 가속도를 설정합니다.
2.10	set_velx (vel1, vel2)	작업 공간 모션의 속도를 전역적으로 설정합니다.
2.11	set_vels (vel)	작업 공간 모션의 속도를 전역적으로 설정합니다.
2.12	set_accx (acc1, acc2)	작업 공간 모션의 가속도를 전역적으로 설정합니다.
2.13	set_accx (acc)	작업 공간 모션의 가속도를 전역적으로 설정합니다.
2.14	set_tcp (name)	티치 팬던트에 등록된 tcp의 이름을 호출하여 현재 tcp로 설정합니다.
2.15	set_ref_coord (coord)	기준 좌표계를 설정합니다.
2.16	movej	로봇이 현재 관절위치에서 목표 관절위치(pos)로 이동합니다.
2.17	movel	로봇이 작업 공간 안에서 목표 위치(pos)로 직선을 따라 이동합니다.

번호	함수명	설명
2.18	movejx	로봇이 관절 공간 안에서 목표 위치(pos)로 이동합니다.
2.19	movec	작업공간(task space)을 기준으로 로봇이 현재 위치에서 경유점(pos1)를 지나 목표위치(pos2)까지의 원호 또는 지정한 각도까지 원호를 따라 이동합니다.
2.20	movesj	로봇이 현재 위치에서 pos_list로 입력된 관절공간(joint space)의 경유점들을 거쳐 목표위치(pos_list의 마지막 경유점)까지 연결되는 스플라인 곡선경로를 따라 이동합니다.
2.21	movesx	로봇이 현재 위치에서 pos_list로 입력된 작업공간(task space)의 경유점들을 거쳐 목표위치(pos_list의 마지막 경유점)까지 연결되는 스플라인 곡선경로를 따라 이동합니다.
2.22	moveb	하나 이상의 경로 세그먼트(line 또는 circle)를 인자로 갖는 list를 받아 각 세그먼트를 설정된 radius로 블렌딩하여 등속으로 이동합니다.
2.23	move_spiral	방사형 방향으로 반경이 증가하며 회전하는 Spiral motion과 축 방향으로 병행하며 이동합니다.
2.24	move_periodic	현재 위치에서 시작하는 상대 모션으로 입력된 기준 좌표계(ref)의 각 축(병진 및 회전)에 대한 Sine 함수 기반으로 주기 모션을 수행합니다.
2.25	move_home	기계적 홈 또는 사용자정의 홈 위치로 조인트모션으로 이동하여 호밍을 수행합니다.
2.26	amovel	비동기(async.)방식의 movevel모션으로 블렌딩을 위한 radius인자를 갖지 않는 점을 제외하고 movevel와 동일하게 작동합니다.
2.27	amovejx	비동기(async.)방식의 movejx모션으로 블렌딩을 위한 radius인자를 갖지 않는 점을 제외하고 movejx와 동일하게 작동합니다.
2.28	amovec	비동기(async.)방식의 movec모션으로 블렌딩을 위한 radius인자를 갖지 않는 점을 제외하고 movec와 동일하게 작동합니다.
2.29	amovesj	비동기(async.)방식의 movesj모션으로 async 처리 외에는 movesj()와 동일하게 동작합니다.
2.30	amovesx	비동기(async.)방식의 movesx모션으로 async 처리 외에는 movesx()와 동일하게 동작합니다.
2.31	amoveb	비동기(async.)방식의 moveb모션으로 async 처리 외에는 moveb()와 동일하게 동작하며 명령어를 수행한 후 바로 다음 라인을 수행합니다.

번호	함수명	설명
2.32	amove_spiral	비동기(async.)방식의 move_spiral모션으로 async 처리 외에는 move_spiral()와 동일하게 동작하며 명령어를 수행한 후 바로 다음 라인을 수행합니다.
2.33	amove_periodic	비동기(async.)방식의 move_periodic모션으로 async 처리 외에 move_periodic()와 동일하게 동작하며 명령어를 수행한 후 바로 다음 라인을 수행합니다.
2.34	mwait(time=0)	선행된 모션 명령어와 다음 라인의 모션 명령어 사이의 대기 시간을 설정합니다.
2.35	begin_blend(radius=0)	블렌딩 구간을 시작합니다.
2.36	end_blend()	블렌딩 구간을 종료합니다.
2.37	check_motion()	현재 진행 중인 모션의 상태를 확인합니다.
2.38	stop(st_mode)	수행 중인 모션을 정지합니다. 인자로 받는 st_mode에 따라 다르게 정지하며 Estop을 제외한 모든 Stop 모드는 현재 수행하고 있는 구간의 모션을 정지합니다.
2.39	change_operation_speed(speed)	작동 속도를 조정합니다.
2.40	wait_manual_guide()	프로그램 수행 중 핸드가이딩을 허용하며 사용자는 핸드가이딩을 완료한 후 아래의 두 가지 방법 중 하나를 수행하면 다음 명령어로 진행합니다
2.41	wait_nudge()	프로그램 수행 중 일시정지상태에서 사용자의 넛지입력(로봇에 외력을 가하는 행동)을 통한 작업재개를 허용합니다.
2.42	enable_alter_motion(n, mode,ref,limit_dPOS,limit_dPOS_per)	경로 수정 기능을 활성화 합니다.
2.43	alter_motion([x,y,z,rx,ry,rz])	입력 인자 pos에 해당하는 양만큼 경로 수정을 진행합니다.
2.44	disable_alter_motion()	경로 수정 기능을 비활성화 합니다.

• 제어 보조 명령어

번호	함수명	설명
3.1	get_control_mode()	현재 제어 모드를 리턴합니다.
3.2	get_control_space()	현재 제어 공간을 리턴합니다.

번호	함수명	설명
3.3	get_current_posj()	현재 관절 각도를 리턴합니다.
3.4	get_current_velj()	현재 관절 속도를 리턴합니다.
3.5	get_desired_posj()	현재의 목표(target) 관절각을 리턴합니다.
3.6	get_desired_velj()	현재의 목표(target) 관절 속도를 리턴합니다.
3.7	get_current_posx(ref)	현재 태스크 좌표계의 자세와 solution space를 리턴합니다.
3.8	get_current_tool_flange_posx(ref)	현재 툴 플랜지의 자세를 리턴합니다.
3.9	get_current_velx(ref)	현재 툴 속도를 리턴합니다.
3.10	get_desired_posx(ref)	현재의 툴의 목표(target) 자세를 리턴합니다.
3.11	get_desired_velx(ref)	현재의 툴의 목표(target) 속도를 리턴합니다.
3.12	get_current_solution_space()	현재 solution space 값을 리턴합니다.
3.13	get_current_rotm(ref)	현재 툴의 방향과 행렬을 리턴합니다.
3.14	get_joint_torque()	현재 조인트의 센서 토크 값을 리턴합니다.
3.15	get_external_torque()	현재 각 관절에서 외력에 의해 발생하는 토크 값을 리턴합니다.
3.16	get_tool_force(ref)	현재 툴에 작용하는 외력 값을 리턴합니다.
3.17	get_solution_space(pos)	Solution space의 값을 구합니다.
3.18	get_orientation_error(xd, xc, axis)	axis에 대한 임의의 pose xd와 xc 사이의 Orientation error 값을 리턴합니다.

• 기타 설정 및 안전 관련 명령어

4.1	get_workpiece_weight()	작업물의 무게를 측정하여 리턴합니다.
4.2	reset_workpiece_weight()	소재의 무게를 측정하기 전 알고리즘을 초기화를 위해 소재의 무게정보를 초기화합니다.
4.3	set_tool(name)	티치팬던트에 등록된 툴 정보 중에서 입력된 name의 tool을 활성화 합니다.

4.4	set_tool(name)	티치팬던트에 등록된 툴 형상 정보 중에서 입력된 name의 tool 형상을 활성화 합니다.
4.5	set_singularity_handling (mode)	task motion에서 특이점의 영향으로 path deviation이 발생할 경우 대응 정책을 사용자가 선택 할 수 있도록 합니다.

• 힘/강성 제어 및 기타 사용자 편의 기능

번호	함수명	설명
5.1	parallel_axis(x1, x2, x3, axis, ref)	get_normal(x1, x2, x3)에 의해 얻어지는 normal vector와 Tool Frame 중 인자 axis로 받는 축을 일치시킵니다.
5.2	parallel_axis(vect, axis, ref)	주어진 vect 방향으로 Tool Frame 중 axis로 받는 축을 일치시킵니다.
5.3	align_axis(x1, x2, x3, pos, axis, ref)	get_normal(x1, x2, x3)에 의해 얻어지는 normal vector와 Tool Frame 중 인자 axis로 받는 축을 일치시킵니다.
5.4	align_axis(vect, pos, axis, ref)	주어진 vect 방향으로 Tool Frame 중 axis로 받는 축을 일치시킵니다.
5.5	is_done_bolt_tightening(m=0, timeout=0, axis=None)	툴의 조임 토크를 모니터링하여 주어진 시간 내에 설정된 토크(m)에 도달한 경우는 True를 리턴하고, 주어진 시간을 초과한 경우에는 False를 리턴합니다.
5.6	release_compliance_ctrl()	Compliance control을 종료하고 현재 위치에서 위치 제어를 시작합니다.
5.7	task_compliance_ctrl	기준에 설정한 기준 좌표계를 기준으로 태스크 Compliance control을 시작합니다.
5.8	set_stiffnessx	강성값을 설정합니다.
5.9	calc_coord	최대 4개의 로봇 자세(x1~x4), 입력 모드(mod) 및 기준 좌표계(ref)를 기반으로 새로운 직교 좌표계를 계산할 수 있습니다.
5.10	set_user_cart_coord	위치정보(pos)와 기준 좌표계(ref) 기반의 새로운 사용자좌표계를 설정할 수 있습니다.
5.11	set_user_cart_coord	사용자가 x1, x2, x3를 사용하여 새로운 직교 좌표계를 설정할 수 있습니다.
5.12	set_user_cart_coord	사용자가 u1과 v1를 사용하여 새로운 직교 좌표계를 설정할 수 있습니다.

번호	함수명	설명
5.13	overwrite_user_cart_coord	요청하는 ID(id)의 사용자 좌표계의 좌표계 위치(pos), 기준 좌표계(ref) 정보를 변경합니다.
5.14	get_user_cart_coord	해당하는 ID(id)의 사용자 좌표계의 정보인 참조 기준 및 위치정보를 조회합니다.
5.15	set_desired_force	힘 제어 목표값, 힘 제어 방향, 힘 목표값, 변화 시간을 설정합니다.
5.16	release_force(time=0)	힘 제어 목표값을 time 값 동안 0으로 줄이고 작업 공간을 순응 제어로 리턴합니다.
5.17	check_position_condition	주어진 위치 상태를 확인합니다.
5.18	check_force_condition	주어진 힘 상태를 확인합니다. 단, 힘의 방향은 고려하지 않고 크기로만 비교합니다.
5.19	check_orientation_condition	현재 로봇 엔드이펙터의 자세 정보와 주어진 위치 자세 간 차이의 상태를 확인합니다.
5.20	check_orientation_condition	현재 로봇 엔드이펙터의 자세와 회전각 범위 차이에 대한 상태를 확인합니다.
5.21	coord_transform(pose_in, ref_in, ref_out)	'ref_in' 기준 좌표계에서 표현되는 'pose_in' Task 좌표를 'ref_out' 기준 좌표계에서 표현되는 Task 좌표로 변환하여, 출력합니다.

• 시스템 명령어

번호	함수명	설명
6.1.1	set_digital_output(index, val =None)	컨트롤러의 디지털 접점에서 신호를 내보내기 위한 명령문입니다.
6.1.2	set_digital_outputs(bit_list)	컨트롤러의 디지털 출력 복수개의 접점에서 신호를 내보내기 위한 명령문입니다.
6.1.3	set_digital_outputs(bit_start, bit_end, val)	컨트롤러의 디지털 출력 시작 접점(bit_start)부터 마지막 접점(bit_end)까지 한 번에 복수 신호를 내보내기 위한 명령문입니다.
6.1.4	get_digital_input(index)	컨트롤러의 디지털 입력 점접에서 신호를 불러오기 위한 명령문으로 디지털 입력 점접 값을 읽습니다.
6.1.5	get_digital_inputs(bit_list)	컨트롤러의 디지털 입력 복수 개의 접점에서 신호를 불러오기 위한 명령문입니다.

번호	함수명	설명
6.1.6	get_digital_inputs (bit_start, bit_end)	컨트롤러의 디지털 입력 시작 접점(start_index)부터 마지막 접점(end_index)까지 한 번에 복수 신호를 불러오기 위한 명령문입니다.
6.1.7	wait_digital_input (index, val, timeout=None)	컨트롤러의 디지털 입력 레지스터의 신호값이 val(ON or OFF)이 될 때까지 대기합니다.
6.1.8	set_tool_digital_output (index, val=None)	로봇 툴의 신호를 디지털 접점에서 내보내기 위한 명령문입니다.
6.1.9	set_tool_digital_outputs (bit_list)	로봇 툴의 신호를 디지털 접점에서 내보내기 위한 명령문으로 bit_list에 정의된 접점들의 디지털 신호를 한 번에 출력할 수 있습니다
6.1.10	set_tool_digital_outputs (bit_start, bit_end, val)	로봇 툴의 신호를 디지털 접점에서 내보내기 위한 명령문으로 시작 접점(bit_start)부터 마지막 접점(bit_end)까지 복수 신호를 한 번에 출력할 수 있습니다.
6.1.11	get_tool_digital_input (index)	로봇 툴의 신호를 디지털 접점에서 불러오기 위한 명령문입니다.
6.1.12	get_tool_digital_inputs (bit_list)	로봇 툴의 신호를 디지털 입력 접점에서 불러오기 위한 명령문으로 bit_list에 정의된 접점들의 디지털 신호를 한 번에 입력할 수 있습니다.
6.1.13	get_tool_digital_inputs (bit_start, bit_end)	로봇 툴의 신호를 디지털 접점에서 불러오기 위한 명령문으로 시작 접점(start_index)부터 마지막 접점(end_index)까지 복수 신호를 한 번에 입력할 수 있습니다.
6.1.14	wait_tool_digital_input (index, val, timeout=None)	로봇 툴의 디지털 입력 신호값이 val(ON or OFF)이 될 때까지 대기합니다.
6.1.15	set_mode_analog_output (ch, mod)	컨트롤러 아날로그 출력에 대한 채널 모드를 설정합니다.
6.1.16	set_mode_analog_input (ch, mod)	컨트롤러 아날로그 입력에 대한 채널 모드를 설정합니다.
6.1.17	set_analog_output (ch, val)	컨트롤러 아날로그 출력에 해당하는 채널의 값을 출력합니다.
6.1.18	get_analog_input (ch)	컨트롤러 아날로그 입력에 해당하는 채널의 값을 불러옵니다.
6.2.1	tp_popup (message, pm_type=DR_PM_MESSAGE, button_type=0)	터치 팬던트를 통해 사용자에게 메시지를 제공합니다.

번호	함수명	설명
6.2.2	tp_log(message)	티치 팬던트에 사용자가 작성한 로그를 기록합니다.
6.2.3	tp_progress(cur_progress, total_progress)	티치 팬던트를 통해 사용자에게 메시지를 제공합니다.
6.2.4	tp_get_user_input(message, input_type)	티치 팬던트를 통해 사용자 입력 정보를 받습니다.
6.3.1	thread_run(th_func_name, loop=False)	Thread를 생성하여 수행하며 Thread가 수행할 기능은 th_func_name에 지정된 함수에 따라 결정됩니다.
6.3.2	thread_stop(th_id)	Thread를 종료합니다.
6.3.3	thread_pause(th_id)	Thread를 일시 정지합니다.
6.3.4	thread_resume(th_id)	일시 정지된 Thread를 다시 시작합니다.
6.3.5	thread_state(th_id)	Thread의 상태를 확인합니다.
6.3.6	통합 예제	Thread 통합 예제
6.4.1	wait(time)	지정된 시간만큼 대기합니다.
6.4.2	exit()	현재 수행 중인 프로그램을 종료합니다.
6.4.3	sub_program_run(name)	별도의 파일로 작성된 서브프로그램을 실행합니다.

수학 함수

번호	함수명	설명
7.1	sin(x)	x radians의 sine 값을 리턴합니다.
7.2	cos(x)	x radians의 cosine 값을 리턴합니다.
7.3	tan(x)	x radians의 tangent 값을 리턴합니다.
7.4	asin(x)	x radians의 arc sine of 값을 리턴합니다.
7.5	acos(x)	x radians의 arc cosine of 값을 리턴합니다.
7.6	atan(x)	x radians의 arc tangent of 값을 리턴합니다.
7.7	atan2(y, x)	y/x radians의 arc tangent of 값을 리턴합니다.
7.8	ceil(x)	x값보다 큰 정수 중 가장 작은 정수 값을 리턴합니다.
7.9	floor(x)	x값보다 작은 정수 중 가장 큰 정수 값을 리턴합니다.

번호	함수명	설명
7.10	pow(x, y)	Return x raised to the power of y.
7.11	sqrt(x)	x의 제곱근을 리턴합니다.
7.12	log(x, b)	x의 로그를 밑수 b로 리턴합니다.
7.13	d2r(x)	x radians 값을 degree로 리턴합니다.
7.14	r2d(x)	x radians 값을 degree로 리턴합니다.
7.15	norm(x)	x의 L2 norm을 리턴합니다.
7.16	random()	랜덤 번호를 리턴합니다.
7.17	rotx(angle)	x축을 기준으로 angle 값만큼 회전시키는 회전행렬을 리턴합니다
7.18	roty(angle)	y축을 기준으로 angle 값만큼 회전시키는 회전행렬을 리턴합니다.
7.19	rotz(angle)	z축을 기준으로 angle 값만큼 회전시키는 회전행렬을 리턴합니다.
7.20	rotm2eul(rotm)	회전행렬을 받아 Euler angle(zyz order)을 degree 값으로 리턴합니다.
7.21	rotm2rotvec(rotm)	회전행렬을 받아 rotation vector(angle/axis representation)를 리턴합니다.
7.22	eul2rotm([alpha,beta,gamma])	Euler angle(zyz order)을 회전행렬로 변환합니다.
7.23	eul2rotvec([alpha,beta,gamma])	Euler angle(zyz order)을 rotation vector로 변환합니다.
7.24	rotvec2eul([rx,ry,rz])	Rotation vector를 Euler Angle(zyz)로 변환합니다.
7.25	rotvec2rotm([rx,ry,rz])	rotation vector를 rotation matrix로 변환합니다.
7.26	htrans(posx1,posx2)	posx1과 posx2로부터 구한 Homogeneous Transformation matrix를 T1, T2라 할 때 T1*T2에 해당하는 자세를 리턴합니다.
7.27	get_intermediate_pose(posx1, posx2, alpha)	posx1으로부터 posx2로의 linear transition 중 alpha에 위치한 posx를 리턴합니다.
7.28	get_distance(posx1, posx2)	두 pose 포지션 사이의 거리를 [mm] 단위로 리턴합니다.
7.29	get_normal(x1, x2, x3)	3개의 작업 공간 지점(posx)으로 이루어진 평면의 normal vector를 리턴합니다. 방향은 시계방향으로 기준입니다.
7.30	add_pose(posx1, posx2)	두 자세의 합을 구합니다.

번호	함수명	설명
7.31	subtract_pose (posx1, posx2)	두 자세의 차이를 구합니다.
7.32	inverse_pose (posx1)	posx의 inverse에 해당하는 posx를 리턴합니다.
7.33	dot_pose (posx1, posx2)	두 개의 자세가 주어졌을 때 translation 성분의 내적을 구합니다.
7.34	cross_pose (posx1, posx2)	두 개의 자세가 주어졌을 때 translation 성분의 외적을 구합니다.
7.35	unit_pose (posx1)	주어진 posx translation 성분의 단위 백터를 구합니다.

• 외부 통신 명령어

번호	함수명	설명
8.1.1	serial_open (port=None, baudrate=115200, bytesize=DR_EIGHTBITS, parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE)	Serial 통신 포트를 오픈합니다.
8.1.2	serial_close (ser)	Serial 통신 포트를 닫습니다.
8.1.3	serial_state (ser)	Serial 통신 포트의 상태를 리턴합니다.
8.1.4	serial_set_inter_byte_timeout (ser, timeout=None)	포트에 읽기/쓰기를 수행할 때 바이트 간(inter-byte)의 timeout을 설정합니다.
8.1.5	serial_write (ser, tx_data)	Serial 포트에 데이터(tx_data)를 씁니다.
8.1.6	serial_read (ser, length=-1, timeout=-1)	Serial 포트에서 데이터를 읽어옵니다.
8.1.7	serial_get_count()	연결된 USB to Serial에 연결된 장치의 개수를 읽어옵니다.
8.1.8	serial_get_info (id)	연결된 USB to Serial의 포트정보, 장치이름을 읽어옵니다.
8.1.9	통합 예제	Serial 통합 예제
8.2.1	client_socket_open (ip, port)	소켓을 생성하고, 서버(ip, port)에 연결을 시도합니다.
8.2.2	client_socket_close (sock)	서버와의 통신을 종료합니다.
8.2.3	client_socket_state (sock)	소켓의 상태를 리턴합니다.
8.2.4	client_socket_write (sock, tx_data)	서버에 데이터를 송신합니다.

번호	함수명	설명
8.2.5	client_socket_read(sock, length=-1, timeout=-1)	서버로부터의 데이터를 수신합니다.
8.2.6	통합 예제	Client socket 통합 예제
8.3.1	server_socket_open(port)	소켓을 생성하고, Client와의 연결을 대기합니다. Client와 연결되면 연결된 소켓을 반환합니다.
8.3.2	server_socket_close(sock)	Client와의 통신을 종료합니다.
8.3.3	server_socket_state(sock)	소켓의 상태를 리턴합니다.
8.3.4	server_socket_write(sock, tx_data)	Client에 데이터를 송신합니다.
8.3.5	server_socket_read(sock, length=-1, timeout=-1)	Client로부터 데이터를 수신합니다.
8.3.6	통합 예제	Server socket 통합 예제
8.4.1	add_modbus_signal(ip, port, name, reg_type, index, value=0, slaveid=255)	Modbus의 신호를 등록합니다.
8.4.2	add_modbus_rtu_signal(slaveid=1, port=None, baudrate=115200, bytesize=DR_EIGHTBITS, parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE, name, reg_type, index, value=0)	ModbusRTU의 신호를 등록합니다
8.4.3	del_modbus_signal(name)	등록된 Modbus의 신호를 삭제합니다
8.4.4	set_modbus_output(iobus, val)	외부 Modbus 장치에 신호를 보내내기 위한 명령어입니다.
8.4.5	get_modbus_input(iobus)	Modbus 장치에서 신호를 읽어오기 위한 명령어입니다.
8.4.6	set_modbus_outputs(iobus_list, val_list)	Modbus Slave장치에 복수 개의 신호를 보내내기 위한 명령입니다
8.4.7	get_modbus_inputs(iobus_list)	외부 Modbus 디지털 I/O 장치의 디지털 입력 접점의 복수 개 신호를 읽어오기 위한 명령어입니다.
8.4.8	get_modbus_inputs_list(iobus_list)	Modbus Slave 장치의 Register Type의 복수 개 신호를 읽어오기 위한 명령어입니다.

번호	함수명	설명
8.4.9	wait_modbus_input (ibus, val, timeout=None)	Modbus 디지털 I/O 에서 지정한 신호 접점 값이 val(ON or OFF)이 될 때까지 대기합니다.
8.4.10	set_modbus_slave (address, val)	ModbusTCP Slave의 Gerneral Purpose Register 영역에 값을 내보내기 위해 사용합니다.
8.4.11	get_modbus_slave (address)	ModbusTCP Slave의 Gerneral Purpose Register 영역을 접근하여 값을 가져오기 위해 사용합니다.
8.5.1	set_output_register_bit (address, val)	Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Output영역의 Bit General Purpose Register의 값을 설정
8.5.2	set_output_register_int (address, val)	Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Output영역의 Int General Purpose Register의 값을 설정
8.5.3	set_output_register_float (address, val)	Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Output영역의 Float General Purpose Register의 값을 설정
8.5.4	get_output_register_bit (address)	Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Output영역의 Bit General Purpose Register의 값을 읽어오기
8.5.5	get_output_register_int (address)	Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Output영역의 Int General Purpose Register의 값을 읽어오기
8.5.6	get_output_register_float (address)	Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Output영역의 Float General Purpose Register의 값을 읽어오기
8.5.7	get_input_register_bit (address)	Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Input영역의 Bit General Purpose Register의 값을 읽어오기
8.5.8	get_input_register_int (address)	Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Input영역의 Int General Purpose Register의 값을 읽어오기
8.5.9	get_input_register_float (address)	Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Input영역의 Float General Purpose Register의 값을 읽어오기

• 외부 비전 명령어

번호	함수명	설명
9.1	vs_set_info(type)	사용할 비전시스템의 종류를 설정한다.
9.2	vs_connect(ip_addr, port_num=9999)	비전시스템의 통신을 연결한다.
9.3	vs_disconnect()	비전시스템의 통신을 해제한다.
9.4	vs_get_job()	비전시스템에 현재 로딩되어 있는 작업명을 불러온다. (*VS_TYPE: DR_VS_COGNEX, DR_VS_SICK)
9.5	vs_set_job(job_name)	입력된 작업을 비전시스템에 로딩한다. (*VS_TYPE: DR_VS_COGNEX, DR_VS_SICK)
9.6	vs_trigger()	비전시스템에 측정명령을 전달한다. 측정이 성공하면, 측정결과값을 리턴한다. (*VS_TYPE: DR_VS_COGNEX, DR_VS_SICK)
9.7	vs_set_init_pos(vision_posx_init, robot_posx_init, vs_pos=1)	비전 가이던스 작업을 수행할 대상물의 초기 위치정보를 입력한다. (*VS_TYPE: DR_VS_COGNEX, DR_VS_SICK)
9.8	vs_get_offset_pos(vision_posx _meas, vs_pos=1)	비전 시스템에서 측정된 좌표값을 이용하여 옵셋(가이드)된 로봇작업 좌표를 산출한다. 사전에 vs_set_init_pos를 통해 초기값이 입력되어 있어야 한다.
9.9	vs_request(cmd)	Vision System에 요청할 기능을 설정합니다. (*VS_TYPE: DR_VS_CUSTOM)
9.10	vs_result()	Vision System의 처리 결과를 가져옵니다. (*VS_TYPE: DR_VS_CUSTOM)
9.13	통합 예제	통합 예제(DR_VS_CUSTOM)

• 두산 비전(SVM) 명령어

번호	함수명	설명
10.1	svm_connect(ip="192.168.137.5" , port=20)	SVM의 통신을 연결한다.
10.2	svm_disconnect()	SVM의 통신을 해제한다.
10.3	svm_set_job(job_id)	입력된 id에 해당하는 비전작업을 SVM에 로딩한다.

번호	함수명	설명
10.4	svm_get_robot_pose(job_id)	입력된 비전작업에 설정되어 있는 로봇자세정보(조인트 좌표계)를 불러온다.
10.5	svm_get_vision_info(job_id)	입력된 비전작업에 해당하는 측정명령을 수행한다.
10.6	svm_get_variable(tool_id, var_type)	svm_get_vision_info를 수행하여 물체검출/측정에 성공(1)한 경우, 검출/측정 데이터를 불러온다.
10.7	svm_get_offset_pos(posx_robot_init, job_id, tool_id)	사용자가 입력한 로봇 작업 좌표정보에 비전 측정결과를 반영한 로봇 작업 좌표정보를 불러온다.
10.8	svm_set_init_pos_data(id_list, Pos_list)	비전 가이던스 작업을 수행할 대상물의 초기 id_list와 posx_list 정보를 입력한다.
10.9	svm_set_tp_popup (svm_flag)	SVM 오류 발생 시 tp_popup을 띄울 것인지 선택하는 함수이다.
10.10	svm_set_led_brightness(value)	SVM LED 값을 설정한 값으로 변경한다.
10.11	svm_get_led_brihtness()	SVM LED 현재값을 불러온다.
10.12	svm_set_camera_exp_val(value)	SVM 노출 값을 설정한 값으로 변경한다.
10.13	svm_set_camera_gain_val(value)	SVM 개인 값을 설정한 값으로 변경한다.
10.14	svm_set_camera_load(job_id)	SVM에 저장된 LED/exp/gain/초점 정보를 불러온다.
10.15	통합 예제	SVM 통합 예제

• Application 명령어

번호	함수명	설명
11.1.1	set_conveyor(name)	Conveyor Tracking Application을 시작할 수 있도록 Conveyor를 설정하고 ID를 획득합니다.
11.1.2	set_conveyor_ex(name, conv_type, encoder_channel, triggering_mute_time, count_per_dist, conv_coord, obj_offset_coord, conv_speed, speed_filter_size, min_dist, max_dist, watch_window, out_tracking_dist)	Conveyor Tracking Application을 시작할 수 있도록 Conveyor를 설정하고 ID를 획득합니다.
11.1.3	get_conveyor_obj(conv_id, timeout=None, container_type=DR_FIFO, obj_offset_coord=None)	해당 Conveyor로부터 작업 가능한 작업물 좌표계 ID를 리턴합니다.

번호	함수명	설명
11.1.4	tracking_conveyor(conv_id)	로봇이 Conveyor Tracking을 시작합니다.
11.1.5	untracking_conveyor(conv_id, time=None)	로봇이 Conveyor Tracking을 종료합니다.
11.2.1	set_extenc_polarity(channel, polarity_A, polarity_B, polarity_Z, polarity_S)	해당 채널 엔코더의 A, B, Z상 극상과 S상의 트리거 방식을 설정한다.
11.2.2	set_extenc_mode(channel, mode_AB, pulse_AZ, mode_Z, mode_S, inverse_cnt)	해당 채널 엔코더의 A, B, Z, S상의 동작 모드를 설정한다.
11.2.3	get_extenc_count(channel)	해당 채널 엔코더의 Count를 구한다.
11.2.4	clear_extenc_count(channel)	해당 채널 엔코더의 카운트 값을 0으로 초기화한다.

머리말

본 매뉴얼은 총 12장으로 구성되어 있으며 1장부터 11장까지는 M 시리즈 모델과 A 시리즈 모델에 공통으로 적용되는 명령어이며, 12장은 A 시리즈 모델에만 적용되는 명령어입니다.

본 매뉴얼의 내용은 작성된 시점을 기준으로 하며, 제품에 대한 정보는 사용자에게 사전 고지 없이 변경될 수 있습니다.

개정된 매뉴얼에 대한 상세 정보는 Robot LAB (<https://robotlab.doosanrobotics.com/>)에서 확인하십시오.

저작권

본 매뉴얼의 모든 내용과 도안에 대한 저작권 및 지적재산권은 두산로보틱스에 있습니다. 따라서 두산로보틱스의 서면 허가 없이 사용, 복사, 유포하는 어떠한 행위도 금지됩니다. 또한 특허권을 오용하거나 변용하는데 따르는 책임은 전적으로 사용자에게 있습니다.

본 매뉴얼은 신뢰할 수 있는 정보지만 오류 또는 오탈자로 인한 손실에 대해 어떠한 책임도 지지 않습니다. 제품 개선에 따라 매뉴얼에 포함된 정보는 예고 없이 변경될 수 있습니다.

개정된 매뉴얼에 관한 상세 정보는 두산로보틱스 홈페이지(www.doosanrobotics.com)에서 확인하십시오.

© 2018 Doosan Robotics Inc., All rights reserved

1. DRL Basic Syntax

⚠ 주의

DRL의 문법은 python과 동일하지만, Python의 모든 문법과 기능을 포함하지 않습니다.
본 매뉴얼에 기술된 내용만 지원합니다.

1.1 들여쓰기

▪ 기능

각 코드 블록과의 구분을 위해 사용합니다. 들여쓰기를 준수하지 않을 시 오류가 발생합니다.

- 들여 쓰기는 각 코드 블록을 구분하는 데 사용됩니다.
- 들여 쓰기에는 2 칸, 4 칸 또는 탭 문자를 사용할 수 있습니다.
- 블록의 경우 동일한 유형의 들여 쓰기가 사용되어야합니다.

▪ 예제

코드블럭1

[TAB]코드블럭2

예)

```
if x == 3:
```

```
    y += 1
```

```
# No Error
```

```
def fnSum(x,y):
```

```
[공백4]sum = x + y
```

```
[공백4]return sum
```

```
# No Error
```

```
def fnSubtract(x,y):
```

```
[TAB]diff = x - y
[TAB]return diff

# 들여쓰기 오류
def fnProduct(x,y):
[공백4]product = x * y
[TAB]return product
```

1.2 주석

▪ 기능

해당 코드를 부가적으로 설명하기 위해 사용합니다. 주석은 실제 코드 처리 과정에서 제외되기 때문에 소스 코드에 영향을 주지 않습니다.

이후로는 주석으로 인식합니다. ””로 시작해서 ””로 끝나는 블록은 주석으로 인식합니다.

- 주석은 코드에 대한 추가 설명을 제공하는 데 사용됩니다. 주석은 코드 처리에서 제외되므로 소스 코드에 영향을 미치지 않습니다.
- "#" 다음의 명령문은 주석으로 인식됩니다.
- ””로 시작하고 ””로 끝나는 블록은 주석으로 인식됩니다.

▪ 예제

```
# 주석 예1
```

```
""
```

```
주석 예2
```

```
""
```

1.3 변수명

▪ 가능

- 자료의 값을 나타내기 위해 사용하며 문자, 숫자, 밑줄(_)로 구성할 수 있습니다. 문자, 숫자, 밑줄(_)로 구성합니다(첫 글자는 숫자 불가).
- 대소문자를 구분합니다.
- 변수 명을 예약어나 DRL 명령어 이름과 동일하게 사용하면 인터프리터에 치명적인 에러가 발생 합니다.

이를 피하기 위해서 가급적 "var_" 이란 prefix를 사용하여 변수명을 사용하시기 바랍니다.

⚠ 주의

하기 예약어들을 변수명이나 함수명으로 절대로 사용하지 마십시오.

and	assert	break	class	continue
def	del	elif	else	except
exec	finally	for	from	global
if	import	in	is	lambda
list	not	open	or	pass
print	raise	return	try	while
with	yield			

▪ 예제

```
friend = 10
Friend = 1
_myFriend = None
```

Pass = 10 # Syntax error

movej = 0 # movej는 DRL 명령어 이름으로 변수로 사용하면 안됩니다.

1.4 수치

▪ 가능

int, float, complex로 수치를 입력합니다.

▪ 예제

```
10, 0x10, 0o10, 0b10
```

```
3.14, 314e-2
```

```
x = 3-4j
```

```
int_value = 10
```

```
hexa_value = 0x10
```

```
octa_value = 0o10
```

```
binary_value = 0b10
```

```
double_value = 3.14
```

```
double value = 314e-2
```

```
complex_value = 3-4j
```

1.5 문자

1.5.1 String

■ 기능

모든 문자열은 기본적으로 유니코드입니다.

- Escape characters

\n: New line

\t: Tab

\r: Carriage return

\0: Null string

\\\: back slash(\) in string

\': single quote mark in string

\": double quote mark in string

- String concatenation: "py" + "ton" → "python"
- String repetition: "py" * 3 → "pypy"
- String indexing: "python" [0] → "p"
- String slicing: "python" [1:4] → "yth"

■ 예제

```
"string1"  
'string2'  
  
tp_log("st"+ "ring")  
    #expected result: string  
tp_log("str" * 3)  
    #expected result: strstrstr  
tp_log("line1\nline2")  
    #expected result: line1  
    # line2  
tp_log("\\"+ "string\\")
```

```
#expected result: "string"  
tp_log("str"[0])  
    #expected result: s  
tp_log("string"[1:3])  
    #expected result: tr
```

1.5.2 +, *

- 예제

```
"Doosan"+ "Robotics" ➔ "DoosanRobotics"  
"Doo"* 3 ➔ "DooDooDoo"
```

1.5.3 인덱싱 & 슬라이싱

- 예제

```
" Doosan" [0] ➔ "D"  
" Doosan" [1:4] ➔ "oos"
```

1.6 list

▪ 가능

- 값들의 나열로, 각 item은 변경 가능하며 순서가 있습니다.
- 인덱싱 및 슬라이싱이 가능합니다.
- append, insert, extend, +연산자
- count, remove, sort 연산자

▪ 예제

```
colors = ["red", "green", "blue"]
tp_log(colors[0]+","+colors[1]+","+colors[2])
#expected print result: red,green,blue
```

```
numbers = [1, 3, 5, 7, 9]
sum = 0
for number in numbers:
    sum += number
tp_log( str(sum) )
#expected result: 25
```

1.7 tuple

- **기능**

list와 비슷하나, 읽기 전용으로 속도가 빠릅니다.

- **예제**

```
colors = ("red", "green", "blue")
numbers = (1, 3, 5, 7, 9)

def fnMinMax(numbers):
    numbers.sort()
    return (numbers[0], numbers[-1])
minmax = fnMinMax([4,1,2,9,5,7])
tp_log("Min Value= " + str(minmax[0]))
    #expected result: Min Value = 1
tp_log("Max Value= " + str(minmax[1]))
    #expected result: Max Value = 9
```

1.8 dictionary

- **기능**

Key와 Value를 지정하여 값을 나열합니다.

```
d = dict(a = 1, b = 3, c = 5)

colors = dict()
colors["cherry"] = "red"

ages = {'Kim':35, 'Lee':38, 'Chang':37}
tp_log("Ages of Kim = " + str(ages['Kim']))
#expected print result: Ages of Kim = 35
```

1.9 함수

▪ 기능

- 선언: def로 시작하고 콜론(:)으로 끝냅니다.
- 함수의 시작과 끝은 코드의 들여쓰기로 구분합니다.
- interface/implementation을 따로 구분하지 않습니다. 단, 사용 전에 미리 정의되어야 합니다.
- 함수 명을 예약어나 DRL 명령어 이름과 동일하게 사용하면 인터프리터에 치명적인 에러가 발생합니다.

이를 피하기 위해서 가급적 "fn_" 이란 prefix를 사용하여 함수 이름을 사용하시기 바랍니다.

⚠ 주의

하기 예약어들을 변수명이나 함수명으로 절대로 사용하지 마십시오.

And	assert	break	class	continue
Def	del	elif	else	except
exec	finally	for	from	global
If	import	in	is	lambda
list	Not	open	or	pass
print	raise	return	try	while
with	yield			

▪ 문법

```
def <함수명>(인수1, 인수2, ... 인수N):
    <구문> ...
    return <반환값>
```

```
#예)
def fn_Times(a, b):
    return a * b

fn_Times (10, 10)
```

```

def fn_Times(a, b):
    return a * b

tp_log(str(fn_Times(10, 5)))
#expected result: 50

def movej():    #movej는 DRL 명령어로 함수명으로 사용하면 안 됩니다.
    return 0

```

1.10 스코핑 룰

▪ 기능

- 함수내에 지역변수에 해당하는 이름의 값이 없을 경우 LGB 규칙에 기반하여 이름을 찾을 수 있습니다.
- 이름공간: 변수의 이름이 저장되어 있는 장소
- Local scope: 함수 내부의 이름 공간, 지역 영역
- Global scope: 함수 밖의 영역, 전역 영역
- Built-in scope: 파이썬 자체에서 정의한 내용에 대한 영역, 내장 영역
- LGB 규칙: 변수 이름을 찾는 순서 local → global → built-in

▪ 예제

```

# Error: can't find simple_pi in circle_area
simple_pi = 3.14
def circle_area(r):
    return r*r*simple_pi

# simple_pi should be declared as global if it is used in circle_area_ok
def circle_area_ok(r):
    global simple_pi
    return r*r*simple_pi

tp_log(str(circle_area(3.0)))

```

```
#expected result: 28.26
```

1.11 인수 모드

- **기능**

기본 인수 값, 키워드 인수, 가변 인수를 사용합니다.

- **예제**

```
def fn_Times(a = 10, b = 20):
    return a * b

#Example - Default parameter value
tp_log(str(fn_Times(5)))
#expected result: 100

#Example - Keyword parameter
tp_log(str(fn_Times(b=5)))
#expected result: 50
tp_log(str(fn_Times(a=5, b=5)))
#expected result: 25

#Example - arbitrary parameter
def fn_myUnion(*args):
    for arg in args:
        tp_log(str(arg))
fn_myUnion("red", 1)
#expected print result: red
#
```

- 예제 - 기본 인수 값

```
def fn_Times(a = 10, b = 20):
    return a * b

fn_Times(5)
```

- 예제 - 키워드 인수

```
def fn_Times(a = 10, b = 20)
    return a * b

fn_Times(a=5, b=5)
```

- 예제 - 가변 인수

```
def fn_myUnion(*ar)
.....
fn_myUnion("red", "white", "black")
```

1.12 pass

- 기능

어떠한 동작을 실행하지 않을 때 사용합니다.

- 예제

```
while True:
    pass #pass means empty statement, so while statement continues to run.
    tp_log("This line never reached")
```

1.13 if

▪ 기능

조건문 함수로 if 문의 조건식에 참인가 거짓인가에 따라 elif와 else를 사용할 수 있습니다.

▪ 문법

if <조건식>:

 <구문>

 if <조건식1>:

 <구문1>

 elif <조건식2>:

 <구문2>

 else:

 <구문3>

▪ 예제 - if, elif, else

```
numbers = [2,5,7]
for number in numbers:
    if number%2==0:
        tp_log(str(number) + " is even")
    else:
        tp_log (str(number) + " is odd")
#expected result:
#2 is even
#5 is odd
#7 is odd
```

1.14 while

- **기능**

조건문 함수로 참과 거짓에 따라 반복작업을 실행합니다.

- **문법**

while <조건식>:

 <구문>

- **예제**

```
sum = 0
cnt = 1
while cnt < 10:
    sum = sum+cnt
    cnt = cnt+1
tp_log("sum = " + str(sum))
#expected result:
#sum = 45
```

for

1.15 for

- 기능

range로 반복 범위를 설정하여 반복작업을 실행합니다.

- 문법

```
for <item> in <sequence형 객체 S>:  
    <구문>
```

- 예제

```
x=0  
for i in range(0, 3): # i는 0 -> 1 -> 2  
    x= x + 1  
  
sum = 0  
for i in range(0, 10):  
    sum = sum + i  
tp_log("sum = " + str(sum))  
#expected result:  
#sum = 45
```

1.16 break

- 기능

반복문 내부 블록을 빠져나올 때 사용합니다.

- 예제

```
x =0  
while True:  
    x = x + 1  
    if x > 10:  
        break
```

```

sum = 0;cnt = 0
while True:
    if cnt > 9:
        break
    sum = sum + cnt
    cnt = cnt+1
    tp_log("sum = " + str(sum))
#expected print result:
#sum = 45

```

1.17 continue

- 기능

반복문 내부 블록을 더 이상 수행하지 않고 반복문 시작 지점으로 이동할 때 사용합니다.

- 예제

```

#<ex> 1
x=0
y=0
while True:
    x = x + 1
    if x > 10:
        continue
    y += 100

#<ex> 2
sum = 0
for i in range(0, 10):
    if i%2==0:
        continue
    sum = sum + i
tp_log("sum of odd numbers = " + str(sum))
#sum of odd numbers = 25

```

1.18 반복문의 else

- **기능**

반복문 수행 도중 break 함수로 인하여 반복문이 종료되지 않고 끝까지 수행되었을 때 else 블록이 실행됩니다.

- **예제**

```
L = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 }

for i in L:
    if i % 2 == 0:
        continue
    else:
        tp_log("exit without break")
```

2. 모션 관련 명령어

2.1 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)

```

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

- 관련 명령어

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

2.2 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)
movej(x2, v=100, a=200)
```

▪ 관련 명령어

movej() / movec() / movejx() / amovej() / amovec() / amovejx()

2.3 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	-	posx 또는 position list
	list (float[6])		
delta	posx	-	posx 또는 position list
	list (float[6])		
ref	int	None	reference coordinate <ul style="list-style-type: none"> DR_BASE : base coordinate DR_WORLD : world coordinate DR_TOOL : tool coordinate user coordinate: 사용자 정의
ref_out	int	DR_BASE	reference coordinate <ul style="list-style-type: none"> DR_BASE : base coordinate DR_WORLD : world coordinate user coordinate: 사용자 정의

▪ 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

예외	설명
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)
move(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)
move(x9, v=100, a=100, ref=DR_BASE)

```

■ 관련 명령어

posx()/addto()

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

2.4 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 입력 합니다. radius는 이어지는 segment와의 blending 반경을 설정합니다.

▪ 인수

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

▪ 리턴

posbt

▪ 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()

2.5 fkin(pos, ref)

▪ 기능

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

▪ 인수

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

▪ 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

```

▪ 관련 명령어

set_tcp() #티칭펜던트(TP)에 등록한 이름의 tcp 정보가 fkin연산 시 반영

posj()/**posx()**

2.6 ikin(pos, sol_space, ref)

▪ 기능

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

▪ 인수

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

▪ 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)
```

■ 관련 명령어

set_tcp()/posj()/posx()

addto(pos, add_val=None)

2.7 addto(pos, add_val=None)

▪ 기능

posj의 각 관절 값에 add_val 만큼 추가하여 새로운 posj 객체를 생성합니다.

▪ 인수

인수명	자료형	기본값	설명
pos	posj	-	posj 또는 position list
	list (float[6])		
add_val	list (float[6])	None	Position에 추가할 add 값 목록 *None 또는 [] 일 시에는, 값을 추가 안 함

▪ 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

▪ 예제

```
q1 = posj(10, 20, 30, 40, 50, 60)
movej (q1, v=10, a=20)
q2 = addto(q1, [0, 0, 0, 0, 45, 0])
movej (q2, v=10, a=20) # 로봇이 관절(10, 20, 30, 40, 95, 60)로 이동합니다.
q3 = addto(q2, [])
q4 = addto(q3)
```

▪ 관련 명령어

posj()

2.8 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	-	velocity(모든 축에 동일) 또는
	list (float[6])		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
#2
20(deg/sec)입니다.
set_velj(20.5) # 소수점 입력 가능합니다.
set_velj([10, 10, 20, 20, 30, 10]) # 축별 전역 속도 지정 가능합니다.
```

set_velj(vel)

- 관련 명령어

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

2.9 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 인자 를 명시적으로 입력하지 않는 경우에 default 가속도로 적용됩니다.

▪ 인수

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

2.10 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 설정하십시오.
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.
```

set_velx(vel1, vel2)

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

- **관련 명령어**

**set_accx()/moveL()/moveC()/moveSX()/moveB()/moveSpiral()/amoveL()/amoveC()/
amoveSX()/amoveB()/amoveSpiral()**

2.11 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 각속도를 60(mm/sec2)로 설정하십시오. 전역 태스크 각각속도는 자동결정됨
movel(P2) # P2로의 태스크 모션 선속도는 전역 속도인 30(mm/sec)
movel(P1, vel=20, acc=40) # P1으로의 태스크 모션 선속도는 지정 속도인
```

set_velx(vel)

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

- **관련 명령어**

**set_accx()/moveL()/moveC()/moveSX()/moveB()/moveSpiral()/amoveL()/amoveC()/
amoveSX()/amoveB()/amoveSpiral()**

2.12 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.
```

set_accx(acc1, acc2)

- 관련 명령어

set_velx()/moveL()/moveC()movesX()/moveB()/move_spiral()/amoveL()/amoveC()/amoveX()/amoveB()/amove_spiral()

2.13 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 정의 하며 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)
movej(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 설정하십시오. 전역 태스크 각각 가속도는 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()

2.14 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()

set_ref_coord(coord)

2.15 **set_ref_coord(coord)**

▪ 기능

기준 좌표계를 설정합니다.

▪ 인수

인수명	자료형	기본값	설명
coord	int	-	<p>기준 좌표계</p> <ul style="list-style-type: none">• DR_BASE: Base Coordinate• DR_WORLD: World Coordinate• DR_TOOL: Tool Coordinate• user Coordinate: 사용자 정의

▪ 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)
x1 = posx(370.9, 419.7, 651.5, 90,-180,0)
movel(x1, v=100, a=100) # Base Coordinate 기준
uu1 = [-1, 1, 0] # 사용자좌표계의 x축방향벡터(Base Coordinate 기준)
vv1 = [1, 1, 0] # 사용자좌표계의 y축방향벡터(Base Coordinate 기준)
pos = posx(370.9, -419.7, 651.5, 0, 0, 0) # 사용자좌표계의 원점좌표
DR_USER1 = set_user_cart_coord(uu1, vv1, pos) # 사용자좌표계 설정
set_ref_coord(DR_USER1) # 사용자좌표계 DR_USER1를 전역좌표계로 지정
movel([0,0,0,0,0,0],v=100,a=100) # 별도의 ref지정 없는 경우 전역좌표를 따름
# DR_USER1좌표계의 원점 및 방향으로 이동
movel([0,200,0,0,0,0],v=100,a=100) # DR_USER1좌표계의 (0,200,0) 지점으로 이동
```

- 관련 명령어

move()/**movej()**/**movec()**/**movesx()**/**moveb()**/**move_spiral()**/**move_periodic()**

2.16 movej

▪ 기능

로봇이 현재 관절위치에서 목표 관절위치(pos)로 이동합니다.

▪ 인수

인수명	자료형	기본값	설명
pos	posj	-	posj 또는 joint angle list
	list (float[6])		
vel (v)	float	None	velocity(모든 축에 동일) 또는 velocity(축별 velocity)
	list (float[6])		
acc (a)	float	None	acceleration(모든 축에 동일) 또는 acceleration(축별 acceleration)
	list (float[6])		
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

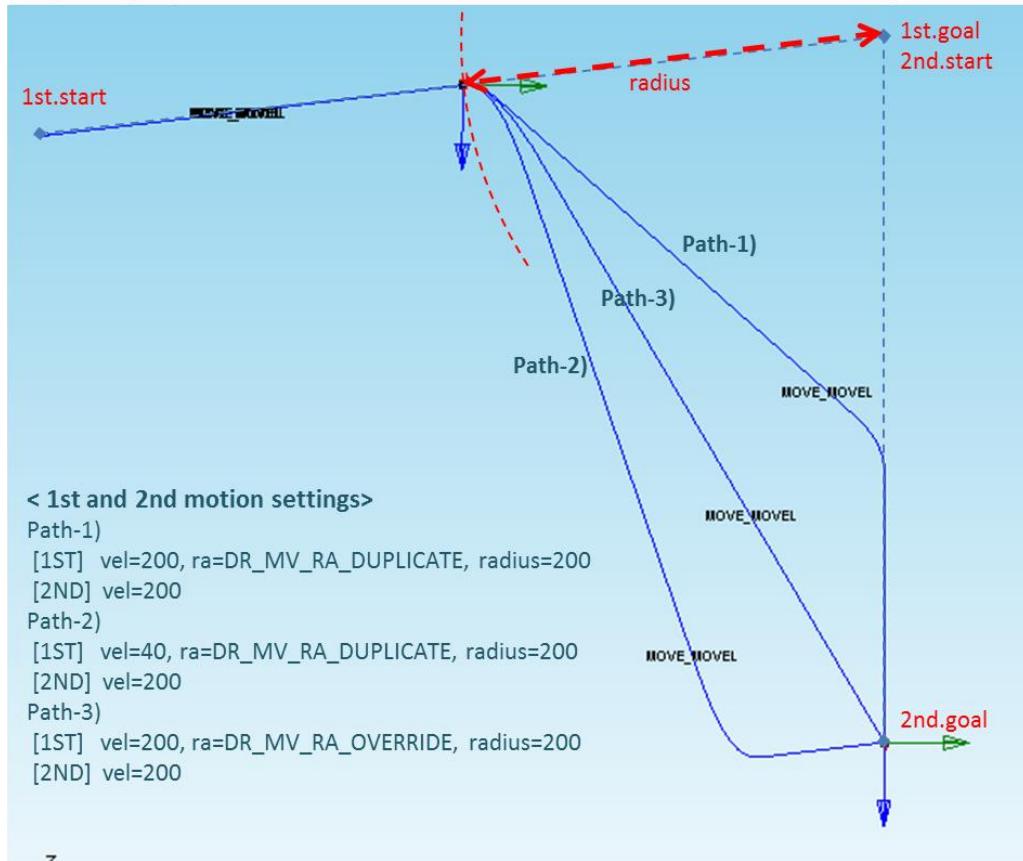
알아두기

- 단축 인수를 지원합니다. (v:vel, a:acc, t:time, r:radius)
- vel 이 None 인 경우, _global_velj 이 적용됩니다. (_global_velj 초기값은 0.0이며, set_velj에 의해 설정 가능)
- acc 이 None 인 경우, _global_accj 이 적용됩니다. (_global_accj 초기값은 0.0이며, set_accj에 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 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

```

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/sec2)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()

2.17 movel

▪ 기능

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

▪ 인수

인수명	자료형	기본값	설명
pos	posx	-	posx 또는 position list
	list (float[6])		
vel (v)	float	None	velocity 또는 velocity1, velocity2
	list (float[2])		
acc (a)	float	None	acceleration 또는 acceleration1, acceleration2
	list (float[2])		
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_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
ra	int	DR_MV_RA_DUPLICATE	Reactive motion mode • DR_MV_RA_DUPLICATE: duplicate • DR_MV_RA_OVERRIDE: override

알아두기

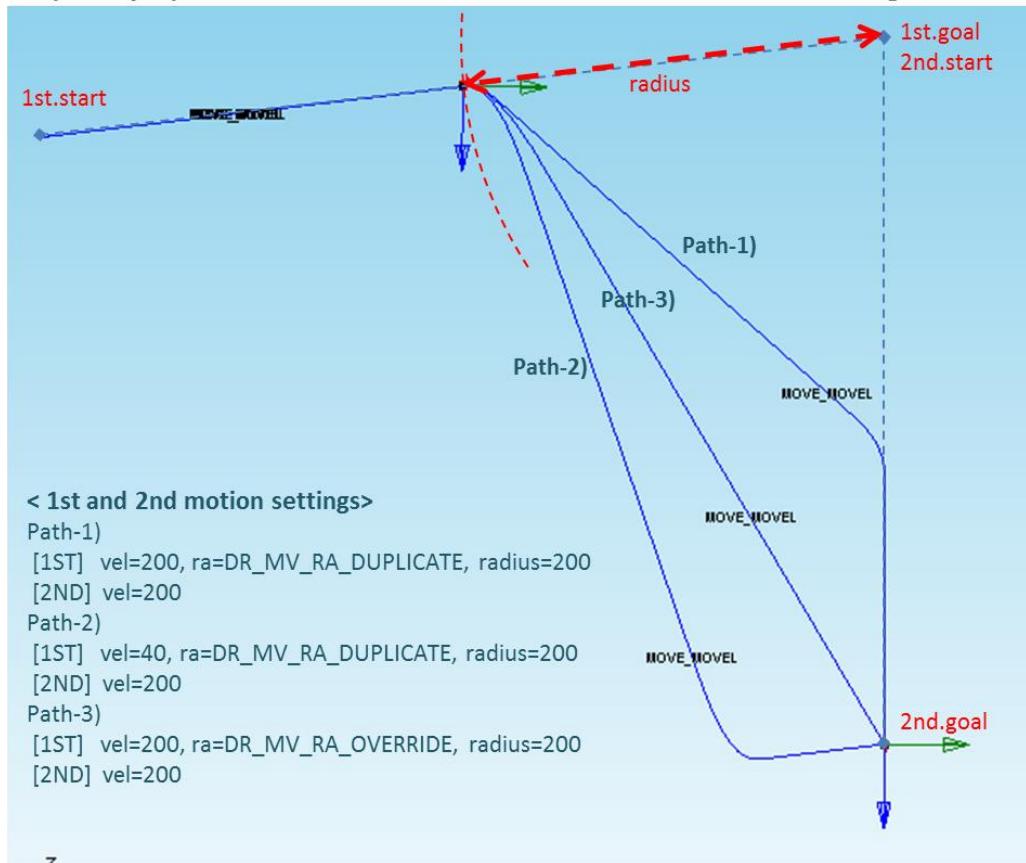
- 단축 인수를 지원합니다. (v:vel, a:acc, t:time, r:radius)
- vel 가 None 인 경우, _global_velx 0이 적용됩니다. (_global_velx 초기값은 0.0이며, set_velx에 의해 설정 가능)

- acc 가 None 인 경우, _global_accx 이 적용됩니다. (_global_accx 초기값은 0.0이며, 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 초기값은 DR_BASE 이며, set_ref_coord 명령에 의해 설정 가능)

⚠ 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

예외	설명
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_vlx\(\)](#)/[set_accx\(\)](#)/[set_tcp\(\)](#)/[set_ref_coord\(\)](#)/[amovel\(\)](#)

2.18 movejx

▪ 기능

로봇이 관절 공간 안에서 목표 위치(pos)로 이동합니다.

목표 위치는 작업공간 상의 posx형으로 입력하므로 movej과 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	-	posx 또는 position list
	list (float[6])		
vel (v)	float	None	velocity(모든 축에 동일) 또는 velocity(축별 velocity)
	list (float[6])		
acc (a)	float	None	acceleration(모든 축에 동일) 또는 acceleration(축별 acceleration)
	list (float[6])		
time (t)	float	None	도달 시간 [sec]
radius (r)	float	None	blending시 radius
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
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 초기값은 0.0이며, set_velj 의해 설정 가능)
- acc 가 None 인 경우, _global_accj 가 적용됩니다. (_global_accj 초기값은 0.0이며, set_accj 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 초기값은 DR_BASE이며, set_ref_coord 명령에 의해 설정 가능)
- 상대모션으로 입력하는 경우(mod=DR_MV_MOD_REL), 선행모션에 블렌딩을 사용하는 경우 예리가 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)
movej(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()/movejx()`

2.19 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	-	posx 또는 position list
	list (float[6])		
pos2	posx		posx 또는 position list
	list (float[6])		
vel (v)	float	None	velocity 또는 velocity1, velocity2
	list (float[2])		
acc (a)	float	None	acceleration 또는 acceleration1, acceleration2
	list (float[2])		
time (t)	float	None	도달 시간 [sec]
radius (r)	float	None	blending시 radius
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
angle (an)	float	None	angle 또는 angle1, angle2
	list (float[2])		
ra	int	DR_MV_RA_DUPLICATE	Reactive motion mode • DR_MV_RA_DUPLICATE: duplicate • DR_MV_RA_OVERRIDE: override



알아두기

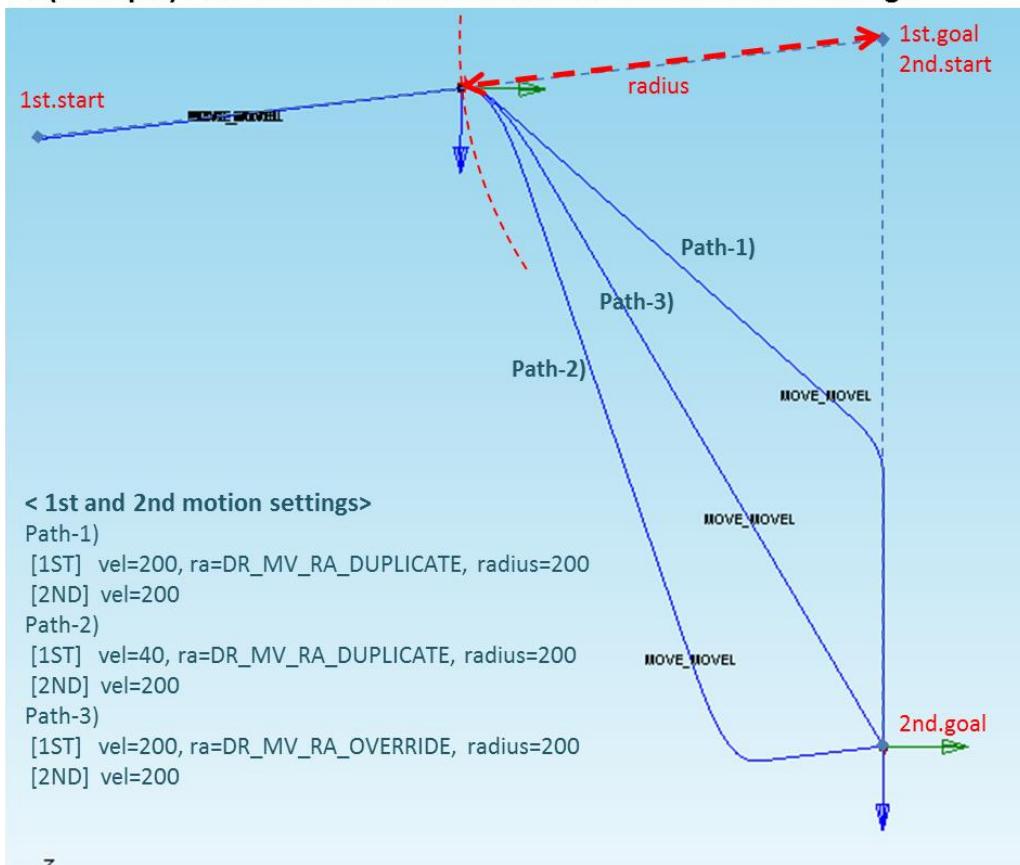
- 단축 인수를 지원합니다. (v:vel, a:acc, t:time, r:radius, angle:an)
- vel 이 None 인 경우 _global_velx 가 적용됩니다. (_global_velx 초기값은 0.0이며, set_velx 의해 설정 가능)
- acc 가 None 인 경우 _global_accx 가 적용됩니다. (_global_accx 초기값은 0.0이며, set_accx 의해 설정 가능)
- vel 에 하나의 인자를 입력한 경우(예를들어, vel=30) 입력된 인자는 모션의 선속도에 대응되며, 각속도는 선속도에 비례하여 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 초기값은 DR_BASE이며, set_ref_coord 명령에 의해 설정 가능)
- 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 참고하십시오.

< (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()`

2.20 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	None	velocity(모든 축에 동일) 또는 velocity(축별 velocity)
	list (float[6])		
acc (a)	float	None	acceleration(모든 축에 동일) 또는 acceleration(축별 acceleration)
	list (float[6])		
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 초기값은 0.0이며, set_velj에 의해 설정 가능)
- acc 이 None 인 경우 _global_accj 가 적용됩니다. (_global_accj 초기값은 0.0이며, set_accj에 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()

2.21 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	None	velocity 또는 velocity1, velocity2
	list (float[2])		
acc (a)	float	None	acceleration 또는 acceleration1, acceleration2
	list (float[2])		
time (t)	float	None	도달 시간 [sec]
ref	int	None	reference coordinate <ul style="list-style-type: none"> • DR_BASE: base coordinate • DR_WORLD: world coordinate • DR_TOOL: tool coordinate • user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 <ul style="list-style-type: none"> • DR_MV_MOD_ABS: 절대 • DR_MV_MOD_REL: 상대
vel_opt	int	DR_MVS_VEL_NONE	속도 옵션 <ul style="list-style-type: none"> • 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 초기값은 0.0이며, set_velx 의해 설정 가능)
- acc 이 None 인 경우 _global_accx 가 적용됩니다. (_global_accx 초기값은 0.0이며, set_accx 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_BASE이며, set_ref_coord 명령에 의해 설정 가능)
- 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()`

2.22 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	None	velocity 또는 velocity1, velocity2
	list (float[2])		
acc (a)	float	None	acceleration 또는 acceleration1, acceleration2
	list (float[2])		
time (t)	float	None	도달 시간 [sec] * time 지정 시, vel, acc를 무시하고 time 기준으로 처리
ref	int	None	reference coordinate <ul style="list-style-type: none"> • DR_BASE: base coordinate • DR_WORLD: world coordinate • DR_TOOL: tool coordinate • user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 <ul style="list-style-type: none"> • 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 초기값은 0.0이며, set_velx 의해 설정 가능)
- acc 이 None 인 경우 _global_accx 가 적용됩니다.(_global_accx 초기값은 0.0이며, set_accx 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_BASE이며, set_ref_coord 명령에 의해 설정 가능)
- 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
DR_Error (DR_ERROR_RUNTIME)	C Extension 모듈 에러 발생 시

예외	설명
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()

2.23 move_spiral

▪ 기능

지정한 좌표계(ref) 상에서 현재위치를 중심으로 지정축(axis)에 수직한 평면 위의 나선궤적을 따르는 모션을 수행한다. 이동거리(lmax)의 추가 입력시 원뿔의 꼭지점으로 부터 시작하여 주위를 따라 도는 궤적을 움직이게 할 수 있다.

▪ 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_WORLD : world 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 0.0이며, 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 초기값은 0.0이며, 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.
- 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

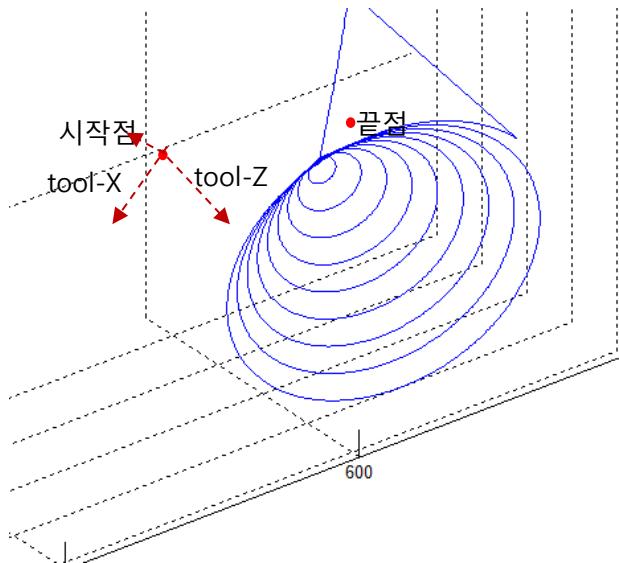
■ 예외

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
#(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()

2.24 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 period에 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_WORLD : world coordinate • DR_TOOL : tool coordinate • user coordinate : 사용자 정의

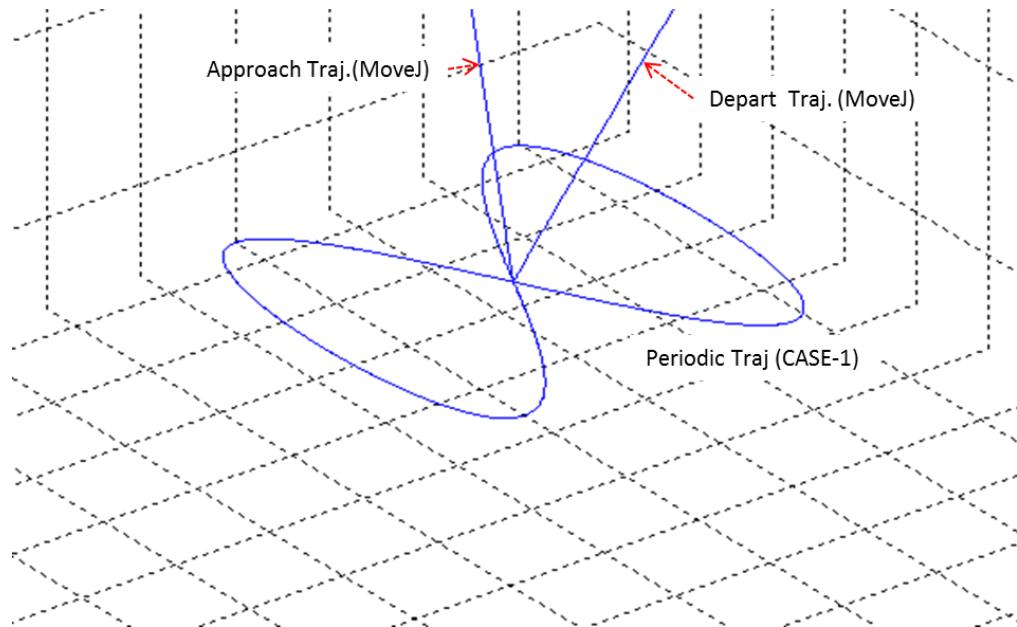
☞ 알아두기

- amp는 진폭(amplitude)을 의미하며, 각 축(x, y, z, rx, ry, rz) 별로 amp를 값으로 하는 6 개 원소의 list 형태로 입력해야 합니다. 단, 주기 모션을 진행하지 않는 축 방향은 amp를 0으로 입력해야 합니다.
- period는 해당 방향 모션의 1회 반복 시간을 의미하며, 각 축(x, y, z, rx, ry, rz) 별 period를 값으로 하는 총 6 개 원소의 list 형태로 입력하거나 대표값을 입력해야합니다.
- atime은 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.
- repeat은 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.

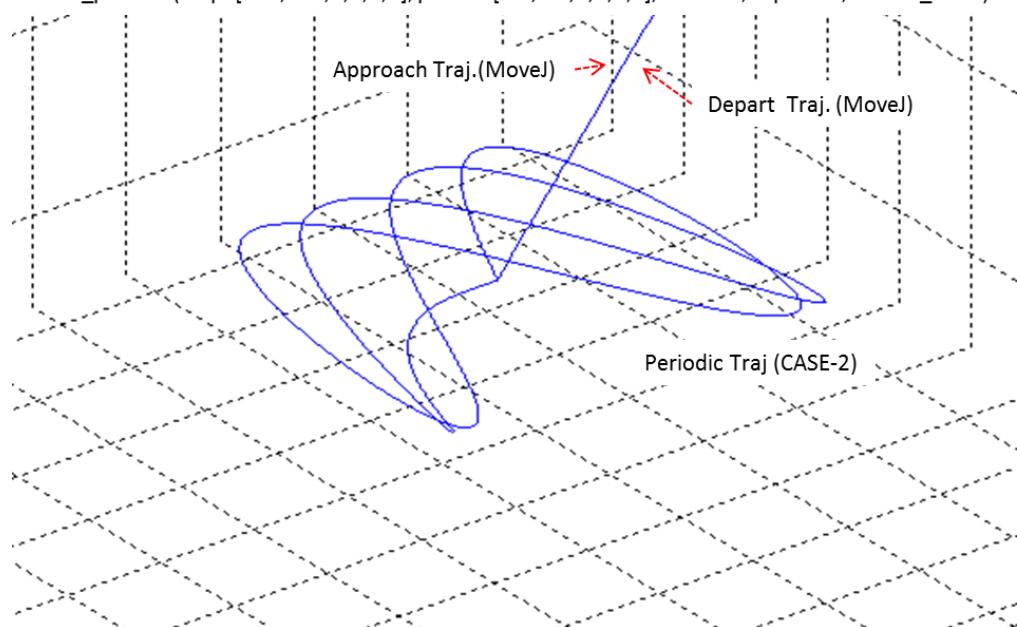
CASE-1) All-axis motions end at the same time

```
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.
- 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) 모션  
이  
# 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 1  
초)  
# 이며, x축은 4.5회 반복 수행
```

- 관련 명령어

set_ref_coord()/amove_periodic()

2.25 move_home

▪ 기능

기계적 홈 또는 사용자정의 홈 위치로 조인트모션으로 이동하여 호밍을 수행합니다. 입력인자 [target]에 따라 시스템에 정의된 기계적 홈 또는 사용자설정 홈으로 이동합니다.

▪ 인수

인수명	자료형	기본값	범위	설명
target	int	-		목표하는 홈자세 . DR_HOME_TARGET_MECHANIC : 기계적 홈, 조인트각도 (0,0,0,0,0,0) . DR_HOME_TARGET_USER : 사용자설정 홈위치

알아두기

- 호밍동작은 크게 두 동작으로 나뉘어 순차적으로 수행됩니다.
 - 1) 호밍위치로 시스템에 지정된속도로 이동
 - 2) 정밀하게 홈위치를 찾는 동작
- 호밍동작 중 주변에 충돌위험이 있지않도록 안전을 확보하여야 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

```
move_home(DR_HOME_TARGET_USER) # 사용자가 설정한 홈 위치로 이동
```

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

2.26 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	-	posj 또는 joint angle list
	list (float[6])		
vel (v)	float	None	velocity(모든 축에 동일) 또는 velocity(축별 velocity)
	list (float[6])		
acc (a)	float	None	acceleration(모든 축에 동일) 또는 acceleration(축별 acceleration)
	list (float[6])		
time (t)	float	None	도달 시간 [sec]
mod	int	DR_MV_MOD_ABS	이동 기준 <ul style="list-style-type: none"> • DR_MV_MOD_ABS: 절대 • DR_MV_MOD_REL: 상대
ra	int	DR_MV_RA_DUPLICATE	Reactive motion mode <ul style="list-style-type: none"> • 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 초기값은 0.0이며, set_velj 의해 설정 가능)
- acc 이 None 인 경우, _global_accj 가 적용됩니다. (_global_accj 초기값은 0.0이며, set_accj 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()

2.27 amovel

▪ 기능

비동기(async.)방식의 movel모션으로 블렌딩을 위한 radius인자를 갖지 않는 점을 제외하고 movel과 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	-	posx 또는 position list
	list (float[6])		
vel (v)	float	None	velocity 또는 velocity1, velocity2
	list (float[2])		
acc (a)	float	None	acceleration 또는 acceleration1, acceleration2
	list (float[2])		
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_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
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 초기값은 0.0이며, set_velx 에 의해 설정 가능)
- acc 가 None 인 경우, _global_accx 적용(_global_accx 초기값은 0.0이며, 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)
- 옵션 ra 및 vel/acc 에 따른 blending 시의 경로는 move() 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()move()`

2.28 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	-	posx 또는 position list
	list (float[6])		
vel (v)	float	None	velocity(모든 축에 동일) 또는 velocity(축별 velocity)
	list (float[6])		
acc (a)	float	None	acceleration(모든 축에 동일) 또는 acceleration(축별 acceleration)
	list (float[6])		
time (t)	float	None	도달 시간 [sec]
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
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 초기값은 0.0이며, set_velj 의해 설정 가능합니다.)
- acc 가 None 인 경우 _global_accj 가 적용됩니다. (_global_accj 초기값은 0.0이며, set_accj 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_BASE이며, set_ref_coord 명령에 의해 설정이 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()

2.29 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	-	posx 또는 position list
	list (float[6])		
pos2	posx	-	posx 또는 position list
	list (float[6])		
vel (v)	float	None	velocity 또는 velocity1, velocity2
	list (float[2])		
acc (a)	float	None	acceleration 또는 acceleration1, acceleration2
	list (float[2])		
time (t)	float	None	도달 시간 [sec]
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
angle (an)	float	None	angle 또는 angle1, angle2
	list (float[2])		

인수명	자료형	기본값	설명
ra	int	DR_MV_RA_DUPLICATE	<p>Reactive motion mode</p> <ul style="list-style-type: none"> • 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 초기값은 0.0이며, set_velx 의해 설정 가능)
- acc 가 None 인 경우 _global_accx 가 적용됩니다. (_global_accx 초기값은 0.0이며, set_accx 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_BASE이며, set_ref_coord 명령에 의해 설정 가능)
- 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()`

2.30 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 수행 합니다.
- 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	None	velocity(모든 축에 동일) 또는 velocity(축별 velocity)
	list (float[6])		
acc (a)	float	None	acceleration(모든 축에 동일) 또는 acceleration(축별 acceleration)
	list (float[6])		
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 초기값은 0.0이며, set_velj 의해 설정 가능)
- acc 이 None 인 경우 _global_accj 가 적용됩니다. (_global_accj 초기값은 0.0이며, set_accj 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. ($pos_list=[q_1, q_2, \dots, q_{(n-1)}, q_n]$)로 이루어질 때 q_1 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()

2.31 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	None	velocity 또는 velocity1, velocity2
	list (float[2])		
acc (a)	float	None	acceleration 또는 acceleration1, acceleration2
	list (float[2])		
time (t)	float	None	도달 시간 [sec]
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
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 초기값은 0.0이며, set_velx 의해 설정 가능)
- acc 이 None 인 경우 _global_accx 가 적용됩니다. (_global_accx 초기값은 0.0이며, set_accx 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_BASE이며, set_ref_coord 명령에 의해 설정 가능)
- 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()    # 모션이 종료할 때까지 프로그램 일시중지
```

■ 관련 명령어

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

2.32 amoveb

▪ 기능

비동기(async.)방식의 moveb모션으로 async 처리 외에는 moveb()와 동일하게 동작하며 명령어를 수행한 후 바로 다음 라인을 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	None	velocity 또는 velocity1, velocity2
	list (float[2])		
acc (a)	float	None	acceleration 또는 acceleration1, acceleration2
	list (float[2])		
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_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



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 초기값은 0.0이며, set_velx에 의해 설정 가능)
- acc 이 None 인 경우, _global_accx 가 적용됩니다. (_global_accx 초기값은 0.0이며, set_accx에 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_BASE이며, set_ref_coord 명령에 의해 설정 가능)
- 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 도달하
# 면
# 다음 segment로 blending이 시작됨
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

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

2.33 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.

지정한 좌표계(ref) 상에서 현재위치를 중심으로 지정축(axis)에 수직한 평면 위의 나선궤적을 따르는 모션을 수행한다. 이동거리(lmax)의 추가 입력시 원뿔의 꼭지점으로부터 시작하여 주위를 따라 도는 궤적을 움직이게 할 수 있다

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_WORLD : world coordinate • DR_TOOL : tool coordinate

인수명	자료형	기본값	범위	설명
				• 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 0.0이며, 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 초기값은 0.0이며, 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.
- 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

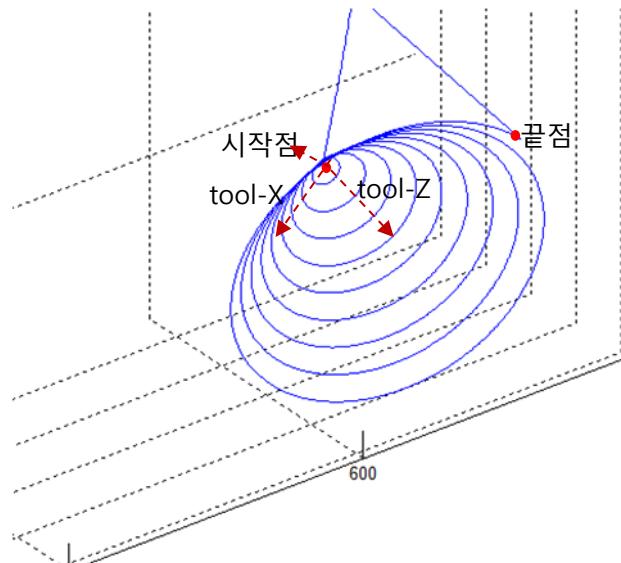
■ 예제

```

## hole search
# (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()

2.34 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

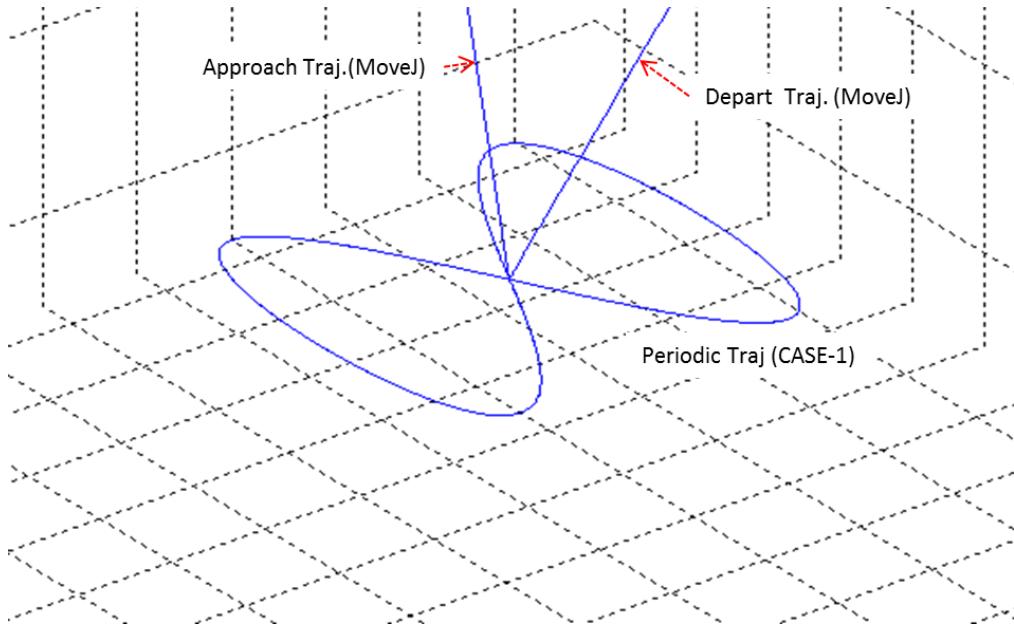
인수명	자료형	기본값	범위	설명
amp	list (float[6])	-	0≤amp	Amplitude(-amp에서 +amp사이 모션) [mm] or [deg]
period	float or list (float[6])		0≤period	period(1주기 소요 시간)[sec]
atime	float	0.0	0≤atime	Acc-, dec- time [sec]
repeat	int	1	> 0	반복 횟수
ref	int	DR_TOOL	-	reference coordinate • DR_BASE : base coordinate • DR_WORLD : world 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 1회 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 동시에 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.
- 모션명령 수행 시 최대속도 에러가 발생하는 경우 다음의 식을 참조하여 진폭 및 주기를 조정할 것을 제안합니다.

$$\text{최대속도} = \text{진폭(amp)} * 2 * \pi(3.14) / \text{주기(period)}$$

(예, 진폭=10mm, 주기=1 초인 경우 최대속도=62.83mm/sec)
- 선행모션과 후행모션에 대한 온라인 블렌딩은 지원하지 않습니다.

amove_periodic

▪ 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()

2.35 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()**

2.36 begin_blend(radius=0)

▪ 기능

블렌딩 구간을 시작합니다. 이후의 블렌딩 구간 인자 radius를 갖는 sync motion 명령어(movej, movel, movec, movejx)들은 기본적으로 인자로 설정된 radius를 이용해 blending됩니다. radius가 0인 경우 실질적인 blending 효과가 없습니다. 또한 설정된 radius와 다른 blending radius의 설정이 필요한 경우는 해당 모션의 인자에 blending radius를 지정함으로써 예외적으로 blending radius의 변경이 가능합니다.

▪ 인수

인수명	자료형	기본값	설명
radius	float	0	Blending 시 radius

▪ 리턴

값	설명
0	성공

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
#1
begin_blend(radius=30)
    # 이후의 radius 옵션을 가지는 모션 명령은 blending 구간을 30mm로
    # 일괄설정함
    Q1 = posj(0,0,90,0,90,0)
    Q2 = posj(0,0,0,0,90,0)
    movej(Q1, vel=10, acc=20)
        # Q1 관절각으로 이동하며 Q1의 공간 위치로부터 30mm 전역 거리가 될 때
        # 다음 모션을 수행하도록 설정됨
```

begin_blend(radius=0)

```
movej(Q2, time=5)
    # 직전모션을 유지하며(모션중첩) Blending하여 Q2 관절각으로 이동하며,
    # Q2의 공간 위치로부터 30mm 전역 거리가 될 때 다음 모션을 수행하도록
    # 설정됨

movej(Q1, v=30, a=60, r=200)
    # 직전모션을 유지하며(모션중첩) Blending하여 Q1 관절각으로 이동하며,
    # Q1의 공간 위치로부터 200mm 거리가 될 때 다음 모션을 수행하도록
    # 설정(전역 설정 미적용)

movej(Q2, v=30, a=60, ra= DR_MV_RA_OVERRIDE)
    # 직전모션을 즉시 종료하며 Blending하여 Q2 관절각으로 이동

end_blend()  # blending 구간 일괄설정 해제
```

- **관련 명령어**

end_blend()/movej()/movel()/movejx()/movec()

2.37 end_blend()

▪ 기능

블렌딩 구간을 종료합니다. begin_blend()로 시작된 블렌딩 구간의 효력이 정지됨을 의미합니다.

▪ 인수

해당 사항 없음

▪ 리턴

값	설명
0	성공

▪ 예외

해당 사항 없음

▪ 예제

```
#1
begin_blend(radius=30)
    # 이후의 radius 옵션을 가지는 모션명령은 blending구간을 30mm로 일괄설정
Q1 = posj(0,0,90,0,90,0)
Q2 = posj(0,0,0,0,90,0)
movej(Q1, vel=10, acc=20)
    # Q1관절각으로 이동시키며 Q1의 공간위치로부터 30mm의 전역거리가 될 때
    # 다음 모션을 수행하도록 설정
movej(Q2, time=5)
    # 직전모션을 유지하며(모션중첩) Blending하여 Q2관절각으로 이동,
    # Q2의 공간위치로부터 30mm의 전역거리가 될 때 다음 모션을
    # 수행하도록 설정
movej(Q1, v=30, a=60, r=200)
    # 직전모션을 유지하며(모션중첩) Blending하여 Q1관절각으로 이동시키며
    # Q1의 공간위치로부터 200mm의 거리가 될 때 다음 모션을 수행하도록
    # 설정 (전역설정 미적용)
movej(Q2, v=30, a=60, ra= DR_MV_RA_OVERRIDE)
```

end_blend()

```
# 직전모션을 즉시 종료시키며 Blending하여 Q2관절각으로 이동  
end_blend() # blending구간 일괄설정 해제
```

- 관련 명령어

begin_blend()/movej()/movel()/movejx()/movec()

2.38 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()/amov
eb()/amove_spiral()/amove_periodic()

2.39 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()/moveel()/movejx()/movec()/movesj()/movesx()/moveb()/move_spiral()
/move_periodic()/amovej()/amovel()/amovejx()/amovec()/amovesj()/amovesx()/amov
eb()/amove_spiral()/amove_periodic()

2.40 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

change_operation_speed(speed)

▪ 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 이
                               동
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()/a
move_spiral()/amove_periodic()**

2.41 wait_manual_guide()

▪ 기능

프로그램 수행 중 핸드가이딩(콕핏 또는 TP의 직접교시 버튼을 누르며 로봇의 위치변경)을 허용하며 사용자는 핸드가이딩을 완료한 후 아래의 두 가지 방법 중 하나를 수행하면 다음 명령어로 진행합니다(프로그램이 종료되지 않는 한, 사용자가 자유롭게 핸드가이딩을 수행한 후 아래의 방법 중 하나를 실행할 때까지 본 명령어에서 대기합니다).

- 1) TP에 발생한 ‘핸드가이딩 수행’ 팝업창에서 ‘확인’ 또는 ‘완료’ 버튼을 누릅니다.
- 2) Safety I/O 설정을 통해 ‘manual guide 해제’로 지정한 Digital Input 채널에 신호를 인가합니다.

본 명령어를 정상적으로 실행하기 위해서는 현재의 TCP위치 및 핸드가이딩 중의 로봇의 TCP 위치가 협동작업영역(collaborative workspace)내에 위치하여야 하므로 사전에 핸드가이딩을 수행할 영역을 협동작업영역으로 설정하고 활성화한 후에 실행하시기 바랍니다. 작업자의 안전을 위하여 현재의 위치 또는 핸드가이딩 중 협동작업영역을 벗어나는 경우에는 에러가 발생하며 작업 프로그램이 정지됩니다.

▪ 인수

해당 사항 없음

▪ 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

```
# 프로그램 실행 전 협동 작업 영역 설정  
# 면1: 점-1(1000,1000), 점-2(0,0)  
# 면2: 점-1(1000,-1000), 점-2(0,0)  
# 영역1 활성화 - 점(1000,0)  
  
j00 = posj(0,0,90,0,90,0)  
movej(j00,vel=20,acc=40) # 협동 작업 영역으로 진입  
wait_manual_guide() # TP에 발생된 팝업의 '완료'버튼을 누를때까지 직접교시 가능  
pos1 = get_current_posx() # 직접교시된 점을 pos1에 저장합니다.  
dposa = posx(0,0,-100,0,0,0)  
movej(dposa, vel=300, acc=600, ref=DR_TOOL)  
# 교시된 위치로 부터 Tool-Z방향으로 100mm 후퇴합니다.
```

▪ 관련 명령어

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

2.42 wait_nudge()

▪ 기능

프로그램 수행 중 일시정지상태에서 사용자의 넛지입력(로봇에 외력을 가하는 행동)을 통한 작업재개를 허용합니다. 로봇이 정지한 상태에서 설정으로 입력한 외력하한치(threshold force) 이상의 외력이 인가되면 설정된 재개시간 후에 다음 명령어로 진행되며, 이는 프로그램 진행 중의 인터락(interlock)으로 활용될 수 있습니다.

다만, 로봇의 관절형태가 특이점영역에 있는 경우, 또는 넛지입력 후 지속적으로 힘이 인가되는 경우 안전을 위해 에러가 발생하는 것을 유의하십시오.

본 명령어를 정상적으로 실행하기 위해서는 현재의 TCP위치가 협동작업영역(collaborative workspace)내에 위치하여야 하므로 사전에 협동작업영역을 활성화하며 명령어를 수행할 로봇의 위치가 협동작업영역내에 있도록 설정하시기 바랍니다. 넛지입력을 인지할 외력의 크기 및 넛지입력 후 프로그램 재개시간의 설정 역시 협동작업영역 설정 창에서 입력할 수 있습니다.

▪ 인수

해당 사항 없음

▪ 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

```
# 프로그램 실행 전 협동 작업 영역 설정  
# 면1: 점-1(1000,1000), 점-2(0,0)  
# 면2: 점-1(1000,-1000), 점-2(0,0)  
# 영역1 활성화 - 점(1000,0)  
  
j00 = posj(0,0,90,0,90,0)  
movej(j00,vel=20,acc=40) # 협동 작업 영역으로 진입  
wait_nudge()           # 로봇에 nudge_threshold 이상의 외력을 가할 때까지 대기  
dposa = posx(0,0,-100,0,0,0)  
movel(dposa, vel=300, acc=600, ref=DR_TOOL)  
# 현재 위치로 부터 Tool-Z방향으로 100mm 후퇴합니다.
```

■ 관련 명령어

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

2.43 enable.Alter_motion(n,mode,ref,limit_dPOS,limit_dPO S_per)

▪ 기능

경로 수정 기능을 활성화 합니다. 경로 생성의 단위 주기는 100msec이며 입력 인자 n 을 설정하여 경로 생성 주기($n * 100\text{msec}$)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 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	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.

enable.Alter_motion(n, mode, ref, limit_dPOS, limit_dPOS_per)

- 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.

■ 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)

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

```
thread_stop(th_id)
disable.Alter_motion() # 경로 수정 기능 비활성화
```

- 관련 명령어

alter_motion(pos), disable.Alter_motion()

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

▪ 기능

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_motion의 설정 값 limit_dPOS, limit_dPOS_per에 따라 경로 수정량은 조정될 수 있습니다.
- pos의 방향값은 Fixed XYZ로 설정하여야 됩니다.

▪ 인수

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

▪ 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()

2.45 **disable.Alter_motion()**

▪ 기능

경로 수정 기능을 비활성화 합니다.

▪ 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)

movev(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)

3. 제어 보조 명령어

3.1 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

3.2 get_control_space()

▪ 기능

현재 제어 공간을 리턴합니다.

▪ 인수

해당 사항 없음

▪ 리턴

값	설명
	제어모드
int	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

3.3 **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()
```

- **관련 명령어**

get_desired_posj()

3.4 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()

3.5 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\(\)](#)

3.6 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()

3.7 **get_current_posx(ref)**

▪ 기능

현재 태스크 좌표계의 자세와 solution space를 리턴합니다. 이때 자세는 ref coordinate 를 기준으로 합니다.

▪ 인수

인수명	자료형	기본값	설명
ref	Int	DR_BASE	reference coordinate <ul style="list-style-type: none"> • DR_BASE : base coordinate • DR_WORLD : world coordinate • user coordinate: 사용자 정의

알아두기

- ref 생략시 DR_BASE 로 적용됩니다.

▪ 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

x1_wld, sol = get_current_posx(ref=DR_WORLD)  #x1 w.r.t. DR_WORLD

DR_USR1=set_user_cart_coord(x1, x2, x3, pos)
set_ref_coord(DR_USR1)

x1, sol = get_current_posx(DR_USR1)  #x1 w.r.t. DR_USR1

```

- 관련 명령어

`get_desired_posx()`

3.8 **get_current_tool_flange_posx(ref)**

▪ 기능

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

▪ 인수

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

▪ 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

3.9 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

▪ 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\(\)](#)

3.10 **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 <ul style="list-style-type: none"> • DR_BASE : base coordinate • DR_WORLD : world coordinate • user coordinate: 사용자 정의



알아두기

- ref 생략시 DR_BASE로 적용됩니다.

▪ 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()`

3.11 `get_desired_velx(ref)`

▪ 기능

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

▪ 인수

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

▪ 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()`

3.12 **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()
```

- **관련 명령어**

get_solution_space()

3.13 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

▪ 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=[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]]
    
```

▪ 관련 명령어

해당 사항 없음

3.14 **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()

3.15 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\(\)](#)

3.16 **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

▪ 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(DR_WORLD) # force_ext: WORLD좌표계 기준 툴의 외력
```

▪ 관련 명령어

get_joint_torque()/get_external_torque()

3.17 get_solution_space(pos)

- **기능**

Solution space 값을 구합니다.

- **인수**

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

- **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()

get_orientation_error(xd, xc, axis)

3.18 get_orientation_error(xd, xc, axis)

▪ 기능

axis에 대한 임의의 pose xd와 xc 사이의 Orientation error 값을 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
xd	posx	-	posx 또는 position list
	list (float[6])		
xc	posx	-	posx 또는 position list
	list (float[6])		
axis	int	-	axis <ul style="list-style-type: none">• DR_AXIS_X: x축• DR_AXIS_Y: y축• DR_AXIS_Z: z축

▪ 리턴

값	설명
float	Orientation error 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

```
xd = posx(0, 0, 0, 0, 0, 0)
xc = posx(10, 20, 30, 40, 50, 60)
diff = get_orientation_error(xd, xc, DR_AXIS_X)
```

- 관련 명령어

`get_current_rotm()`

4. 기타 설정 및 안전 관련 명령어

4.1 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()

4.2 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()`

set_tool(name)

4.3 **set_tool(name)**

▪ 기능

티치팬던트에 등록된 툴 정보 중에서 입력된 name의 tool을 활성화 합니다.

▪ 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 등록된 "tool1" 의 정보를 활성화 한다.
```

▪ 관련 명령어

set_tcp()

4.4 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()`

4.5 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()**

5. 힘/강성 제어 및 기타 사용자 편의 기능

5.1 **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	-	posx 또는 position list
	list (float[6])		
x2	posx	-	posx 또는 position list
	list (float[6])		
x3	posx	-	posx 또는 position list
	list (float[6])		
axis	int	-	axis <ul style="list-style-type: none">• DR_AXIS_X: x축• DR_AXIS_Y: y축• DR_AXIS_Z: z축
ref	int	DR_BASE	reference coordinate <ul style="list-style-type: none">• DR_BASE: base coordinate• DR_WORLD: world coordinate• user coordinate: 사용자 정의

▪ 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)
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()`

5.2 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

▪ 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()`

5.3 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	-	posx 또는 position list
	list (float[6])		
x2	posx	-	posx 또는 position list
	list (float[6])		
x3	posx	-	posx 또는 position list
	list (float[6])		
pos	posx	-	posx 또는 position list
	list (float[6])		
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

▪ 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(0,0,45,0,90,0)
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()`

5.4 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	-	posx 또는 position list
	list (float[6])		
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

▪ 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(0,0,45,0,90,0)
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()`

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

5.5 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()

5.6 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\(\)](#)

5.7 task_compliance_ctrl(stx, time)

▪ 기능

기준에 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()/**release_compliance_ctrl()**

5.8 set_stiffnessx(stx, time)

▪ 기능

전역으로 설정된 좌표계(set_ref_coord() 참조) 기준으로 강성값을 설정합니다. 현재 강성 또는 기본 값으로부터 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, time)

```
set_stiffnessx(stx)      # 현재의 전역좌표계(World) 기준 강성 적용  
release_compliance_ctrl()
```

- 관련 명령어

task_compliance_ctrl()/release_compliance_ctrl()

5.9 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 z축 방향이 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 <ul style="list-style-type: none"> · DR_BASE: base coordinate · DR_WORLD: world coordinate
mod	int	-	입력 모드 (입력점개수가 2개인 경우에만 유효함) <ul style="list-style-type: none"> · 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

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

▪ 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_user21, ref=DR_WORLD)
```

▪ 관련 명령어

set_user_cart_coord()

5.10 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

▪ 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)
```

set_user_cart_coord(pos, ref)

```
user_id2 = set_user_cart_coord(pos2, ref=DR_WORLD)
```

- 관련 명령어

- set_ref_coord()**

5.11 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	-	posx 또는 position list
	list (float[6])		
x2	Posx	-	posx 또는 position list
	list (float[6])		
x3	Posx	-	posx 또는 position list
	list (float[6])		
pos	Posx	-	posx 또는 position list
	list (float[6])		
ref	int	DR_BASE	reference coordinate • DR_BASE: base coordinate • DR_WORLD: world coordinate

▪ 리턴

값	설명
양의 정수	Coordinate 설정 성공 설정된 Coordinate ID (101 ~ 200)
-1	Coordinate 설정 실패

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

▪ 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()`

5.12 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 $u1 \times v1$ 에 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

▪ 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

set_user_cart_coord(u1, v1, pos, ref)

예외	설명
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_tc2 = set_user_cart_coord(u1, v1, pos, ref=DR_WORLD)
```

■ 관련 명령어

[set_ref_coord\(\)](#)

5.13 overwrite_user_cart_coord(id, pos, ref)

▪ 기능

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

▪ 인수

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_cart_coord(pose_user1, ref=DR_BASE)
pose_user2 = posx(100, 150, 200, 45, 180, 0)
overwrite_user_cart_coord(id_user, pose_user2, ref=DR_BASE)

```

overwrite_user_cart_coord(id, pos, ref)

- 관련 명령어

set_user_cart_coord()

5.14 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_cart_coord(pose_user1, ref=DR_BASE)
pose, ref = get_user_cart_coord(id_user)
```

▪ 관련 명령어

`set_user_cart_coord()`

5.15 **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



- `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()`

5.16 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)
move(x3, v=10)
release_force(0.5)
release_compliance_ctrl()
```

- 관련 명령어

`set_desired_force()/task_compliance_ctrl()/set_stiffnessx()/release_compliance_ctrl()`

check_position_condition(axis, min, max, ref, mod, pos)

5.17 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.

입력좌표계(ref)가 DR_TOOL 인 경우 입력위치(pos)는 BASE좌표계 기준의 값을 입력하여야 합니다.

▪ 인수

인수명	자료형	기본값	설명
axis	int	-	Axis <ul style="list-style-type: none">• 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 <ul style="list-style-type: none">• DR_BASE : base coordinate• DR_WORLD : world coordinate• DR_TOOL : tool coordinate• user coordinate: 사용자 정의
mod	int	DR_MV_MOD_ABS	이동 기준 <ul style="list-style-type: none">• 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()`

check_position_condition(axis, min, max, ref, mod, pos)

5.18 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 축방향이고 모멘트 측정 시 axis는 툴 좌표계 기준의 축방향입니다.

▪ 인수

인수명	자료형	기본값	설명
axis	int	-	Axis <ul style="list-style-type: none"> 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 ($\text{min} \geq 0$)
max	float	DR_COND_NONE	최대값 ($\text{max} \geq 0$)
ref	int	None	reference coordinate <ul style="list-style-type: none"> 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

check_force_condition(axis, min, max, ref)

예외	설명
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)
#  $5 \leq f_z \leq 10$ 

while (fcon1):
    fcon2 = check_force_condition(DR_AXIS_C, min=30)           #  $30 \leq m_z$ 
    pcon1 = check_position_condition(DR_AXIS_X, min=0, max=0.1) #  $0 \leq x \leq 0.1$ 

    if (fcon2 and pcon1):
        break
```

▪ 관련 명령어

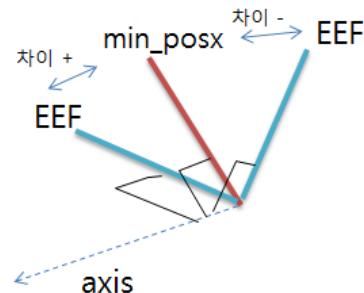
check_position_condition()/check_orientation_condition()/set_ref_coord()

5.19 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 확인 할 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	-	posx 또는 position list
	list (float[6])		
max	posx	-	posx 또는 position list
	list (float[6])		

check_orientation_condition(axis, min, max, ref, mod)

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

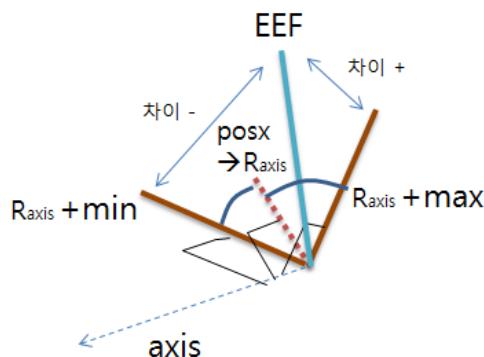
`check_position_condition()/check_force_condition()/check_orientation_condition()
/set_ref_coord()`

5.20 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 position에서 설정된 axis를 기준으로, 상대적인 각도 범위(min, max)를 말합니다. 인자 ref에 따라 주어진 position의 기준 좌표계가 정해집니다.

▪ 인수

인수명	자료형	기본값	설명
axis	int	-	<p>axis</p> <ul style="list-style-type: none">• 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.

```

check_orientation_condition(axis, min, max, ref, mod, pos)

```
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()

5.21 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

coord_transform(pose_in, ref_in, ref_out)

인수명	자료형	기본값	설명
			<ul style="list-style-type: none">• 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()

6. 시스템 명령어

6.1 IO 관련

6.1.1 set_digital_output(index, val =None)

- 기능

컨트롤러의 디지털 접점에서 신호를 보내내기 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

예외	설명
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)

6.1.2 set_digital_outputs(bit_list)

▪ 기능

컨트롤러의 디지털 출력 복수 개의 접점에서 신호를 내보내기 위한 명령문입니다.
bit_list에 정의된 접점들의 디지털 신호를 한 번에 출력할 수 있습니다.

▪ 인수

인수명	자료형	기본값	설명
bit_list	list (int)	-	복수 개 출력하고자 하는 접점 list • 양수의 접점 번호는 ON 출력: 1~16 • 음수의 접점 번호는 OFF 출력: -1~-16

▪ 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_outputs(bit_list=[1,2,3,4,5,6,7,8]) #1번~8번 접점 ON
set_digital_outputs([-1,-2,-3,-4,-5,-6,-7,-8]) #1번~8번 접점 OFF
set_digital_outputs([1,-2,3])                      #1번접점 ON, 2번 접점 OFF, 3번 접점
                                                ON
set_digital_outputs([4,-9,-12])                   #4번접점 ON, 9번 접점 OFF, 12번 접점
                                                OFF
  
```

6.1.3 set_digital_outputs(bit_start, bit_end, val)

▪ 기능

컨트롤러의 디지털 출력 시작 접점(bit_start)부터 마지막 접점(bit_end)까지 한 번에 복수 신호를 내보내기 위한 명령문입니다.

▪ 인수

인수명	자료형	기본값	설명
bit_start	int	-	출력 신호 시작 접점 번호 (1~16)
bit_end	int	-	출력 신호 끝 접점 번호 (1~16)
val	int	-	출력 값



알아두기

- bit_end 는 bit_start 보다 큰 값 이어야 합니다.
- val 은 bit_start =LSB, bit_end=MSB 가 되는 비트 조합의 값 입니다.
Ex) bit_start =1, bit_end=4, val=0b1010 # 4 번=ON, 3 번=OFF, 2 번=ON, 1 번=OFF

▪ 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

```
#접점1=ON, 접점2=ON, 접점3=OFF, 접점4=OFF 출력  
set_digital_outputs(bit_start=1, bit_end=4, val=0b0011) #0b는 2진수 의미
```

```
#접점3=ON, 접점4=OFF 출력  
set_digital_outputs(bit_start=3, bit_end=4, val=0b01) #0b는 2진수 의미
```

```
#접점1 ~ 접점 8까지 모두 ON 출력  
set_digital_outputs(1, 8, 0xff) #0x는 16진수 의미
```

6.1.4 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
```

6.1.5 get_digital_inputs(bit_list)

▪ 기능

컨트롤러의 디지털 입력 복수 개의 접점에서 신호를 불러오기 위한 명령문입니다. bit_list에 정의된 접점들의 디지털 신호를 한 번에 입력할 수 있습니다.

▪ 인수

인수명	자료형	기본값	설명
index	list (int)	-	복수 개로 읽어들일 input 접점 list 1 ~ 16까지의 숫자이며, 컨트롤러에 장착된 I/O 접점 번호

▪ 리턴

값	설명
int (>=0)	한번에 읽은 복수 개의 접점 값 (bit_list 의 첫번째 값=LSB, bit_list 의 마지막 값=MSB 가 되는 비트 조합의 값)
음수 값	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

```
# input 접점: 1번=OFF, 2번=OFF, 3번=ON, 4번=ON인 경우
```

```
res = get_digital_inputs(bit_list=[1,2,3,4])
```

```
#res 기대값 = 0b1100(이진수), 12(십진수), 0x0C(16진수)
```

```
# input 접점: 5번=ON, 6번=ON, 7번=OFF, 8번=ON인 경우
```

```
res = get_digital_inputs([5,6,7,8])
```

```
#res 기대값 = 0b1011(이진수), 11(십진수), 0x0B(16진수)
```

6.1.6 get_digital_inputs(bit_start, bit_end)

- 기능

컨트롤러의 디지털 입력 시작 접점(start_index)부터 마지막 접점(end_index)까지 한 번에 복수 신호를 불러오기 위한 명령문입니다.

- 인수

인수명	자료형	기본값	설명
bit_start	int	-	입력 신호 시작 접점 번호 (1~16)
bit_end	int	-	입력 신호 끝 접점 번호 (1~16)

 알아두기

bit_end 는 bit_start 보다 큰 값 이어야 합니다.

- 리턴

값	설명
int ($>= 0$)	한번에 읽은 복수 개의 접점 값 bit_start =LSB, bit_end=MSB 가 되는 비트 조합의 값
음수 값	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

```
# input 접점: 1번=OFF, 2번=OFF, 3번=ON, 4번=ON인 경우
res = get_digital_inputs(bit_start=1, bit_end=4)
#res 기대값 = 0b1100(이진수), 12(십진수), 0x0C(16진수)
```

6.1.7 wait_digital_input(index, val, timeout=None)

▪ 기능

컨트롤러의 디지털 입력 레지스터의 신호값이 val(ON or OFF)이 될 때까지 대기합니다. 대기 시간은 timeout 설정으로 변경할 수 있으며, 지정된 시간이 지나면 대기 상태가 종료됨과 동시에 결과를 리턴합니다. 단, timeout을 설정하지 않으면 무한 대기합니다.

▪ 인수

인수명	자료형	기본값	설명
index	int	-	1 ~ 16까지의 숫자이며, 컨트롤러에 장착됨 I/O index
value	int	-	I/O value • ON : 1 • OFF : 0
timeout	float	-	대기 시간 [sec] 설정하지 않으면 무한 대기

▪ 리턴

값	설명
0	성공
-1	실패 (time-out)

▪ 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

```
wait_digital_input(1, ON) #1번 접점이 ON될 때까지 무한 대기  
wait_digital_input(2, OFF) #2번 접점이 OFF될 때까지 무한 대기  
res = wait_digital_input(1, ON, 3) #1번 접점이 ON될 때까지 3초간 대기  
#3초 안에 1번 접점이 ON 되면, 대기 중지, res = 0  
#3초 안에 1번 접점이 ON 되지 않았으면, 대기 중지, res = -1
```

6.1.8 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 접점 ON
set_tool_digital_output(6, OFF) #로봇 암의 6번 접점 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
```

6.1.9 set_tool_digital_outputs(bit_list)

▪ 기능

로봇 툴의 신호를 디지털 접점에서 내보내기 위한 명령문으로 bit_list에 정의된 접점들의 디지털 신호를 한 번에 출력할 수 있습니다.

▪ 인수

인수명	자료형	기본값	설명
bit_list	list (int)	-	복수 개를 출력하고자 하는 접점 list • 양수의 접점 번호는 ON 출력: 1~6 • 음수의 접점 번호는 OFF 출력: -1~-6

▪ 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(bit_list=[1,2,3,4,5,6])  #1번~6번 접점 ON
set_tool_digital_outputs([-1,-2,-3,-4,-5,-6]) #1번~6번 접점 OFF
set_tool_digital_outputs([1,-2,3]) #1번접점 ON, 2번 접점 OFF, 3번 접점 ON
```

6.1.10 set_tool_digital_outputs(bit_start, bit_end, val)

▪ 기능

로봇 툴의 신호를 디지털 접점에서 내보내기 위한 명령문으로 시작 접점(bit_start)부터 마지막 접점(bit_end)까지 복수 신호를 한 번에 출력할 수 있습니다.

▪ 인수

인수명	자료형	기본값	설명
bit_start	int	-	출력 신호 시작 접점 번호 (1~6)
bit_end	int	-	출력 신호 끝 접점 번호 (1~6)
Val	int	-	출력 값



알아두기

- bit_end 는 bit_start 보다 큰 값 이어야 합니다.
- val 은 bit_start =LSB, bit_end=MSB 가 되는 비트 조합의 값입니다.
Ex) bit_start =1, bit_end=4, val=0b1010 # 4 번=ON, 3 번=OFF, 2 번=ON, 1 번=OFF

▪ 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

```
#접점1=ON, 접점2=ON, 접점3=OFF, 접점4=OFF 출력  
set_tool_digital_outputs(bit_start=1, bit_end=4, val=0b0011) #0b는 2진수를 의미
```

```
#접점3=ON, 접점4=OFF 출력  
set_tool_digital_outputs(bit_start=3, bit_end=4, val=0b01) #0b는 2진수를 의미
```

```
#접점1 ~ 접점 8까지 모두 ON 출력  
set_tool_digital_outputs(1, 8, 0xff) #0x는 2진수를 의미
```

6.1.11 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

6.1.12 get_tool_digital_inputs(bit_list)

- 기능

로봇 툴의 신호를 디지털 입력 접점에서 불러오기 위한 명령문으로 bit_list에 정의된 접점들의 디지털 신호를 한 번에 입력할 수 있습니다.

- 인수

인수명	자료형	기본값	설명
bit_list	list (int)	-	복수 개로 읽어들일 input 접점 list • (로봇 암에 장착된 I/O 입력 접점 번호, 1~6)

- 리턴

값	설명
int (>=0)	한 번에 읽은 복수 개의 접점 값 (bit_list 의 첫번째 값=LSB, bit_list 의 마지막 값=MSB 가 되는 비트 조합의 값)
음수 값	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

```
# input 접점: 1번=OFF, 2번=OFF, 3번=ON인 경우
res = get_tool_digital_inputs(bit_list=[1,2,3]) # 1,2,3번 접점 한 번에 읽기
#res 기대값 = 0b100(이진수), 4(십진수), 0x04(16진수)
```

```
# input 접점: 4번=ON, 5번=ON, 6번=OFF인 경우
res = get_tool_digital_inputs([4,5,6])
#res 기대값 = 0b011(이진수), 3(십진수), 0x03(16진수)
```

6.1.13 get_tool_digital_inputs(bit_start, bit_end)

- **기능**

로봇 툴의 신호를 디지털 접점에서 불러오기 위한 명령문으로 시작 접점(start_index)부터 마지막 접점(end_index)까지 복수 신호를 한 번에 입력할 수 있습니다.

- **인수**

인수명	자료형	기본값	설명
bit_start	int	-	입력 신호 시작 접점 번호 (1~6)
bit_end	int	-	입력 신호 끝 접점 번호 (1~6)

- **리턴**

값	설명
int (>=0)	한번에 읽은 복수 개의 접점 값 bit_start =LSB, bit_end=MSB 가 되는 비트 조합의 값
음수 값	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**

```
# input 접점: 1번=OFF, 2번=OFF, 3번=ON인 경우
res = get_tool_digital_inputs(bit_start=1, bit_end=3)
# res 기대값 = 0b100(이진수), 4(십진수), 0x04(16진수)

# input 접점: 4번=ON, 5번=ON, 6번=OFF인 경우
res = get_tool_digital_inputs(4, 6)
#res 기대값 = 0b011(이진수), 3(십진수), 0x03(16진수)
```

6.1.14 wait_tool_digital_input(index, val, timeout=None)

▪ 기능

로봇 툴의 디지털 입력 신호값이 val(ON or OFF)이 될 때까지 대기합니다. 대기 시간은 timeout 설정으로 변경할 수 있으며, 지정된 시간이 지나면 대기 상태가 종료됨과 동시에 결과를 리턴합니다. 단, timeout을 설정하지 않으면 무한 대기합니다.

▪ 인수

인수명	자료형	기본값	설명
index	int	-	1 ~ 6까지의 숫자이며, 로봇 암에 장착됨 I/O index
value	int	-	I/O value • ON : 1 • OFF : 0
timeout	float	-	대기 시간 [sec] 설정하지 않으면 무한 대기

▪ 리턴

값	설명
0	성공
-1	실패 (time-out)

▪ 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

```
wait_tool_digital_input(1, ON)    #1번 접점이 ON될 때까지 무한 대기  
wait_tool_digital_input(2, OFF)   #2번 접점이 OFF될 때까지 무한 대기
```

```
res = wait_tool_digital_input(1, ON, 3) #1번 접점이 ON될 때까지 3초간 대기  
#3초 안에 1번 접점이 ON 되면, 대기 중지, res = 0  
#3초 안에 1번 접점이 ON 되지 않았으면 대기 중지, res = -1
```

6.1.15 set_mode_analog_output(ch, mod)

▪ 기능

컨트롤러 아날로그 출력에 대한 채널 모드를 설정합니다.

▪ 인수

인수명	자료형	기본값	설명
ch	int	-	<ul style="list-style-type: none"> • 1 : channel 1 • 2 : channel 2
mod	int	-	analog io mode <ul style="list-style-type: none"> • 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)
```

6.1.16 set_mode_analog_input(ch, mod)

▪ 기능

컨트롤러 아날로그 입력 대한 채널 모드를 설정합니다.

▪ 인수

인수명	자료형	기본값	설명
ch	int	-	<ul style="list-style-type: none"> • 1 : channel 1 • 2 : channel 2
mod	int	-	analog io mode <ul style="list-style-type: none"> • 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)
```

6.1.17 set_analog_output(ch, val)

▪ 기능

컨트롤러 아날로그 출력에 해당하는 채널의 값을 출력합니다.

▪ 인수

인수명	자료형	기본값	설명
ch	int	-	<ul style="list-style-type: none"> • 1 : channel 1 • 2 : channel 2
val	float	-	analog 출력 값 <ul style="list-style-type: none"> • 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

```

6.1.18 get_analog_input(ch)

▪ 기능

컨트롤러 아날로그 입력에 해당하는 채널의 값을 불러옵니다.

▪ 인수

인수명	자료형	기본값	설명
ch	int	-	<ul style="list-style-type: none"> • 1 : channel 1 • 2 : channel 2

▪ 리턴

값	설명
float	<p>해당 channel 의 analog input 값</p> <ul style="list-style-type: none"> • 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.
```

6.2 TP 연동

6.2.1 tp_popup(message, pm_type=DR_PM_MESSAGE, button_type=0)

■ 기능

터치 팬던트를 통해 사용자에게 메시지를 제공합니다. 상위 제어기는 String을 받아서 팝업 창에 표시하며, 사용자의 확인 작업에 의해 창이 닫혀야 합니다.

■ 인수

인수명	자료형	기본값	설명
message	string	-	사용자에게 제공할 메시지 (메시지는 256byte 이내로 제한됩니다.)
pm_type	int	DR_PM_MESSAGE	메시지 유형 • DR_PM_MESSAGE • DR_PM_WARNING • DR_PM_ALARM
button_type	int	0	TP pop 메시지 버튼 type • 0 : Stop & Resume 버튼 표시 • 1 : Stop 버튼 표시

■ 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

```
tp_popup("move done", DR_PM_MESSAGE)
tp_popup("Error!! ", DR_PM_ALARM)
a=1
b=2
c=3
tp_popup("a={0}, b={1}, c={2}" .format(a,b,c) ,DR_PM_MESSAGE)
tp_popup("critical error!! ", DR_PM_ALARM, 1)
```

6.2.2 tp_log(message)

- 기능

티치 팬던트에 사용자가 작성한 로그를 기록합니다.

- 인수

인수명	자료형	기본값	설명
message	string	-	로그 메시지 (메시지는 256byte 이내로 제한됩니다.)

- 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 애러 발생 시
DR_Error (DR_ERROR_STOP)	프로그램 강제 종료 시

- 예제

```
tp_log("movej() is complete! ")
```

6.2.3 tp_progress(cur_progress, total_progress)

- **기능**

티치 팬던트를 통해 사용자에게 메시지를 제공합니다. 상위 제어기는 유형화 프로그램을 실행할 때, 실행 단계의 정보를 받아 GUI 상에 표시합니다.

- **인수**

인수명	자료형	기본값	설명
cur_progress	int	-	현재 단계 값
total_progress	int	-	최종 단계 값

- **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**

```
tp_progress(1, 100)
tp_progress(99, 100)
```

6.2.4 tp_get_user_input(message, input_type)

- 기능

티치 팬던트를 통해 사용자 입력 정보를 받습니다.

- 인수

인수명	자료형	기본값	설명
message	string	-	TP 사용자 입력창에 표시될 문자열 메시지
input_type	int	-	TP 사용자 입력 메시지 타입 • DR_VAR_INT: 정수형 • DR_VAR_FLOAT: 실수형 • DR_VAR_STR: 문자열

- 리턴

값	설명
사용자 입력 데이터	TP로 받은 사용자 입력 데이터

- 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(10, 10, 10, 10, 10, 10)
q2 = posj(20, 20, 20, 20, 20, 20)
q3 = posj(30, 30, 30, 30, 30, 30)
q4 = posj(40, 40, 40, 40, 40, 40)
q5 = posj(50, 50, 50, 50, 50, 50)
q6 = posj(60, 60, 60, 60, 60, 60)

int_y= tp_get_user_input("message1", input_type= DR_VAR_INT)
if int_y==1:    #TP 사용자 입력이 1로 들어온 경우 q1으로 이동
    movej(q1, vel=30, acc=30)
else:           #TP 사용자 입력이 1로 들어오지 않은 경우 q2으로 이동
    movej(q2, vel=30, acc=30)

float_y= tp_get_user_input("message2", input_type= DR_VAR_FLOAT)
if float_y==3.14:      #TP 사용자 입력이 3.14로 들어온 경우 q3으로 이동
    movej(q3, vel=30, acc=30)
else:                  #TP 사용자 입력이 3.14로 들어오지 않은 경우 q4으로 이동
    movej(q4, vel=30, acc=30)

str_y= tp_get_user_input("message3", input_type= DR_VAR_STR)
if str_y=="a":         #TP 사용자 입력이 "a"로 들어온 경우 q5으로 이동
    movej(q5, vel=30, acc=30)
else:                  #TP 사용자 입력이 "a"로 들어오지 않은 경우 q6으로 이동
    movej(q6, vel=30, acc=30)
```

6.3 Thread

6.3.1 `thread_run(th_func_name, loop=False)`

▪ 기능

Thread를 생성하여 수행하며 Thread가 수행할 기능은 th_func_name에 지정된 함수에 따라 결정됩니다.

알아두기

Thread 명령어를 사용할 때, 제약 조건은 아래와 같습니다.

- Thread는 최대 4개까지만 사용 가능합니다.
- Thread 내에서 로봇을 움직이는 하기 모션 명령어는 사용할 수 없습니다.
 - movej, amovej, movejx, amovejx, movel, amovel, movec, amovec, movesj, amovesj,
 - movesx, amovesx, moveb, amoveb, move_spiral, amove_spiral,
 - move_periodic, amove_periodic, move_home
- thread_run 시 loop=True 하고, 해당 쓰레드 함수 내에서 무한루프로 블록되어 있는 경우에는 쓰레드 명령들이 정상 동작하지 않습니다. (단, TP를 통한 STOP 시, 해당 쓰레드는 정상적으로 종료 됩니다.)

▪ 인수

인수명	자료형	기본값	설명
th_func_name	callable	-	Thread가 수행할 function name
loop	bool	False	Thread의 반복 수행 여부 <ul style="list-style-type: none"> • True : th_func_name이 반복적 호출 (interval 0.01second) • False : th_func_name이 1회 호출

▪ 리턴

값	설명
int	등록된 thread ID
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

```
#----- 쓰레드 -----
def fn_th_func():
    if check_motion() == 0:      # 수행 중인 모션이 없는 경우
        set_digital_output(1, OFF)
    else:
        set_digital_output(1, ON)

#----- 메인 루틴 -----
th_id = thread_run(fn_th_func, loop=True) #쓰레드 Run

while 1:
    # do something...
    wait(0.1)
```

6.3.2 thread_stop(th_id)

- 기능

Thread를 종료합니다.

DRL 프로그램이 종료될 경우에 thread_stop() 명령을 사용하지 않아도 자동으로 종료됩니다.

- 인수

인수명	자료형	기본값	설명
th_id	int	-	Stop 하고자 하는 thread ID

- 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

```
def fn_th_func():
    if check_motion()==0:      # 수행 중인 모션이 없는 경우
        set_digital_output(1, OFF)
    else:
        set_digital_output(1, ON)
    #----- 메인 루틴 -----
    th_id = thread_run(fn_th_func, loop=True)

    # do something...
    thread_stop(th_id) #쓰레드 종료
```

6.3.3 thread_pause(th_id)

- **기능**

Thread를 일시 정지합니다.

- **인수**

인수명	자료형	기본값	설명
th_id	int	-	일시 정지하고자 하는 thread ID

- **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**

```

def fn_th_func():
    if check_motion() == 0:      # 수행 중인 모션이 없는 경우
        set_digital_output(1, OFF)
    else:
        set_digital_output(1, ON)
    #----- 메인 루틴 -----
    th_id = thread_run(fn_th_func, loop=True)

    # do something...

    thread_pause(th_id) #쓰레드 일시 정지

```

6.3.4 thread_resume(th_id)

- 기능

일시 정지된 Thread를 다시 시작합니다.

- 인수

인수명	자료형	기본값	설명
th_id	int	-	일시 정지를 풀고 재개하고자 하는 thread ID

- 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

```
def fn_th_func():
    if check_motion() == 0:      # 수행 중인 모션이 없는 경우
        set_digital_output(1, OFF)
    else:
        set_digital_output(1, ON)

#----- 메인 루틴 -----
th_id = thread_run(fn_th_func, loop=True)
```

```
# do something...
thread_pause(th_id) #쓰레드 일시 정지

# do something...
thread_resume(th_id) #쓰레드 일시 정지 해제
```

6.3.5 thread_state(th_id)

- 기능

Thread의 상태를 확인합니다.

- 인수

인수명	자료형	기본값	설명
th_id	int	-	상태 정보를 알고자 하는 thread ID

- 리턴

값	설명
1	RUN (TH_STATE_RUN)
2	PAUSE (TH_STATE_PAUSE)
3	STOP (TH_STATE_STOP)

- 예외

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

- 예제

```

def fn_th_func():
    if check_motion() == 0:      # 수행 중인 모션이 없는 경우
        set_digital_output(1, OFF)
    else:
        set_digital_output(1, ON)

    th_id = thread_run(fn_th_func, loop=True)
    state1 = thread_state(th_id)

    thread_pause(th_id)
    state2 = thread_state(th_id)

```

6.3.6 통합 예제

예제를 통한 쓰레드 사용법을 설명합니다.

- 예제 1: 쓰레드 예제

```
#----- thread 1: client comm. -----
def fn_th_client():
    global g_sock
    global g_cmd
    res, rx_data = client_socket_read(g_sock)
    if res > 0:
        g_cmd = rx_data.decode() #decode: byte형을 string으로 변환
    else: # 통신에러가 발생한 경우
        client_socket_close(g_sock)
        exit() #프로그램 종료
    wait(0.1)
    return 0

#----- thread 2: check IO -----
def fn_th_check_io():
    if get_digital_input(1) == ON:
        exit() #프로그램 종료
    wait(0.1)
    return 0

#----- main -----
g_sock = client_socket_open("192.168.137.2", 20002) #서버에 접속
g_cmd = ""

g_th_id1 = thread_run(fn_th_client, loop=True) # th_client 쓰레드 실행
g_th_id2 = thread_run(fn_th_check_io, loop=True) # th_check_io 쓰레드 실행

p1 = posj(0, 0, 90, 0, 90, 0)
p2 = posj(10, 0, 90, 0, 90, 0)
```

```
p3 = posj(20, 0, 90, 0, 90, 0)

while 1:
    if g_cmd == "a":
        g_cmd = ""
        movej(p1,vel=100,acc=100)
        client_socket_write(g_sock, b"end")
    if g_cmd == "b":
        g_cmd = ""
        movej(p2,vel=100,acc=100)
        client_socket_write(g_sock, b"end")
    if g_cmd == "c":
        g_cmd = ""
        movej(p3,vel=100,acc=100)
        client_socket_write(g_sock, b"end")
    wait(0.1)
```

th_client 쓰레드: server로부터 수신한 데이터를 string형으로 변하여 g_cmd에 저장

th_check_io 쓰레드: 1번 접점의 상태를 체크하여 ON 이면 프로그램 종료.

main : 서버에 접속,

2개의 쓰레드 실행 : th_client, th_check_io

서버로부터 "a"가 수신되면 p1으로 이동 후, 서버에게 "end" 전송

서버로부터 "b"가 수신되면 p2으로 이동 후, 서버에게 "end" 전송

서버로부터 "c"가 수신되면 p3으로 이동 후, 서버에게 "end" 전송

6.4 기타

6.4.1 wait(time)

- 기능

지정된 시간만큼 대기합니다.

- 인수

인수명	자료형	기본값	설명
Time	Float	-	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

```
wait(1.3) # 1.3초 대기
```

```
while 1: #0.1초 마다 접점 1번 상태를 체크
    if get_digital_input(1) == ON:
        set_digital_output(1, ON)
    wait(0.1)
```

6.4.2 exit()

- **기능**

현재 수행 중인 프로그램을 종료합니다.

- **리턴**

값	설명
0	성공

- **예외**

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

- **예제**

```
exit()
```

6.4.3 sub_program_run(name)

▪ 기능

별도의 파일로 작성된 서브프로그램을 실행합니다.

▪ 인수

인수명	자료형	기본값	설명
name	string	-	서브 프로그램 이름

▪ 리턴

값	설명
module	실행된 서브 프로그램의 모듈 객체

▪ 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

- 서브프로그램의 첫줄에 반드시 "from DRCF import *" 구문이 있어야만 합니다.
- 티칭 펜던트 사용하는 경우, 본 구문이 자동 삽입됩니다.
- 메인프로그램과 서브프로그램의 전역 변수 이름이 같은 경우, 각각 다른 변수로 동작함에 유의 바랍니다. 즉, 서로 변수 참조가 되지 않습니다.
- 메인프로그램과 서브프로그램간에 변수 공유가 필요할 경우는 시스템 변수를 사용합니다.
- 시스템 변수는 티칭 펜던트를 통해서 설정합니다. 자세한 사용법은 사용자 매뉴얼을 참조바랍니다.

- 예제

```
# subProgramA, subProgramB 가 사전에 작성 저장되어 있어야 합니다.  
<subProgramA.drl>  
from DRCF import *  
movej([0,0,90,0,90,0], vel=30, acc=30)  
  
<subProgramB.drl>  
from DRCF import *  
movej[(10,0,90,0,90,0), vel=30, acc=30]  
  
<main program>  
while True:  
    var_select = tp_get_user_input("Select File", DR_VAR_INT)  
    if var_select == 0:  
        sub_program_run("subProgramA") # subProgramA 실행  
    elif var_select == 1:  
        sub_program_run("subProgramB") # subProgramB 실행
```

6.4.4 drl_report_line(option)

▪ 기능

DRL 스크립트 구동 시, 실행라인 표시 기능을 ON/OFF 할 수 있는 명령어입니다. 실행라인 표시 기능을 OFF 하면, 실행라인 표시 기능을 수행하기 위한 시간이 줄어들어 DRL 실행 속도가 현저히 빨라지게 됩니다.

주의

실행라인 표시 기능을 OFF 한 구간에서는 하기의 기능이 동작하지 않습니다.

- 라인 별 실행 시간 표시
- 변수 모니터링
- 시스템 변수 업데이트
- Debug 모드에서 Step by Step
- Debug 모드에서 Brake Point

▪ 인수

인수명	자료형	기본값	설명
option	Int	-	DRL 실행 라인 표시 여부 ON(1), OFF(0)

▪ 리턴

값	설명
없음	-

▪ 예제

```
x=0
y=0
drl_report_line(OFF) #실행라인 표시 기능 OFF
while x < 1000:      #실행라인 표시 안 됨 (실행 속도 향상)
    x += 1          #실행라인 표시 안 됨 (실행 속도 향상)
drl_report_line(ON) #실행라인 표시 기능 ON
x=0                 #실행라인 표시 됨
y=0                 #실행라인 표시 됨
```

6.4.5 set_fm(key, value)

▪ 기능

기 정의되어 KT Smart Factory와 연동되고 있는 시스템 정보 이외에 프로그램 실행 시 생성된 변수(전역 변수 및 시스템 변수 등)정보에 대하여 연동이 필요한 경우에 사용하는 명령어 입니다.

주의

Setup 메뉴의 KT Smart Factory 메뉴에서 연동 정보 미설정 시 기능이 동작하지 않음에 유의 바랍니다. KT Smart Factory 메뉴는 KT 전용 라이선스 설정 시에만 표시됩니다.

▪ 인수

인수명	자료형	기본값	설명
key	string	-	연동 데이터 이름
value	int float string	-	연동 데이터 변수 가능한 데이터 타입 <ul style="list-style-type: none"> • 정수형 데이터 • 실수형 데이터 • 문자열 데이터

▪ 리턴

값	설명
없음	-

▪ 예제

```
count = 0

movej(posj(0, 0, 90, 0,90,0), vel=30, acc=30)
while True:
    movej(posj(0, 0, -90, 0,90,0), vel=30, acc=30)
    movej(posj(0, 0, 90, 0,90,0), vel=30, acc=30)
    count = count + 1
    set_fm("TotalCount", count)
```

7. 수학 함수

7.1 $\sin(x)$

- **기능**

x radians의 sine 값을 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
x	float	-	-

- **리턴**

값	설명
the sine of x	-

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.2 **cos(x)**

- **기능**

x radians의 cosine 값을 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
x	float	-	-

- **리턴**

값	설명
the cosine of x	

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.3 tan(x)

- **기능**

x radians의 tangent 값을 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
x	float	-	-

- **리턴**

값	설명
the tangent of x	-

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.4 **asin(x)**

- **기능**

x radians의 arc sine of 값을 리턴합니다

- **인수**

인수명	자료형	기본값	설명
x	float	-	

- **리턴**

값	설명
the arc sine of x	-

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.5 **acos(x)**

▪ 기능

x radians의 arc cosine of 값을 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
x	float	-	

▪ 리턴

값	설명
the arc cosine of x	-

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.6 **atan(x)**

- **기능**

x radians의 arc tangent of 값을 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
x	float	-	-

- **리턴**

값	설명
the arc tangent of x	-

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.7 atan2(y, x)

▪ 기능

y/x radians의 arc tangent of 값을 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
y	float	-	-
x	float	-	-

▪ 리턴

값	설명
the arc tangent of y/x	The result is between -pi and pi

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.8 **ceil(x)**

- **기능**

x값보다 큰 정수 중 가장 작은 정수 값을 리턴합니다. 단, 소수점 이하는 모두 올립니다.

- **인수**

인수명	자료형	기본값	설명
x	float	-	-

- **리턴**

값	설명
rounded integer	-

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.9 floor(x)

▪ 기능

x 보다 작은 정수 중 가장 큰 정수 값을 리턴합니다. 단, 소수점 이하는 모두 버립니다.

▪ 인수

인수명	자료형	기본값	설명
x	float	-	-

▪ 리턴

값	설명
rounded integer	-

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.10 **pow(x, y)**

- **기능**

Return x raised to the power of y.

- **인수**

인수명	자료형	기본값	설명
x	float	-	
y	float	-	

- **리턴**

값	설명
x raised to the power y	-

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.11 sqrt(x)

▪ 기능

x의 제곱근을 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
x	float	-	-

▪ 리턴

값	설명
the square root of x	성공

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.12 **log(x, b)**

- **기능**

x의 로그를 밑수 b로 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
x	float	-	-
b	float	-	base, e (natural logarithm)

- **리턴**

값	설명
the logarithm of f to the base of b.	-

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.13 d2r(x)

▪ 기능

x radians 값을 degree로 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
x	float	-	The angle in degrees

▪ 리턴

값	설명
The angle in radians	-

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.14 **r2d(x)**

- **기능**

x radians 값을 degree로 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
x	float		The angle in radians

- **리턴**

값	설명
The angle in degrees	-

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.15 norm(x)

▪ 기능

x의 L2 norm을 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
x	float[3]	-	Point 좌표(x, y, z)

▪ 리턴

값	설명
float	Point 좌표 벡터의 크기

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.16 **random()**

- **기능**

0 이상 1 미만의 난수를 리턴합니다.

- **리턴**

값	설명
random number	0 이상 1 미만의 난수(float)

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

7.17 rotx(angle)

▪ 기능

x축을 기준으로 angle 값만큼 회전시키는 회전행렬을 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
angle	float	0	회전각 [deg]

▪ 리턴

값	설명
float[3][3]	회전행렬

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
rotm = rotx(30)
```

7.18 **roty(angle)**

- **기능**

y축을 기준으로 angle 값만큼 회전시키는 회전행렬을 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
angle	float	0	회전각 [deg]

- **리턴**

값	설명
float[3][3]	회전행렬

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

- **예제**

```
rotm = roty(30)
```

7.19 rotz(angle)

▪ 기능

*z*축을 기준으로 *angle* 값만큼 회전시키는 회전행렬을 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
angle	float	0	회전각 [deg]

▪ 리턴

값	설명
float[3][3]	회전행렬

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
rotm = rotz(30)
```

7.20 **rotm2eul(rotm)**

- **기능**

회전행렬을 받아 Euler angle(zyz order)을 degree 값으로 리턴합니다. 단, 결과값으로 리턴되는 Euler angle(rx, ry, rz) 중 ry 값은 항상 양수로 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
rotm	Float[3][3]	-	회전행렬

- **리턴**

값	설명
float[3]	ZYZ Euler angle [deg]

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

- **예제**

```
rotm = [[1,0,0],[0,0.87,-0.5],[0,0.5,0.87]]  
eul = rotm2eul(rotm)
```

7.21 rotm2rotvec(rotm)

▪ 기능

회전행렬을 받아 rotation vector(angle/axis representation)를 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
rotm	float[3][3]	-	회전행렬

▪ 리턴

값	설명
float[3]	rotation vector

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
rotm = [[1,0,0],[0,0.87,-0.5],[0,0.5,0.87]]
eul = rotm2rotvec(rotm)
```

7.22 **eul2rotm([alpha,beta,gamma])**

▪ 기능

Euler angle(zyz order)을 회전행렬로 변환합니다.

▪ 인수

인수명	자료형	기본값	설명
eul	float[3]	[0 0 0]	Euler angle (zyz) [deg]

▪ 리턴

값	설명
float[3][3]	회전행렬

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
eul = [90, 90, 0]
rotm = eul2rotm (eul)
```

7.23 eul2rotvec([alpha,beta,gamma])

▪ 기능

Euler angle(zyz order)을 rotation vector로 변환합니다.

▪ 인수

인수명	자료형	기본값	설명
eul	float[3]	[0 0 0]	Euler angle (zyz) [deg]

▪ 리턴

값	설명
float[3]	rotation vector

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
eul = [90, 90, 0]
rotvec = eul2rotvec (eul)
```

7.24 **rotvec2eul([rx,ry,rz])**

▪ 기능

Rotation vector를 Euler Angle(zyz)로 변환합니다.

▪ 인수

인수명	자료형	기본값	설명
rotvec	float[3]	-	rotation vector

▪ 리턴

값	설명
float[3]	ZYZ Euler angle [deg]

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
rotvec = [0.7854, 0, 0]
eul = rotvec2eul(rotvec) # eul=[45,0,0]
```

7.25 **rotvec2rotm([rx,ry,rz])**

- **기능**

rotation vector를 rotation matrix로 변환합니다.

- **인수**

인수명	자료형	기본값	설명
rotvec	float[3]	-	rotation vector

- **리턴**

값	설명
float[3][3]	회전행렬

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

- **예제**

```
rotm = rotvec2eul([0.7854,0,0])
```

7.26 htrans(posx1,posx2)**▪ 기능**

posx1과 posx2로부터 구한 Homogeneous Transformation matrix를 T1, T2라 할 때 $T1 \cdot T2$ 에 해당하는 자세를 리턴합니다.

$$H_1 H_2 = \begin{bmatrix} R_1 & r_1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R_2 & r_2 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R_1 R_2 & r_1 + R_1 r_2 \\ 0 & 1 \end{bmatrix}$$

▪ 인수

인수명	자료형	기본값	설명
posx1	posx list (float[6])	-	posx 또는 position list [mm, deg]
posx2	posx list (float[6])	-	posx 또는 position list [mm, deg]

▪ 리턴

값	설명
posx	[mm, deg]

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
posx1 = [100, 20, 300, 90, 0, 180]
posx2 = [200, 50, 100, 90, 30, 150]
posx = htrans(posx1,posx2)
```

7.27 get_intermediate_pose(posx1, posx2, alpha)

▪ 기능

posx1으로부터 posx2로의 linear transition 중 alpha에 위치한 posx를 리턴합니다. alpha가 0이면 posx1을, alpha가 0.5이면 두 자세의 중간값을, alpha가 1이면 posx2를 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
posx1	posx list (float[6])	-	posx 또는 position list [mm, deg]
posx2	posx list (float[6])	-	posx 또는 position list [mm, deg]
alpha	float	-	$0.0 \leq \text{alpha} \leq 1.0$

▪ 리턴

값	설명
posx	[mm, deg]

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
posx1 = [100, 20, 300, 90, 0, 180]
posx2 = [200, 50, 100, 90, 30, 150]
alpha = 0.5
posx = get_intermediate_pose(posx1, posx2, alpha)
```

get_distance(posx1, posx2)

7.28 get_distance(posx1, posx2)

▪ 기능

두 pose 포지션 사이의 거리를 [mm] 단위로 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
posx1	posx list (float[6])	-	posx 또는 position list [mm]
posx2	posx list (float[6])	-	posx 또는 position list [mm]

▪ 리턴

값	설명
float	[mm]

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
posx1 = [100, 20, 300, 90, 0, 180]  
posx2 = [200, 50, 100, 90, 30, 150]  
dis_posx = get_distance(posx1, posx2)
```

7.29 get_normal(x1, x2, x3)

▪ 기능

3개의 작업 공간 지점(posx)으로 이루어진 평면의 normal vector를 리턴합니다. 방향은 시계방향 기준입니다.

▪ 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

▪ 리턴

값	설명
float[3]	normal vector

▪ 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, 30, 0, 90)
x2 = posx(500, 0, 700, 0, 0, 45)
x3 = posx(300, 100, 500, 45, 0, 45)
vect = get_normal(x1, x2, x3)

```

add_pose(posx1, posx2)

7.30 add_pose(posx1, posx2)

▪ 기능

두 자세의 합을 구합니다.

$$\text{add_pose}\left(\begin{bmatrix} R_1 & r_1 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} R_2 & r_2 \\ 0 & 1 \end{bmatrix}\right) \Rightarrow \begin{bmatrix} R_1 R_2 & r_1 + r_2 \\ 0 & 1 \end{bmatrix}$$

▪ 인수

인수명	자료형	기본값	설명
posx1	posx list (float[6])	-	posx 또는 position list [mm, deg]
posx2	posx list (float[6])	-	posx 또는 position list [mm, deg]

▪ 리턴

값	설명
posx	[mm, deg]

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
posx1 = [100, 20, 300, 90, 0, 180]
posx2 = [200, 50, 100, 90, 30, 150]
add_posx = add_pose(posx1, posx2)
```

7.31 subtract_pose(posx1, posx2)

▪ 기능

두 자세의 차이를 구합니다.

$$\text{subtract_pose}(\begin{bmatrix} R_1 & r_1 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} R_2 & r_2 \\ 0 & 1 \end{bmatrix}) \Rightarrow \begin{bmatrix} R_2^T R_1 & r_1 - r_2 \\ 0 & 1 \end{bmatrix}$$

▪ 인수

인수명	자료형	기본값	설명
posx1	posx list (float[6])	-	posx 또는 position list [mm, deg]
posx2	posx list (float[6])	-	posx 또는 position list [mm, deg]

▪ 리턴

값	설명
posx	[mm, deg]

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
posx1 = [100, 20, 300, 90, 0, 180]
posx2 = [200, 50, 100, 90, 30, 150]
subtract_posx = subtract_pose(posx1, posx2)
```

7.32 **inverse_pose(posx1)**

▪ 기능

posx의 inverse에 해당하는 posx 값을 리턴합니다.

$$\text{inv_pose} \left(\begin{bmatrix} R_1 & r_1 \\ 0 & 1 \end{bmatrix} \right) = \begin{bmatrix} R_1 & r_1 \\ 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} R_1^T & -R_1^T r_1 \\ 0 & 1 \end{bmatrix}$$

▪ 인수

인수명	자료형	기본값	설명
posx1	posx list (float[6])	-	posx 또는 position list [mm, deg]

▪ 리턴

값	설명
posx	[mm, deg]

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
posx1 = [100, 20, 300, 90, 0, 180]
inv_posx = inverse_pose(posx1)
```

7.33 dot_pose(posx1, posx2)

▪ 기능

두 개의 pose가 주어졌을 때 translation 성분의 내적을 구합니다.

▪ 인수

인수명	자료형	기본값	설명
posx1	posx list (float[6])	-	posx 또는 position list [mm, deg]
posx2	posx list (float[6])	-	posx 또는 position list [mm, deg]

▪ 리턴

값	설명
float	두 개 pose의 내적 결과

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
posx1 = [100, 20, 300, 90, 0, 180]
posx2 = [200, 50, 100, 90, 30, 150]
res= dot_pose(posx1, posx2)
```

cross_pose(posx1, posx2)

7.34 **cross_pose(posx1, posx2)**

▪ 기능

두 개의 pose가 주어졌을 때 translation 성분의 외적을 구합니다.

▪ 인수

인수명	자료형	기본값	설명
posx1	posx list (float[6])	-	posx 또는 position list [mm, deg]
posx2	posx list (float[6])	-	posx 또는 position list [mm, deg]

▪ 리턴

값	설명
float[3]	두 개 pose의 외적 결과

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
posx1 = [100, 20, 300, 90, 0, 180]  
posx2 = [200, 50, 100, 90, 30, 150]  
res= cross_pose(posx1, posx2)
```

7.35 unit_pose(posx1)

▪ 기능

주어진 posx translation 성분의 단위 백터를 구합니다.

▪ 인수

인수명	자료형	기본값	설명
posx1	posx list (float[6])	-	posx 또는 position list [mm, deg]

▪ 리턴

값	설명
float[3]	주어진 posx의 단위 백터

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

▪ 예제

```
posx1 = [100, 20, 300, 90, 0, 180]
```

```
res = unit_pose(posx1)
```

8. 외부 통신 명령어

8.1 Serial

8.1.1 `serial_open(port=None, baudrate=115200, bytesize=DR_EIGHTBITS, parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE)`

- **기능**

Serial 통신 포트를 오픈합니다.

- **인수**

인수명	자료형	기본값	설명
port	string	None	<ul style="list-style-type: none"> • D-SUB(9 pin) 연결 : "COM" • USB to Serial 연결 : "COM_USB"
baudrate	int	115200	Baud rate 2400, 4800, 9600, 19200, 38400, 57600, 115200
bytesize	int	8	<p>데이터 bit 수</p> <ul style="list-style-type: none"> • DR_FIVEBITS: 5 • DR_SIXBITS: 6 • DR_SEVENBITS: 7 • DR_EIGHTBITS: 8
parity	str	"N"	<p>Parity checking</p> <ul style="list-style-type: none"> • DR_PARITY_NONE: "N" • DR_PARITY EVEN: "E" • DR_PARITY ODD: "O" • DR_PARITY MARK: "M" • DR_PARITY SPACE: "S"
stopbits	int	1	<p>Stop bit의 수</p> <ul style="list-style-type: none"> • DR_STOPBITS_ONE = 1 • DR_STOPBITS_ONE_POINT_FIVE = 1.5 • DR_STOPBITS_TWO = 2

■ 리턴

값	설명
serial.Serial instance	연결 성공

■ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	Serial.SerialException 예외 발생

■ 예제

```
# 시리얼 포트 D-SUB(9 pin)에 연결 한 경우
ser = serial_open(port="COM", baudrate=115200, bytesize=DR_EIGHTBITS,
parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE)
```

```
res = serial_write(ser, b"123ABC")
```

```
serial_close(ser)
```

```
# USB to serial 장비를 USB 포트에 연결 한 경우
```

```
ser = serial_open(port="COM_USB", baudrate=115200, bytesize=DR_EIGHTBITS,
parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE)
```

```
res = serial_write(ser, b"123ABC")
```

```
serial_close(ser)
```

8.1.2 serial_close(ser)

- 기능

Serial 통신 포트를 닫습니다.

- 인수

인수명	자료형	기본값	설명
ser	serial.Serial	-	Serial instance

- 리턴

값	설명
0	Serial 포트 닫기 성공

- 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수의 데이터형 오류 시

- 예제

```
ser = serial_open(port="COM", baudrate=115200, bytesize=DR_EIGHTBITS,  
parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE)
```

```
res = serial_write(ser, b"123456789")
```

```
serial_close(ser)
```

8.1.3 serial_state(ser)

- **기능**

Serial 통신 포트의 상태를 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
ser	serial.Serial	-	Serial instance

- **리턴**

값	설명
1	Serial 포트 opened 상태
0	Serial 포트 closed 상태

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수의 데이터형 오류 시

- **예제**

```
ser = serial_open(port="COM", baudrate=115200, bytesize=DR_EIGHTBITS,
                  parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE)

state = serial_state(ser)

serial_close(ser)
```

8.1.4 serial_set_inter_byte_timeout(ser, timeout=None)

▪ 기능

포트에 읽기/쓰기를 수행할 때 바이트 간(inter-byte)의 timeout을 설정합니다.

▪ 인수

인수명	자료형	기본값	설명
ser	serial.Serial	-	Serial instance
timeout	float	None	읽기/쓰기 수행 시, 바이트 간의 timeout • timeout 발생 이전까지 처리된 데이터를 연속적인 데이터로 처리 • None : inter-byte timeout을 지정하지 않음

▪ 리턴

값	설명
0	성공

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수의 데이터형 오류 시

▪ 예제

```

ser = serial_open(port="COM", baudrate=115200, bytesize=DR_EIGHTBITS,
                  parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE)

res = serial_set_inter_byte_timeout(ser, 0.1)

serial_close(ser))

```

8.1.5 serial_write(ser, tx_data)

■ 기능

Serial 포트에 데이터(tx_data)를 기록합니다.

■ 인수

인수명	자료형	기본값	설명
ser	serial.Serial	-	Serial instance
tx_data	byte	-	송신할 데이터 • 데이터 타입 byte 형이어야 합니다. • 하기 예제 참조 바랍니다.

■ 리턴

값	설명
0	성공
-1	Port가 open 상태가 아닙니다.
-2	serial.SerialException 예외 발생

■ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수의 데이터형 오류 시

■ 예제

```
ser = serial_open(port="COM", baudrate=115200, bytesize=DR_EIGHTBITS,
                  parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE)
```

serial_write(ser, b"123456789") #b 는 bytes 형을 의미합니다.

```
# string 을 byte 형으로 변환
msg = "abcd"           # msg는 string 변수
serial_write(ser, msg.encode()) # encode()는 string형을 byte형으로 변환

serial_close(ser)
```

8.1.6 serial_read(ser, length=-1, timeout=-1)

▪ 기능

Serial 포트에서 데이터를 읽어옵니다.

▪ 인수

인수명	자료형	기본값	설명
Ser	serial.Serial	-	Serial instance
Length	int	-1	Read 할 바이트 수 • -1: 미지정(read 된 데이터만큼 읽습니다.) • n(>=0): 지정한 바이트 수만큼 읽습니다.
timeout	int float	-1	Read 대기 시간 • -1: 무한 대기 • n(>0): n 초(second)

▪ 리턴

값(res, rx_data)	설명	
res	n	수신한 데이터의 바이트 수
	-1	Port가 open 상태가 아닙니다.
	-2	serial.SerialException 예외 발생
rx_data	Read 한 데이터(byte type)	

▪ 예외

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

■ 예제

```
ser = serial_open(port="COM", baudrate=115200, bytesize=DR_EIGHTBITS,
                  parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE)

res, rx_data = serial_read(ser)
#데이터가 수신될 때 까지 무한 대기

res, rx_data = serial_read(ser, timeout=3)
#데이터가 수신될 때 까지 대기, 3초 타임 아웃 설정
#3초 이내 수신되면 읽은 데이터 바로 리턴
#3초가 지나면 현재까지 읽은 값을 리턴한다

res, rx_data = serial_read(ser, length=100)
#100byte 읽을 때까지 무한대기

res, rx_data = serial_read(ser, length=100, timeout=3)
#100byte 읽을 때 까지 대기, 3초 타임 아웃 설정
#3초 이내 100byte 수신되면 읽은 데이터 바로 리턴
#3초가 지나면 현재까지 읽은 값을 리턴한다

#수신된 byte형을 string 형으로 변환
rx_msg = rx_data.decode() #rx_data는 byte형이고 string형으로 변환하기 위해서는
                          #decode() 사용합니다.
                          #예를 들어, rx_data = b"abcd" 이면,
                          #rx_msg="abcd"가 됩니다.

res, rx_data = serial_close(ser)
```

8.1.7 serial_get_count()

- 기능

연결된 USB to Serial의 포트정보, 장치이름을 읽어옵니다.

- 인수

인수명	자료형	기본값	설명
없음	-	-	

- 리턴

값(port_info, device_name)	설명
count	연결된 시리얼 포트 개수

- 예외

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

- 예제

```
count = serial_get_count() # 연결된 시리얼 포트 개수 읽기

for i in range(count):
    port_info, device_name = serial_get_info(i+1)
    tp_popup("i={}, port ={}, dev ={}".format(i, port_info, device_name))
```

8.1.8 serial_get_info(id)

- 기능

연결된 USB to Serial의 포트정보, 장치이름을 읽어옵니다.

- 인수

인수명	자료형	기본값	설명
id	int	1	읽고싶은 USB to Serial의 ID (1~10)

- 리턴

값(port_info, device_name)	설명
port_info	포트정보 (값이 NULL인 경우 연결된 장치 없음)
device_name	장치이름 (값이 NULL인 경우 연결된 장치 없음)

- 예외

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

- 예제

```
port_info, device_name = serial_get_info(1) #1번쨰 연결된 장치의 정보조회
#port_info = "COM_USB"
ser = serial_open(port=port_info, baudrate=115200, bytesize=DR_EIGHTBITS,
                  parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE)
```

8.1.9 통합 예제

serial 포트에서 RXD(2번pin) 와 TxD(3번pin) 결선한 후, 자체 loop-back 테스트, 하는 예제 입니다.

■ 예제 1: 자체 loop-back test 예제

```
# 시리얼 포트 OPEN
# D-SUB(9pin) 연결한 경우 : port="COM"
# USB to Serial 로 USB에 연결한 경우 : port="COM_USB"
ser = serial_open(port="COM_USB", baudrate=115200, bytesize=DR_EIGHTBITS,
parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE)
wait(1)

# SEND DATA : "123ABC"
res = serial_write(ser, b"123ABC") # b는 바이트 타입을 의미
wait(1)

# READ DATA
res, rx_data = serial_read(ser)
# H/W 적으로 RXD,TxD가 연결되어 있어 res=6(바이트) rx_data = b"123ABC" 가 수
신 됨

tp_popup("res ={0}, rx_data={1}".format(res, rx_data))

#해당 시리얼 포트를 닫음
serial_close(ser)
```

송신 데이터를 그대로 수신데이터로 받아서 결과를 TP pop-up 메시지로 출력합니다.
정상적으로 동작한 경우 res=6 rx_data = b'123ABC' 결과를 출력합니다.

- 예제 2: 다양한 패킷 전송 예제

송신 패킷 : "MEAS_START" +data1[4byte]+data2[4byte]

data1: integer를 4byte로 변환 ex) 1 → 00000001

data2: integer를 4byte로 변환 ex) 2 → 00000002

ex) data1=1, data2=2 인 경우: "MEAS_START"+00000001+00000002

실제 패킷: 4D4541535F53544152540000000100000002

수신 패킷 : res=18, rx_data="MEAS_START"+00000001+00000002

rx1 추출 : 10~14번째 byte를 interger로 변환

rx2 추출 : 14~18번째 byte를 interger로 변환

```

ser = serial_open(port="COM_USB", baudrate=115200, bytesize=DR_EIGHTBITS,
parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE)
wait(1)

send_data = b"MEAS_START" # b는 바이트 타입을 의미
data1 =1
data2 =2
send_data += (data1).to_bytes(4, byteorder='big')
send_data += (data2).to_bytes(4, byteorder='big')

# SEND DATA
res = serial_write(ser, send_data)
wait(1)

# READ DATA
# H/W 적으로 RXD,TXD가 연결되어 있어 send_data 가 그대로 수신 됨
res, rx_data = serial_read(ser)

tp_popup("res ={0}, rx_data={1}".format(res, rx_data))

rxd1 = int.from_bytes(rx_data[10:10+4], byteorder='big', signed=True)
rxd2 = int.from_bytes(rx_data[14:14+4], byteorder='big', signed=True)

```

```
tp_popup("res={0}, rxd1={1}, rxd2={2}".format(res, rxd1, rxd2))
```

```
#해당 시리얼 포트를 닫음  
serial_close(ser)
```

USB to serial 장비를 USB 포트에 연결하고 byte형 send_data 를 전송합니다.
송신 데이터를 그대로 수신하도록 RXD(2pin) 와 TXD(3pin)를 결선하였으므로
res = 18, rx_data는 send_data와 같은 패킷을 가집니다.
rxd1 추출 : 10~14번째 byte를 interger로 변환
rxd2 추출 : 14~18번째 byte를 interger로 변환
최종 결과는 res=18, rxd1=1, rxd2=2 가 됩니다.

8.2 Tcp/Client

8.2.1 client_socket_open(ip, port)

- 기능

로봇 제어기가 클라이언트 소켓을 생성하고, 서버(ip, port)에 연결을 시도합니다.
서버에 연결되면, 연결된 소켓을 반환합니다.

- 인수

인수명	자료형	기본값	설명
ip	str	-	Server IP address: (예) "192.168.137.200"
port	int	-	Server Port number (예) 20002

- 리턴

값	설명
socket.socket instance	연결 성공

- 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	연결 처리 중 socket.error 예외 발생

- 예제

```

sock = client_socket_open("192.168.137.200", 20002)
# 서버(ip="192.168.137.200", port=20002)에 무한 접속 시도를 합니다.
# 정상적으로 연결된 경우, 연결된 소켓을 반환합니다.
# 반환된 소켓(sock)을 이용하여 하기와 같이 데이터를 쓰고 읽고 닫습니다.

client_socket_write(sock, b"123abc")    #서버에 데이터전송 (b는 byte형을 의미)
res, rx_data = client_socket_read(sock) #서버로부터 데이터 수신
client_socket_close(sock)             #서버와의 연결 닫기

```

8.2.2 client_socket_close(sock)

- 기능

서버와의 통신을 종료합니다. 서버에 다시 접속하려면 반드시 client_socket_close(sock)로 소켓을 닫고, 재 open 해야 합니다.

- 인수

인수명	자료형	기본값	설명
sock	socket.socket	-	client_socket_open()에서 리턴 받은 socket instance

- 리턴

값	설명
0	연결 종료 성공

- 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수의 데이터형 오류 시
DR_Error (DR_ERROR_RUNTIME)	종료 처리 중 socket.error 예외 발생

- 예제

```

sock = client_socket_open("192.168.137.200", 20002)
# 서버(ip="192.168.137.200", port=20002)에 무한 접속 시도를 합니다.

# do something ...

client_socket_close(sock) #서버와의 통신을 종료합니다.

```

8.2.3 client_socket_state(sock)

- **기능**

통신 소켓의 연결 상태를 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
sock	socket.socket	-	client_socket_open()에서 리턴 받은 socket instance

- **리턴**

값	설명
1	connected 상태
0	disconnected 상태

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수의 데이터형 오류 시

- **예제**

```
sock = client_socket_open("192.168.137.200", 20002)
```

```
state = client_socket_state(sock) #소켓의 상태를 읽습니다.
```

```
client_socket_close(sock)
```

8.2.4 client_socket_write(sock, tx_data)

▪ 기능

서버에게 데이터를 송신합니다.

▪ 인수

인수명	자료형	기본값	설명
sock	socket.socket	-	client_socket_open()에서 리턴 받은 socket instance
tx_data	byte	-	송신할 데이터 • 데이터 타입이 byte 형이어야 합니다. • 하기 예제 참조 바랍니다.

▪ 리턴

값	설명
0	성공
-1	Server와 연결된 상태가 아닙니다.
-2	송신 처리 중 socket.error 예외 발생

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수의 데이터형 오류 시

▪ 예제

```

sock = client_socket_open("192.168.137.200", 20002)

client_socket_write(sock, b"1234abcd") #b 는 byte 형을 의미합니다.

# string 을 byte 형으로 변환
msg = "abcd"                      # msg는 string 변수
client_socket_write(sock, msg.encode()) # encode()는 string형을 byte형으로 변환

client_socket_close(sock)

```

8.2.5 client_socket_read(sock, length=-1, timeout=-1)

▪ 기능

서버로부터 데이터를 수신합니다.

▪ 인수

인수명	자료형	기본값	설명
sock	socket.socket	-	client_socket_open()에서 리턴 받은 socket instance
length	int	-1	수신할 데이터의 바이트 수 • -1 : 미지정 (수신된 데이터만큼 읽습니다.) • n(>=0) : 지정한 바이트 수만큼 읽습니다.
timeout	int float	-1	수신 대기 시간 • -1 : 무한 대기 • n(>0) : n 초(second)

▪ 리턴

값(res, rx_data)	설명	
res	>0	수신한 데이터의 바이트 수
	-1	Server와 연결 된 상태가 아닙니다.
	-2	수신 처리 중 socket.error 예외 발생
	-3	지정한 수신 timeout 발생
rx_data	수신한 데이터(byte 형)	

▪ 예외

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

- 예제

```
sock = client_socket_open("192.168.137.200", 20002)

res, rx_data = client_socket_read(sock)      # 데이터를 수신할때까지 무한 대기
# length 생략했으므로 수신된 데이터만큼 읽고,
# timeout 생략했으므로 수신될 때 까지 무한 대기 합니다.
# 데이터가 수신되면 (res = 수신 데이터 사이즈, rx_data=수신데이터) 반환됩니다.

res, rx_data = client_socket_read(sock, timeout=3) # 데이터 수신 될까지 3초간 대기
# 데이터가 3초안에 수신되면 (res = 수신 데이터 사이즈, rx_data=수신데이터) 반환
# 데이터가 3초안에 수신 안되면 (res = -3, rx_data=None) 반환

res, rx_data = client_socket_read(sock, length=64)    #수신 데이터 64바이트 읽기

res, rx_data = client_socket_read(sock, length=64, timeout=3)
# 3초 타임 아웃을 가지고 수신 데이터 64바이트 읽기

#수신된 byte형을 string 형으로 변환
rx_msg = rx_data.decode() #rx_data는 byte형이고 string형으로 변환하기 위해서는
                         #decode() 사용합니다.
                         #예를 들어, rx_data = b"abcd" 이면,
                         #rx_msg="abcd"가 됩니다.

client_socket_close(sock)
```

8.2.6 통합 예제

server IP = 192.168.137.200, open port =20002 이고,
수신한 패킷을 그대로 client 에게 전송하는 동작(mirroring)을 한다고 가정합니다.

- 예제 1: 기본 Tcp Client 예제

```
# server IP = 192.168.137,200, open port =20002 로 가정
g_sock = client_socket_open("192.168.137.200", 20002)

tp_popup("connect O.K!",DR_PM_MESSAGE)
while 1:
    client_socket_write(g_sock, b"abcd") # byte형으로 스트링 "abcd" 전송
    wait(0.1)
    res, rx_data = client_socket_read(g_sock) #server로부터 수신 대기
    tp_popup("res={0}, rx_data ={1}".format(res, rx_data), DR_PM_MESSAGE)
    wait(0.1)
```

server 에 접속하고 string "abcd"를 전송합니다. (b는 string을 byte형으로 변환합니다.)

server로 부터 수신된 메시지를 TP에 출력합니다.

server는 수신된 데이터를 그대로 송신하므로 res = 4, rx_data=b"abcd" 가 됩니다.

- 예제 2: 다양한 패킷 전송 예제

송신 패킷 : "MEAS_START " +data1[4byte]+data2[4byte]

data1: integer를 4byte로 변환 ex) 1 → 00000001

data2: integer를 4byte로 변환 ex) 2 → 00000002

ex) data1=1, data2=2 인 경우: "MEAS_START"+00000001+00000002

실제 패킷: 4D4541535F53544152540000000100000002

수신 패킷 : res=18, rx_data= "MEAS_START"+00000001+00000002

rx1 추출 : 10~14번째 byte를 interger로 변환

rx2 추출 : 14~18번째 byte를 interger로 변환

```
g_sock = client_socket_open("192.168.137.100", 20002)
```

```
tp_popup("connect O.K!",DR_PM_MESSAGE)
```

```
send_data = b"MEAS_START"
data1 =1
data2 =2
send_data += (data1).to_bytes(4, byteorder='big')
send_data += (data2).to_bytes(4, byteorder='big')

client_socket_write(g_sock, send_data)

wait(0.1)

res, rx_data = client_socket_read(g_sock)
tp_popup("res={0}, rx_data ={1}".format(res, rx_data), DR_PM_MESSAGE)

rxd1 = int.from_bytes(rx_data[10:10+4], byteorder='big', signed=True)
rxd2 = int.from_bytes(rx_data[14:14+4], byteorder='big', signed=True)

tp_popup("res={0}, rxd1={1}, rxd2={2}".format(res, rxd1, rxd2), DR_PM_MESSAGE)

client_socket_close(g_sock)
```

server에 접속하고 byte형 send_data를 전송합니다.

server는 수신된 데이터를 그대로 송신하므로 res = 18, rx_data는 send_data와 같은 패킷을 가집니다.

rxd1 추출 : 10~14번째 byte를 interger로 변환

rxd2 추출 : 14~18번째 byte를 interger로 변환

최종 결과는 res=18, rxd1=1, rxd2=2가 됩니다.

▪ 예제 3: 재 접속

```
def fn_reconnect():
    global g_sock
    client_socket_close(g_sock)
    g_sock = client_socket_open("192.168.137.200", 20002)
    return

g_sock = client_socket_open("192.168.137.200", 20002)
tp_popup("connect O.K!",DR_PM_MESSAGE)

client_socket_write(g_sock, b"abcd")
wait(0.1)

while 1:
    res, rx_data = client_socket_read(g_sock)
    if res < 0:
        fn_reconnect()
    else:
        tp_popup("res={0}, rx_data ={1}".format(res, rx_data), DR_PM_MESSAGE)
        wait(0.1)
```

client_socket_read() 명령어의 리턴값을 체크 합니다.

server와 접속이 끊어지거나, 통신 문제가 발생하면 음수 값이 리턴됩니다.

음수값이 리턴된 경우 reconnect() 함수를 호출하여 재접속을 시도합니다.

재 접속 시도 시, 기존에 open된 소켓을 닫는 것에 유의 바랍니다.

8.3 Tcp/Server

8.3.1 server_socket_open(port)

- 기능

로봇 제어기가 서버 소켓을 생성하고, client의 연결을 대기합니다. client가 연결되면, 연결된 소켓을 반환합니다.

- 인수

인수명	자료형	기본값	설명
port	int	-	오픈 할 port 번호

- 리턴

값	설명
socket.socket instance	연결 성공

- 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수의 데이터형 오류 시
DR_Error (DR_ERROR_VALUE)	인수의 값이 유효하지 않을 시
DR_Error (DR_ERROR_RUNTIME)	연결 처리 중 socket.error 예외 발생

- 예제

```

sock = server_socket_open(20002)
# 20002 port를 열고 client가 접속해 올 때까지 대기합니다.
# 정상적으로 연결된 경우, 연결된 소켓을 반환합니다.
# 반환된 소켓(sock)을 이용하여 하기와 같이 데이터를 쓰고 읽고 닫습니다.

server_socket_write(sock, b"123abc") #client에 데이터 전송 (b는 byte형을 의미)
res, rx_data = server_socket_read(sock) #client로부터 데이터 수신

server_socket_close(sock)           # client와의 연결 닫기

```

8.3.2 server_socket_close(sock)

- 기능

client 와의 통신을 종료합니다. client와 재 접속하려면 반드시 server_socket_close(sock)로 연결 종료 후, 재 open 해야합니다.

- 인수

인수명	자료형	기본값	설명
sock	socket.socket	-	server_socket_open()에서 리턴 받은 socket instance

- 리턴

값	설명
0	연결 종료 성공

- 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수의 데이터형 오류 시
DR_Error (DR_ERROR_RUNTIME)	종료 처리 중 socket.error 예외 발생

- 예제

```
sock = server_socket_open(20002)
# 20002 port를 열고 client가 접속해 올 때까지 대기합니다.
```

```
# do something ...
```

```
server_socket_close(sock) ) #client 와의 통신을 종료합니다.
```

8.3.3 server_socket_state(sock)

- **기능**

소켓의 연결 상태를 리턴합니다.

- **인수**

인수명	자료형	기본값	설명
sock	socket.socket	-	server_socket_open()에서 리턴 받은 socket instance

- **리턴**

값	설명
1	connected 상태
0	disconnected 상태

- **예외**

예외	설명
DR_Error (DR_ERROR_TYPE)	인수의 데이터형 오류 시

- **예제**

```

sock = server_socket_open(20002)

state = server_socket_state(sock) #소켓의 상태를 읽습니다.

server_socket_close(sock)

```

8.3.4 server_socket_write(sock, tx_data)

▪ 기능

Client에 데이터를 송신합니다.

▪ 인수

인수명	자료형	기본값	설명
sock	socket.socket	-	server_socket_open()에서 리턴 받은 socket instance
tx_data	byte	-	송신할 데이터 • 데이터 타입 byte 형이어야 합니다. • 하기 예제 참조 바랍니다.

▪ 리턴

값	설명
0	성공
-1	Client와 연결된 상태가 아닙니다.
-2	송신 처리 중 socket.error 예외 발생

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수의 데이터형 오류 시

▪ 예제

```

sock = server_socket_open(20002)

server_socket_write(sock, b"1234abcd") #b 는 byte 형을 의미합니다.

# string 을 byte 형으로 변환
msg = "abcd"                                # msg는 string 변수
server_socket_write(sock, msg.encode()) # encode()는 string형을 byte형으로 변환

server_socket_close(sock)

```

8.3.5 server_socket_read(sock, length=-1, timeout=-1)

▪ 기능

Client로부터 데이터를 수신합니다.

▪ 인수

인수명	자료형	기본값	설명
sock	socket.socket	-	server_socket_open()에서 리턴 받은 socket instance
length	int	-1	수신할 데이터의 바이트 수 • -1 : 미지정 (수신된 데이터만큼 읽습니다.) • n(>=0) : 지정한 바이트 수만큼 읽습니다.
timeout	int float	-1	수신 대기 시간 • -1 : 무한 대기 • n(>0) : n 초(second)

▪ 리턴

값(res, rx_data)		설명
res	0	수신한 데이터의 바이트 수
	-1	Client와 연결된 상태가 아닙니다.
	-2	수신 처리 중 socket.error 예외 발생
	-3	지정한 수신 timeout 발생
rx_data		수신한 데이터(byte 형)

- 예제

```
sock = server_socket_open(20002)

res, rx_data = server_socket_read(sock)      # 데이터를 수신할때까지 무한 대기
# length 생략했으므로 수신된 데이터만큼 읽고,
# timeout 생략했으므로 수신될 때 까지 무한 대기 합니다.
# 데이터가 수신되면 (res = 수신 데이터 사이즈, rx_data=수신데이터) 반환됩니다.

res, rx_data = server_socket_read(sock, timeout=3) #데이터 수신될까지 3초간 대기
# 데이터가 3초안에 수신되면 (res = 수신 데이터 사이즈, rx_data=수신데이터) 반환
# 데이터가 3초안에 수신 안되면 (res = -3, rx_data=None) 반환

res, rx_data = server_socket_read(sock, length=64) #수신 데이터 64바이트 읽기

res, rx_data = server_socket_read(sock, length=64, timeout=3)
# 3초 타임 아웃을 가지고 수신 데이터 64바이트 읽기

#수신된 byte형을 string 형으로 변환
rx_msg = rx_data.decode() #rx_data는 byte형이고 string형으로 변환하기 위해서는
#decode() 사용합니다.
#예를 들어, rx_data = b"abcd" 이면,
#rx_msg="abcd"가 됩니다.

server_socket_close(sock)
```

8.3.6 통합 예제

client는 제어기 IP = 192,168,137.100, port = 20002로 접속을 하고
수신한 패킷을 그대로 server에게 전송하는 동작(mirroring)을 한다고 가정합니다.

- 예제 1: 기본 Tcp Server 예제

```

g_sock = server_socket_open(20002)
tp_popup("connect O.K!",DR_PM_MESSAGE)

while 1:
    server_socket_write(g_sock, b"abcd") # byte형으로 스트링 "abcd" 전송
    wait(0.1)
    res, rx_data = server_socket_read(g_sock) #server로부터 수신 대기
    tp_popup("res={0}, rx_data ={1}".format(res, rx_data), DR_PM_MESSAGE)
    wait(0.1)

```

port = 20002를 열고 client가 접속할 때까지 기다립니다.
접속된 client에 접속하고 byte형 "abcd"를 전송합니다.
client로 부터 수신된 메시지를 TP에 출력합니다.
client는 수신된 데이터를 그대로 송신하므로 res = 4, rx_data=b"abcd" 가 됩니다.

- 예제 2: 다양한 패킷 전송 예제

송신 패킷 : "MEAS_START" +data1[4byte]+data2[4byte]
 data1: integer를 4byte로 변환 ex) 1 → 00000001
 data2: integer를 4byte로 변환 ex) 2 → 00000002
 ex) data1=1, data2=2 인 경우: "MEAS_START"+00000001+00000002
 실제 패킷: 4D4541535F53544152540000000100000002

수신 패킷 : res=18, rx_data= "MEAS_START"+00000001+00000002
 rxd1 추출 : 10~14번째 byte를 interger로 변환
 rxd2 추출 : 14~18번째 byte를 interger로 변환

```

g_sock = server_socket_open(20002)
tp_popup("connect O.K!",DR_PM_MESSAGE)

```

```
send_data = b"MEAS_START"
data1 =1
data2 =2
send_data += (data1).to_bytes(4, byteorder='big')
send_data += (data2).to_bytes(4, byteorder='big')

server_socket_write(g_sock, send_data)

wait(0.1)

res, rx_data = server_socket_read(g_sock)
tp_popup("res={0}, rx_data ={1}".format(res, rx_data), DR_PM_MESSAGE)

rxd1 = int.from_bytes(rx_data[10:10+4], byteorder='big', signed=True)
rxd2 = int.from_bytes(rx_data[14:14+4], byteorder='big', signed=True)

tp_popup("res={0}, rxd1={1}, rxd2={2}".format(res, rxd1, rxd2), DR_PM_MESSAGE)

server_socket_close(g_sock)
```

client에 byte형 send_data 를 전송합니다.

client는 수신된 데이터를 그대로 송신하므로 res = 18, rx_data는 send_data와 같은 패킷을 가집니다.

rxd1 추출 : 10~14번째 byte를 interger로 변환

rxd2 추출 : 14~18번째 byte를 interger로 변환

최종 결과는 res=18, rxd1=1, rxd2=2 가 됩니다.

- 예제 3: 재 접속

```
def fn_reopen():
    global g_sock
    server_socket_close(g_sock)
    g_sock = server_socket_open(20002)
    return

g_sock = server_socket_open(20002)
tp_popup("connect O.K!",DR_PM_MESSAGE)

server_socket_write(g_sock, b"abcd")
wait(0.1)

while 1:
    res, rx_data = server_socket_read(g_sock)
    if res < 0:
        fn_reopen()
    else:
        tp_popup("res={0}, rx_data ={1}".format(res, rx_data), DR_PM_MESSAGE)
        wait(0.1)
```

server_socket_read() 명령어의 리턴값을 체크 합니다.

client 와 접속이 끊어지거나, 통신 문제가 발생하면 음수 값이 리턴됩니다.

음수값이 리턴된 경우 reopen() 함수를 호출하여 client 접속을 기다립니다.

재 접속 open 시, 기존에 open된 소켓을 닫는 것에 유의 바랍니다.

8.4 Modbus

8.4.1 add_modbus_signal (ip, port, name, reg_type, index, value=0, slaveid=255)

▪ 기능

ModbusTCP의 신호를 등록합니다. Modbus I/O 설정의 경우 티치팬던트 I/O set-up 메뉴에서 설정 해야하지만 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.)
slaveid	int	255	• Slave ID of the ModbusTCP module (0 or 1-247 or 255) 0 : Broadcast address 255 : Default value for ModbusTCP

■ 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)
```

8.4.2 add_modbus_rtu_signal (slaveid=1, port=None, baudrate=115200, bytesize=DR_EIGHTBITS, parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE, name, reg_type, index, value=0)

▪ 기능

ModbusRTU의 신호를 등록합니다. Modbus I/O 설정의 경우 티치팬던트 I/O set-up 메뉴에서 설정 해야하지만 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
slaveid	int	1	<ul style="list-style-type: none"> ModbusRTU의 Slave ID입력(0 또는 1-247) 0 : Broadcast address
port	string	None	<ul style="list-style-type: none"> D-SUB(9 pin) 연결 : "COM" USB to Serial 연결 : "COM_USB"
baudrate	int	115200	Baud rate 9600, 19200, 57600, 115200, etc
bytesize	int	8	데이터 bit 수 <ul style="list-style-type: none"> DR_FIVEBITS: 5 DR_SIXBITS: 6 DR_SEVENBITS: 7 DR_EIGHTBITS: 8
parity	string	"N"	Parity checking <ul style="list-style-type: none"> DR_PARITY_NONE: "N" DR_PARITY EVEN: "E" DR_PARITY ODD: "O"
stopbits	int	1	Stop bit의 수 <ul style="list-style-type: none"> DR_STOPBITS_ONE =1 DR_STOPBITS_TWO = 2
name	string	-	modbus signal 이름
reg_type	int	-	Modbus의 신호 타입 <ul style="list-style-type: none"> DR_MODBUS_DIG_INPUT DR_MODBUS_DIG_OUTPUT DR_MODBUS_REG_INPUT

인수명	자료형	기본값	설명
			• 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

```
add_modbus_rtu_signal(slaveid=1, port=port_info, baudrate=115200,
bytesize=DR_EIGHTBITS, parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE,
name='di1', reg_type=DR_MODBUS_REG_INPUT, index=0)
add_modbus_rtu_signal(slaveid=1, port=port_info, baudrate=115200,
bytesize=DR_EIGHTBITS, parity=DR_PARITY_NONE, stopbits=DR_STOPBITS_ONE,
name='do1', reg_type=DR_MODBUS_REG_OUTPUT, index=0, value=12345)
```

8.4.3 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
```

8.4.4 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)
```

8.4.5 set_modbus_outputs(iobus_list, val_list)

- 기능

Modbus Slave장치에 복수 개의 신호를 내보내기 위한 명령입니다

- 인수

인수명	자료형	기본값	설명
iobus	string	-	modbus 이름(TP에서 설정)
value	int	-	I/O 출력 값 list

- 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/O 출력 접점 "d1"에 OFF, "d2"에 ON, "d3"에 ON
set_modbus_outputs(iobus_list=[ "d1", "d2", "d3"], val_list=[0,1,1])
```

```
# Modbus digital I/O 출력 접점 "d3"에 OFF, "d4"에 ON
set_modbus_outputs(["d3", "d4"], [0,1])
```

8.4.6 get_modbus_input(iobus)

- **기능**

Modbus Slave 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")
```

8.4.7 get_modbus_inputs(iobus_list)

- 기능

Modbus Slave장치에 복수 개의 신호를 읽어오기 위한 명령입니다.

- 인수

인수명	자료형	기본값	설명
iobus_list	list(string)	-	modbus input 이름 list (TP에서 설정) signal type이 아래의 경우에만 사용가능 • DR_MODBUS_DIG_INPUT • DR_MODBUS_DIG_OUTPUT

- 리턴

값	설명
int (>=0)	한번에 읽은 복수 개의 신호 값 (iobus_list 의 첫번째 값=LSB, iobus_list 의 마지막 값=MSB 가 되는 비트 조합의 값)
음수 값	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/O 입력 신호가 "di1"은 OFF, "di2"은 ON, "di3"은 ON 인 경우
res = get_modbus_inputs(iobus_list=["di1", "di2", "di3"])
#res 기대값 = 0b110(이진수), 6(십진수), 0x06(16진수)
```

```
# Modbus digital I/O 입력 신호가 "di4"은 OFF, "di5"은 ON 인 경우
```

```
res = get_modbus_inputs(["di4", "di5"])
#res 기대값 = 0b10(이진수), 2(십진수), 0x02(16진수)
```

8.4.8 get_modbus_inputs_list(iobus_list)

- **기능**

Modbus Slave 장치의 Register Type의 복수 개 신호를 읽어오기 위한 명령어입니다.

- **인수**

인수명	자료형	기본값	설명
iobus_list	list(string)	-	modbus input 이름 list (TP에서 설정) signal type이 아래의 경우에만 사용가능 • DR_MODBUS_REG_INPUT • DR_MODBUS_REG_OUTPUT

- **리턴**

값	설명
res	읽은 값의 갯수
val_list	한번에 읽은 복수 개의 신호 값의 list

- **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 Register I/O 입력 신호가 "Holding1"은 1234, "Input1"은 567, "Holding2"
은 9876 인 경우
res, val_list = get_modbus_inputs_list(iobus_list=[ "Holding1", "Input1", "Holding2"])
#res 기대값 = 3
#val_list 기대값 = [1234, 567, 9876]
```

8.4.9 wait_modbus_input(iobus, val, timeout=None)

▪ 기능

Modbus Slave 장치에서 지정한 신호값이 val(ON or OFF)이 될 때까지 대기합니다. 대기 시간은 timeout 설정으로 변경할 수 있으며, 지정된 시간이 지나면 대기 상태가 종료됨과 동시에 결과를 리턴합니다. 단, timeout을 설정하지 않으면 무한 대기합니다.

▪ 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
timeout	float	-	대기 시간 [sec] 설정하지 않으면 무한 대기

▪ 리턴

값	설명
0	성공
-1	실패 (time-out)

▪ 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

```
wait_modbus_input("CIN0", ON)  # "CIN0" 신호가 ON될 때까지 무한 대기
wait_modbus_input("CIN0", OFF)  # "CIN0" 신호가 OFF될 때까지 무한 대기
```

```
res = wait_modbus_input("CIN0", ON, 3) # "CIN0" 신호가 ON될 때까지 3초간 대기  
# 3초 안에 1번 신호가 ON 되면, res = 0  
# 3초 안에 1번 신호가 ON 되지 않았으면 res = -1
```

8.4.10 set_modbus_slave(address, val)

- **기능**

ModbusTCP Slave의 Gerneral Purpose Register 영역에 값을 내보내기 위해 사용합니다.

- **인수**

인수명	자료형	기본값	설명
address	int	-	GPR영역의 주소값 (128~255)
val	int	-	2byte 값 (0~65535)

- **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_modbus_slave(128, 0)
set_modbus_slave(255, 65535)
```

8.4.11 get_modbus_slave(address)

- 기능

ModbusTCP Slave의 Gerneral Purpose Register 영역을 접근하여 값을 가져오기 위해 사용합니다.

- 인수

인수명	자료형	기본값	설명
address	int	-	읽으려고 하는 GPR영역의 주소값 (128~255)

- 리턴

값	설명
value	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

```
value1 = get_modbus_slave(128)
value2 = get_modbus_slave(255)
```

8.4.12 modbus_crc16(data)

- 기능

Modbus protocol 사용 시, modbus crc16 계산에 대한 부하를 줄이기 위한 명령어입니다.

- 인수

인수명	자료형	기본값	설명
data	byte	-	crc16계산을 위한 modbus data

- 리턴

값	설명
crchigh	Crc16 계산 결과의 high byte
crclow	Crc16 계산 결과의 low byte

- 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

```
data = b"\x01\x02\x03\x04\x05\x06"
crchigh, crclow = modbus_crc16(data)
#crchigh = 186(DEC), BA(HEX)
#crclow = 221(DEC), DD(HEX)
```

8.4.13 modbus_send_make(send_data)

- 기능

Modbus protocol 사용 시, send data에 대하여 modbus crc16 결과값을 포함하는 결과 data를 제공하기 위한 명령어입니다.

- 인수

인수명	자료형	기본값	설명
send_data	byte	-	crc 계산이 필요한 전송 데이터

- 리턴

값	설명
result_data	send data에 modbus crc16 결과값이 포함된 결과 data

- 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

```
senddata = b"\x01\x02\x03\x04\x05\x06"
resultdata = modbus_send_make(senddata)
#resultdata = b'\x01\x02\x03\x04\x05\x06\xba\xdd'
```

8.4.14 modbus_recv_check(recv_data)

- **기능**

Modbus protocol 사용 시, receive data에 대하여 crc16 값을 이용한 데이터 무결성 체크를 하기 위한 명령어입니다.

- **인수**

인수명	자료형	기본값	설명
recv_data	byte	-	raw modbus data

- **리턴**

값	설명
res	True/False

- **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**

```
#recvdata = b"\x01\x02\x03\x04\x05\x06\xba\xdd"
res = modbus_recv_check(recvdata)
#recv = True
```

8.5 Industrial Ethernet (EtherNet/IP,PROFINET)

8.5.1 set_output_register_bit(address, val)

- **기능**

Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Output영역의 Bit General Purpose Register의 값을 설정하기 위한 명령어입니다.

- **인수**

인수명	자료형	기본값	설명
address	unsigned short	-	Industrial Ethernet Slave의 Output Bit GPR영역 주소값 (0-63)
val	int	-	ON : 1 OFF : 0

- **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_output_register_bit (0, ON)
set_output_register_bit (63, OFF)
```

8.5.2 set_output_register_int(address, val)

■ 기능

Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Output 영역의 Int General Purpose Register의 값을 설정하기 위한 명령어입니다.

■ 인수

인수명	자료형	기본값	설명
address	unsigned short	-	Industrial Ethernet Slave의 Output Int GPR 영역 주소값 (0-23)
val	int	-	int value (4byte)

■ 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_output_register_int (0, 0x00FF00FF)
set_output_register_int (23, 65535)
```

8.5.3 set_output_register_float(address, val)

■ 기능

Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Output영역의 Float General Purpose Register의 값을 설정하기 위한 명령어입니다.

■ 인수

인수명	자료형	기본값	설명
address	unsigned short	-	Industrial Ethernet Slave의 Output Float GPR영역 주소값 (0-23)
val	float	-	float value (4byte)

■ 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_output_register_float (0, 4.5)
set_output_register_float (23, 2.3)
```

8.5.4 get_output_register_bit(address)

- 기능

Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Output영역의 Bit General Purpose Register의 값을 읽어오기 위한 명령어입니다.

- 인수

인수명	자료형	기본값	설명
address	unsigned short	-	Industrial Ethernet Slave의 Output Bit GPR영역 주소값 (0-63)

- 리턴

값	설명
value	해당 GPR 주소의 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

```
a = get_output_register_bit (0)
b = get_output_register_bit (63)
```

8.5.5 get_output_register_int(address)

- 기능

Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Output 영역의 Int General Purpose Register의 값을 읽어오기 위한 명령어입니다.

- 인수

인수명	자료형	기본값	설명
address	unsigned short	-	Industrial Ethernet Slave의 Output Int GPR 영역 주소값 (0-23)

- 리턴

값	설명
value	해당 GPR 주소의 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

```
a = get_output_register_int (0)
b = get_output_register_int(23)
```

8.5.6 get_output_register_float(address)

■ 기능

Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Output영역의 Float General Purpose Register의 값을 읽어오기 위한 명령어입니다.

■ 인수

인수명	자료형	기본값	설명
address	unsigned short	-	Industrial Ethernet Slave의 Output Float GPR영역 주소값 (0-23)

■ 리턴

값	설명
value	해당 GPR 주소의 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

```
a = get_output_register_float (0)
b = get_output_register_float (63)
```

8.5.7 get_input_register_bit(address)

- **기능**

Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Input영역의 Bit General Purpose Register의 값을 읽어오기 위한 명령어입니다.

- **인수**

인수명	자료형	기본값	설명
address	unsigned short	-	Industrial Ethernet Slave의 Input Bit GPR영역 주소값 (0-63)

- **리턴**

값	설명
value	해당 GPR 주소의 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**

```
a = get_input_register_bit (0)
b = get_input_register_bit (63)
```

8.5.8 get_input_register_int(address)

■ 기능

Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Input영역의 Int General Purpose Register의 값을 읽어오기 위한 명령어입니다.

■ 인수

인수명	자료형	기본값	설명
address	unsigned short	-	Industrial Ethernet Slave의 Input Int GPR영역 주소값 (0-63)

■ 리턴

값	설명
value	해당 GPR 주소의 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

```
a = get_input_register_int (0)
b = get_input_register_int (23)
```

8.5.9 get_input_register_float(address)

- 기능

Industrial Ethernet(EtherNet/IP, PROFINET) Slave의 Input영역의 Float General Purpose Register의 값을 읽어오기 위한 명령어입니다.

- 인수

인수명	자료형	기본값	설명
address	unsigned short	-	Industrial Ethernet Slave의 Input Float GPR영역 주소값 (0-23)

- 리턴

값	설명
value	해당 GPR 주소의 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

```
a = get_input_register_float (0)
b = get_input_register_float (63)
```

9. 외부 비전 명령어

개요

임의의 외부 비전 시스템을 당사 로봇과 연결하여 작업(Vision guided robotics) 하기 위한 명령어를 제공합니다. 평면 상에 위치하는 물체의 2차원 변위 정보(위치 Tx, Ty, 회전 Rz)를 측정하는 2D 비전 시스템에 연결이 가능하며, 다수 물체에 대한 측정정보 입력도 가능하도록 명령어가 구성되어 있습니다. 2D 비전 시스템을 사용하는 비전-가이드 로봇의 경우 작업대상 물체에 대하여 정의된 로봇작업 좌표를 기준 이미지와 측정이미지 간의 비전측정좌표 변위 정보를 이용하여 로봇작업 좌표를 산출합니다. 2D 비전 시스템을 이용한 로봇 애플리케이션의 설치 및 작업은 비전시스템의 이미지 좌표계를 로봇시스템의 물리적 좌표계로 캘리브레이션(좌표계 보정) 시키는 것이 필요하며, 외부 비전시스템을 사용할 때에는 비전 자체적으로 좌표계 캘리브레이션 작업을 수행한 후 보정된 좌표정보를 로봇에 전송해야 합니다.

비전시스템의 설치는 로봇과 연결된 Eye-in-hand 방식과 로봇과 분리된 in-line 방식으로 모두 설치할 수 있으며, 작업하는 상황에서 로봇과 비전시스템의 위치관계가 변하지 않도록 고정되어 있어야 합니다. 비전시스템과 로봇제어기는 TCP/IP 방식으로 통신하며, 로봇제어기의 유선공유기 포트에 비전시스템 케이블을 연결해주어야 통신이 가능합니다.

9.1 **vs_set_info(type)**

- **기능**

사용할 비전시스템의 종류를 설정한다.

- **인수**

인수명	자료형	기본값	설명
type	int	DR_VS_CUSTOM	DR_VS_CUSTOM(0) DR_VS_COGNEX(1) DR_VS_SICK(2)

- **리턴**

값	설명
type의 ID	설정하고자 하는 type의 ID

- **예제**

```
vs_set_info(DR_VS_COGNEX) #Vision type information setting  
vs_connect("192.168.137.10") #Connect to vision - Vision IP, Default port  
# Enter your task  
vs_disconnect() #Disconnect to vision
```

*지원 모델

Type	Model
DR_VS_COGNEX	COGNEX IS2000M-23M Series COGNEX IS5600/5705 Series COGNEX IS7000Series COGNEX IS8000Series
DR_VS_SICK	SICK Inspector PIM60 Series SICK Inspector PI50 Series
DR_VS_CUSTOM	

9.2 vs_connect(ip_addr, port_num=9999)

▪ 기능

비전시스템의 통신을 연결한다.

▪ 인수

인수명	자료형	기본값	설명
ip_addr	str	-	비전 모듈의 Server IP (예, 192.168.137.200)
port_num	int	9999	Port 번호 (예, 9999)

▪ 리턴

값	설명
0	연결 성공
-1	연결 실패

▪ 예제

```

vs_set_info(DR_VS_COGNEX) #Vision type information setting
vs_connect("192.168.137.10") #Connect to vision - Vision IP, Default port
# Enter your task
vs_disconnect()           #Disconnect to vision

```

9.3 **vs_disconnect()**

- **기능**

비전시스템의 통신을 해제한다.

- **리턴**

없음	
----	--

- **예제**

```
vs_set_info(DR_VS_COGNEX) #Vision type information setting  
vs_connect("192.168.137.10") #Connect to vision - Vision IP, Default port  
# Enter your task  
vs_disconnect()                  #Disconnect to vision
```

9.4 vs_get_job()

- **기능**

비전시스템에 현재 로딩되어 있는 작업명을 불러온다. (*VS_TYPE: DR_VS_COGNEX, DR_VS_SICK)

- **리턴**

값	자료형	설명
job_name	string	연결 성공
-1	int	연결 실패

- **예제**

```

vs_set_info(DR_VS_COGNEX)    #Vision type information setting
vs_connect("192.168.137.10") #Connect to vision - Vision IP, Default port

vs_set_job("test.job")        # Set (load) the current vision job
job_name=vs_get_job()         # Get the current setting vision job
tp_popup("{0}.format(job_name))

vs_disconnect()               # Disconnect to vision

```

9.5 **vs_set_job(job_name)**

▪ 기능

입력된 작업을 비전시스템에 로딩한다. (*VS_TYPE: DR_VS_COGNEX, DR_VS_SICK)

▪ 인수

인수명	자료형	기본값	설명
job_name	string		로딩할 작업명

▪ 리턴

값	자료형	설명
0	int	성공
-1	int	실패

▪ 예제

```
vs_set_info(DR_VS_COGNEX) #Vision type information setting  
vs_connect("192.168.137.10") #Connect to vision - Vision IP, Default port  
  
vs_set_job("test.job") # Set (load) the current vision job  
job_name=vs_get_job() # Get the current setting vision job  
tp_popup("{0}.format(job_name))  
  
vs_disconnect() # Disconnect to vision
```

9.6 vs_trigger()

▪ 기능

비전시스템에 측정명령을 전달한다. 측정이 성공하면, 측정결과값을 리턴한다.
(*VS_TYPE: DR_VS_COGNEX, DR_VS_SICK)

▪ 리턴

값	자료형	설명
pos	posx	pos: 대상물체의 측정 위치 (posx 변수 타입)
var_list	list[float]	var_list: 사용자가 추가로 입력한 측정결과값 예) 검사합불, 거리측정, 각도측정값 등
-1, []	int	실패

▪ 예제

```

vs_set_info(DR_VS_COGNEX)           # Select type of vision sensor
vs_connect("192.168.137.10")        # Vision IP, Default port

vis_posx = posx (410,310,300,0,0,0) # Define the initial posx data - vision
rob_posx = posx (400,300,300,0,180,0) # Define the initial posx data - robot

vs_set_init_pos(vis_posx, rob_posx, VS_POS1) # Enter the initial posx data to Vision

for i in range(10):
    pos, var_list = vs_trigger()          # Execute the vision meausrement
    if var_list[0] == 1:                  # Check the inspection result
        robot_posx_meas = vs_get_offset_pos(pos, VS_POS1) # offset the robot pose
        movev(robot_posx_meas)            # move the robot pose
    else:
        tp_popup("Inspection Fail")

vs_disconnect()

```

9.7 **vs_set_init_pos(vision_posx_init, robot_posx_init, vs_pos=1)**

▪ 기능

비전 가이던스 작업을 수행할 대상물의 초기 위치정보를 입력한다.

(*VS_TYPE: DR_VS_COGNEX, DR_VS_SICK)

▪ 인수

인수명	자료형	기본값	설명
vision_posx_init	posx	-	비전측정 좌표 초기값
robot_posx_init	posx	-	로봇작업 좌표 초기값
vs_pos	int	1	입력한 초기값의 pos 번호

▪ 리턴

값	자료형	설명
vs_pos	int	성공 – 입력된 pos 번호
-1	int	실패

▪ 예제

```

vs_set_info(DR_VS_COGNEX)           # Select type of vision sensor
vs_connect("192.168.137.10")        # Vision IP, Default port
vis_posx = posx (410,310,300,0,0,0) # Define the initial posx data - vision
rob_posx = posx (400,300,300,0,180,0) # Define the initial posx data - robot
vs_set_init_pos(vis_posx, rob_posx, VS_POS1) # Enter the initial posx data to Vision
for i in range(10):
    pos, var_list = vs_trigger()      # Execute the vision meausrement
    if var_list[0] == 1:               # Check the inspection result
        robot_posx_meas = vs_get_offset_pos(pos, VS_POS1) # offset the robot pose
        movel(robot_posx_meas)         # move the robot pose
    else:
        tp_popup("Inspection Fail")
vs_disconnect()

```

9.8 vs_get_offset_pos(vision_posx_meas, vs_pos=1)

▪ 기능

비전 시스템에서 측정된 좌표값을 이용하여 옵셋(가이드)된 로봇작업 좌표를 산출한다.
사전에 vs_set_init_pos를 통해 초기값이 입력되어 있어야 한다.

▪ 인수

인수명	자료형	기본값	설명
vision_posx_meas	posx	-	vs_trigger를 이용하여 얻어진 비전측정 좌표값
vs_pos	int	1	옵셋 좌표를 계산할 로봇 초기값 pos 번호

▪ 리턴

값	자료형	설명
robot_posx_meas	posx	성공
-1	int	실패

▪ 예제

```

vs_set_info(DR_VS_COGNEX)           # Select type of vision sensor
vs_connect("192.168.137.10")        # Vision IP, Default port

vis_posx = posx (410,310,300,0,0,0) # Define the initial posx data - vision
rob_posx = posx (400,300,300,0,180,0) # Define the initial posx data - robot

vs_set_init_pos(vis_posx, rob_posx, VS_POS1) # Enter the initial posx data to Vision

for i in range(10):
    pos, var_list = vs_trigger()          # Execute the vision measurement
    if var_list[0] == 1:                  # Check the inspection result
        robot_posx_meas = vs_get_offset_pos(pos, VS_POS1) # offset the robot pose
        movel(robot_posx_meas)            # move the robot pose
    else:
        tp_popup("Inspection Fail")

vs_disconnect()

```

9.9 **vs_request(cmd)**

▪ 기능

Vision System에 요청할 기능을 설정합니다.

(*VS_TYPE: DR_VS_CUSTOM)

▪ 인수

인수명	자료형	기본값	설명
cmd	int	-	비전시스템에서 검출하고자 하는 물체 개수

▪ 리턴

값	설명
0	성공
-1	실패
-2	통신 타임 아웃(3초) 발생

▪ 예제

```
vs_request(1)                           # request the vision measurement on the "1" job
```

9.10 vs_result()

▪ 기능

Vision System의 처리 결과를 가져옵니다.

(*VS_TYPE: DR_VS_CUSTOM)

▪ 리턴

값	설명
cnt (>=1)	성공 비전시스템에서 검출된 물체 개수
result	비전 결과 위치 리스트 (x좌표, y좌표, 회전값)
-	실패 시, cnt =-2 res = 빈 리스트

▪ 예제

```
cnt , result = vs_result()      # get the vision measurement result
for i in range(cnt):           # cnt : the count of the measured items
    x = result[i][0]            # result = [x1,y1,t1, x2,y2,t2,...]
    y = result[i][1]
    t = result[i][2]
```

9.11 통합예제 (DR_VS_COGNEX, DR_VS_SICK)

▪ 예제

```

vs_set_info(DR_VS_COGNEX)           # Select type of vision sensor
if ( vs_connect("192.168.137.10") != 0 ):    # Vision IP, Default port
    tp_popup("connection fail",DR_PM_MESSAGE)
    exit()

vis_posx_init = posx (410,310,300,0,0,0)      # Define the initial posx data - vision
rob_posx_init1 = posx (400,300,300,0,180,0)   # Define the initial posx data - robot
rob_posx_init2 = posx (420,320,300,0,180,0)   # Define the initial posx data - robot

vs_set_init_pos(vis_posx_init, rob_posx_init1, VS_POS1) # Enter the initial posx data to Vision
vs_set_init_pos(vis_posx_init, rob_posx_init2, VS_POS2)

for i in range(10):
    pos_meas, var_list = vs_trigger()          # Execute the vision measurement
    if pos_meas == -1:                         # Vision Fail to measure the object
        tp_popup("Vision measure fail")
        continue
    if var_list[0] == 1:                        # Check the inspection result
        # Get guided posx data
        rob_posx1_meas = vs_get_offset_pos(pos_meas, VS_POS1) # offset the robot pose
        rob_posx2_meas = vs_get_offset_pos(pos_meas, VS_POS2) # offset the robot pose
        movel(rob_posx1_meas)
        movel(rob_posx2_meas)

    else:
        tp_popup("Inspection Fail")
        continue

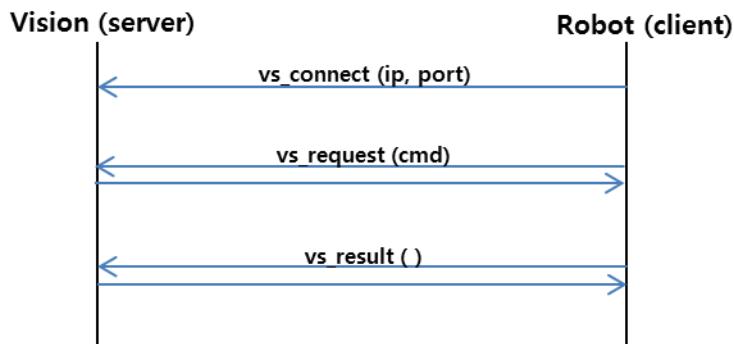
vs_disconnect()

```

9.12 통합예제 (DR_VS_CUSTOM)

- 통신프로토콜

비전 시스템에서 하기 프로토콜을 준수해야만, 비전 명령어가 정상적으로 동작합니다.



vs_request (cmd)

1) 로봇 제어기 → 비전 시스템

"MEAS_START" + cmd[4byte]

- cmd는 검출 물체 개수: integer를 4byte로 변환 ex) cmd=1 → 00000001

ex) cmd= 1 인 경우: "MEAS_START"+**00000001**

실제 패킷: 4D4541535F5354415254**00000001**

2) 비전 시스템 → 로봇 제어기

비전 시스템이 정상인 경우 "MEAS_OK" 비 정상인 경우 "MEAS_NG"를 전송

vs_result ()

1) 로봇 제어기 → 비전 시스템

"MEAS_REQUEST"

2) 비전 시스템 → 로봇 제어기

"MEAS_INFO" + cnt[4byte] + [(x[4byte] + y[4byte] + t[4byte]) × cnt개수]

- cnt 는 검출된 물체의 개수를 의미 함.

- x(x좌표), y(y좌표), t(회전값)는 100배 스케일링한 값을 전송해야 함.

ex) cnt = 1, (x=1.1, y=2.2, t=3.3)

"MEAS_INFO"+**1[4byte]** +110[4byte] +220[4byte] +330[4byte]

실제 패킷: 4D4541535F494E464F**00000001**0000006E**000000DC**0000014A

ex) cnt = 2, (x=1.1, y=2.2, t=3.3) (**x=1.1**, y=-2.2, t=-3.3)

"MEAS_INFO"+**2[4byte]** +110[4byte] +220[4byte] +330[4byte]

+110[4byte] -220[4byte] -330[4byte]

실제 패킷: 4D4541535F494E464F**00000002**0000006E**000000DC**0000014A

0000006E**FFFFFFFFFF24**FFFFFEB6

■ 예제

```
vs_set_info(DR_VS_CUSTOM)
res = vs_connect("192.168.137.200", 9999)      #비전과 통신 연결 시도
if res !=0:                                     #통신 연결 결과 확인
    tp_popup("connection fail",DR_PM_MESSAGE)   #접속 실패 시, 프로그램 종료
    exit()

ret = vs_request(1)                            #1번 물체 비전측정 정보 요청

cnt, result = vs_result()                     #물체 측정결과 정보 가져 오기

for i in range(cnt):
    x = result[i][0]
    y = result[i][1]
    t = result[i][2]
    tp_popup("x={0},y={1}, t={2}".format(result[i][0], result[i][1],
result[i][2]),DR_PM_MESSAGE)
```

10. 두산비전(SVM) 명령어

10.1 svm_connect(ip="192.168.137.5", port=20)

- 기능

SVM의 통신을 연결한다.

- 인수

인수명	자료형	기본값	설명
ip	str	"192.168.137.5"	SVM Server IP address
port	int	20	Port 번호

- 리턴

값	설명
0	연결 성공
-1	연결 실패

- 예제

```
svm_connect()      #Connect to vision - Default IP address and port number
# Enter the vision task
svm_disconnect()   #Disconnect to vision
```

10.2 svm_disconnect()

- **기능**

SVM의 통신을 해제한다.

- **예제**

```
svm_connect() #Connect to vision – Default IP address and port  
# Enter the vision task  
svm_disconnect() #Disconnect to vision
```

10.3 svm_set_job(job_id)

- **기능**

입력된 id에 해당하는 비전작업을 SVM에 로딩한다.

- **인수**

인수명	자료형	기본값	설명
job_id	int	-	비전작업 id (예. 1000, 2000, ...)

- **리턴**

값	설명
0	작업 로딩 성공
-1	작업 로딩 실패

- **예제**

```

svm_connect()           #Connect to vision - Vision IP, Default port
vision_test=1000        # Define vision job ID
svm_set_job(vision_test) # Load the vision_test (1000)
svm_disconnect()        #Disconnect to vision

```

svm_get_robot_pose(job_id)

10.4 svm_get_robot_pose(job_id)

▪ 기능

입력된 비전작업에 설정되어 있는 로봇자세정보(조인트 좌표계)를 불러온다. 로봇 자세정보는 비전작업 수행을 위한 shoot_pose로 활용된다.

▪ 인수

인수명	자료형	기본값	설명
job_id	int	-	비전작업 id (예. 1000, 2000, ...)

▪ 리턴

값	설명
float [6]	로봇의 조인트 좌표 정보 (posj type)
-1	실패

▪ 예제

```
svm_connect()                                # Connect to vision
vision_test=1000                            # Define vision job ID
svm_set_job(vision_test)                      # Load the vision_test (1000)
shoot_pos=svm_get_robot_pose (vision_test)    # Load the robot pose of vision_test
tp_popup("{0}".format(shoot_pos))
svm_disconnect()                             # Disconnect to vision
```

10.5 svm_get_vision_info(job_id)

▪ 기능

입력된 비전작업에 해당하는 측정명령을 수행한다. WCM(워크셀 매니저)를 통하여 비전작업의 측정명령 상세정보를 미리 입력하여야 한다.

▪ 인수

인수명	자료형	기본값	설명
job_id	int	-	비전작업 id (예. 1000, 2000, ...)

▪ 리턴

값	설명
1	측정 성공 - 1개의 물체를 검출/측정에 성공했다는 의미
0	측정 실패 - 해당하는 비전작업 물체를 검출하지 못함.
-1	측정 실패 - 통신 오류 발생 (timeout)

▪ 예제

```

svm_connect()                                # Connect to vision
vision_test=1000                            # Define vision job ID
svm_set_job(vision_test)                     # Load the vision_test (1000)
shoot_pos=svm_get_robot_pose (vision_test)    # Load the robot pose of vision_test
count=svm_get_vision_info(vision_test)         # Execute the vision measurement
tp_popup("{0}".format(count))                # Check the result
svm_disconnect()                            # Disconnect to vision

```

svm_get_variable(tool_id, var_type)

10.6 svm_get_variable(tool_id, var_type)

▪ 기능

svm_get_vision_info를 수행하여 물체검출/측정에 성공(1)한 경우, 검출/측정 데이터를 불러온다. 불러온 데이터에 대한 툴(tool) id와 데이터 형식 (variable type)을 입력한다.

- Position tool: POSX_TYPE (물체위치), VALUE_TYPE (검출 유사도)
- Presence tool: INSP_TYPE (유무 검사결과), VALUE_TYPE (픽셀 카운트)
- Distance tool: INSP_TYPE (거리측정 검사결과), VALUE_TYPE (거리측정값)
- Angle tool: INSP_TYPE (각도측정 검사결과), VALUE_TYPE (각도측정값)
- Diameter tool: INSP_TYPE (직경측정 검사결과), VALUE_TYPE (직경측정값),
POSX_TYPE (원 중심 위치)

▪ 인수

인수명	자료형	기본값	설명
tool_id	int	-	비전툴 id (예. 1000, 1001, 1002, ...)
var_type	int	-	POSX_TYPE: 좌표정보 변수 (posx) INSP_TYPE: 검사결과 변수 (int) VALEU_TYPE: 측정결과 변수 (int 또는 float)

▪ 리턴

값	설명
variable	POSX_TYPE - 좌표정보 변수, 예. Posx(x,y,z,rx,ry,rz) INSP_TYPE: 검사결과 변수 int (성공이면 1 리턴) VALEU_TYPE: 측정결과 변수 (int 또는 float)
-1	실패 – 측정데이터 없음 또는 입력 변수 오류

▪ 예제

```
svm_connect()                                # Connect to vision
vision_test=1000                             # Define vision job ID
print_insp=1001                               # Define inspection tool ID
box_size=1002                                 # Define measurement tool ID
count=svm_get_vision_info(vision_test)        # Execute the vision measurement
if (count==1):                                # Check the result
    # Get the position information (posx) of vision_test tool
```

```
pos_result=svm_get_variable(vision_test, POSX_TYPE)
tp_popup("{0}".format(pos_result))

# Get the inspection information (PASS or Fail) of print_insp tool
inspection_result=svm_get_variable(print_insp, INSP_TYPE)
tp_popup("{0}".format(inspection_result))

# Get the distance information (distance) of box_size tool
measurement_result= svm_get_variable(box_size, VALUE_TYPE)
tp_popup("{0}".format(measurement_result))

vs_disconnect() # Disconnect to vision
```

svm_get_offset_pos(posx_robot_init, job_id, tool_id)

10.7 **svm_get_offset_pos(posx_robot_init, job_id, tool_id)**

▪ 기능

사용자가 입력한 로봇 작업 좌표정보에 비전 측정결과를 반영한 로봇 작업 좌표정보를 불러온다.

*순서: posx_robot_init 입력 → vision 측정 → svm_get_offset_pos 호출
→ 변경된 로봇작업 좌표 (posx_robot_offset) 출력

▪ 인수

인수명	자료형	기본값	설명
posx_robot_init	posx	-	로봇 작업 좌표정보 (직접교시 등의 방법으로 입력됨)
job_id	Int	-	비전잡 id (예. 1000, 2000, 3000, ...)
tool_id	int	-	비전툴 id (예. 1000, 1001, 1002, ...)

▪ 리턴

값	설명
posx	비전 측정결과를 반영한 로봇 작업 좌표정보
-1	실패 – 측정데이터 없음 또는 입력 변수 오류

▪ 예제

```
svm_connect()                                # Connect to vision
vision_test=1000                            # Define vision job ID
count=svm_get_vision_info(vision_test)        # Execute the vision measurement
if (count == 1):                             # Check the result
    # Get the position information (posx) of vision_test tool
    pos_result=svm_get_variable(vision_test, POSX_TYPE)
    tp_popup("{0}".format(pos_result))
    # Get the vision guided robot pose
    Id_list =[vision_test]
    pos_list =[pos_result]
    svm_set_init_pos_data(Id_list,pos_list)
```

```
rob_posx=svm_get_offset_pos(posx(200,200,100,0,180,0), vision_test)
tp_popup("{0}".format(rob_posx))
# move to the rob_posx
moveL(rob_posx, vel=30, acc=100)
vs_disconnect()                                # Disconnect to vision
```

10.8 **svm_set_init_pos_data(Id_list, Pos_list)**

▪ 기능

비전 가이던스 작업을 수행할 대상물의 초기 id_list와 posx_list 정보를 입력한다.

⚠ 주의

- * 함수 svm_get_offset_pos (posx_robot_init, job_id, tool_id)를 호출하기 전에 세팅 반드시 세팅해 줄 것.
- * 주의 : id_list 와 pos_list 는 각 id 에 해당하는 posx 와 짹과 맞춰서 사용할 것.

▪ 인수

인수명	자료형	기본값	설명
Id_list	List(int)	-	Id의 list ([id, id, id, ...])
Pos_list	List(Posx)	-	Posx의 list (예.[posx, posx, posx, ...])

▪ 리턴

값	설명
없음	-

▪ 예제

```
svm_connect()                                # Connect to vision
vision_test=1000                             # Define vision job ID
count=svm_get_vision_info(vision_test)        # Execute the vision measurement
if (count == 1):                             # Check the result
    # Get the position information (posx) of vision_test tool
    pos_result=svm_get_variable(vision_test, POSX_TYPE)
    tp_popup("{0}".format(pos_result))
    # Get the vision guided robot pose
    Id_list =[vision_test]
    pos_list =[pos_result]
    svm_set_init_pos_data(Id_list,pos_list)

    rob_posx=svm_get_offset_pos(posx(200,200,100,0,180,0), vision_test)
    tp_popup("{0}".format(rob_posx))
```

```
# move to the rob_posx  
moveL(rob_posx, vel=30, acc=100)  
svm_disconnect() # Disconnect to vision
```

svm_set_tp_popup (svm_flag)

10.9 svm_set_tp_popup (svm_flag)

- **기능**

SVM 오류 발생 시 tp_popup을 띄울지 말지를 설정합니다..

- **인수**

인수명	자료형	기본값	설명
svm_flag	int	1	tp_popup은 1(활성화), 0(비활성화)

- **리턴**

값	설명
없음	-

- **예제**

```
svm_set_tp_popup(0)          # Hide tp_popup  
svm_connect()               # Connect to vision  
svm_disconnect()            # Disconnect to vision
```

10.10 svm_set_led_brightness(value)

- **기능**

SVM의 LED 밝기 값을 설정합니다. .

- **인수**

인수명	자료형	기본값	설명
Value	int	-	LED 밝기 값 (0-1000)

- **리턴**

값	설명
-1	실패 – 측정데이터 없음 또는 입력 변수 오류

예제

```
svm_connect()                                # Connect to vision  
svm_set_led_brightness(500)  
svm_disconnect()                            # Disconnect to vision
```

10.11 svm_get_led_brightness()

- **기능**

SVM에 설정된 LED 밝기 값을 불러옵니다.

- **리턴**

값	설명
int	SVM LED 밝기 값 (0-1000)
-1	실패 – 측정데이터 없음 또는 입력 변수 오류

예제

```
svm_connect()                                # Connect to vision  
svm_get_led_brightness()  
svm_disconnect()                            # Disconnect to vision
```

10.12 svm_set_camera_exp_val(value)

▪ 기능

SVM의 노출 값을 설정합니다.

▪ 인수

인수명	자료형	기본값	설명
value	int	-	SVM 노출 값 (2,660,000 – 29,260,000)

▪ 리턴

값	설명
-1	실패 – 측정데이터 없음 또는 입력 변수 오류

예제

```
svm_connect()                                # Connect to vision  
svm_set_camera_exp_val(2,660,000)  
svm_disconnect()                            # Disconnect to vision
```

svm_set_camera_gain_val(value)

10.13 svm_set_camera_gain_val(value)

▪ **기능**

SVM 게인 값을 설정합니다.

▪ **인수**

인수명	자료형	기본값	설명
value	int	1	SVM 게인 값(0-1600).

▪ **리턴**

값	설명
-1	실패 – 측정데이터 없음 또는 입력 변수 오류

예제

```
svm_connect()                                # Connect to vision  
svm_set_camera_gain_val(500)  
svm_disconnect()                            # Disconnect to vision
```

10.14 svm_set_camera_load(job_id)

▪ 기능

SVM 해당 Job에 저장되어있는 LED밝기/노출(exp)/gain/초점 정보를 불러옵니다.

▪ 인수

인수명	자료형	기본값	설명
job	int		job id(ex - 1000, 2000, 3000)

▪ 리턴

값	설명
-1	실패 – 측정데이터 없음 또는 입력 변수 오류

예제

```

svm_connect()                                # Connect to vision
svm_set_camera_load(job_id)
svm_disconnect()                             # Disconnect to vision

```

10.15 통합예제 (SVM)

- 예제

Vision 작업 설정 상태

- WCM에 저장된 모든 작업을 삭제한 후 아래와 같이 비전작업/툴을 생성한다.
- 비전작업생성: vision_test (position tool, 1000)
- 비전 툴 추가: print_insp (presence tool, 1001)
box_size (distance tool, 1002)
- TW에서 Vision Command에 "vision_test" 작업을 선택하고, 변수정보를 설정한다.
- Custom code에 아래 예제를 추가하여 테스트를 진행한다.

```

svm_connect()                                # Connect to vision
vision_test=1000                            # Define vision job ID
print_insp=1001                            # Define inspection tool ID
box_size=1002                             # Define measurement tool ID
svm_set_job(vision_test)                   # Load the vision_test (1000)
movej(svm_get_robot_pose(vision_test), vel=10, acc=20) # Move to shoot pose (movej)

if (svm_get_vision_info(vision_test)== 1):   # Execute the vision measurement
    # Load the vision variables
    # Get the position information (posx) of vision_test tool
    pos_result=svm_get_variable(vision_test, POSX_TYPE)
    tp_popup("pos_result {0}".format(pos_result))

    # Get the inspection information (PASS or Fail) of print_insp tool
    inspection_result=svm_get_variable(print_insp, INSP_TYPE)
    tp_popup("inspection_result {0}".format(inspection_result))

    # Get the distance information (distance) of box_size tool
    measurement_result= svm_get_variable(box_size, VALUE_TYPE)
    tp_popup("measurement_result {0}".format(measurement_result))

    # Move to the vision guided robot pose
    # Get the vision guided robot pose

```

```
Id_list =[vision_test]
pos_list =[pos_result]
svm_set_init_pos_data(Id_list,pos_list)

rob_posx=svm_get_offset_pos(posx(200,200,100,0,180,0), vision_test)
tp_popup("rob_posx {0}".format(rob_posx))

# move to the rob_posx
moveL(rob_posx, vel=30, acc=100)

svm_disconnect() # Disconnect to vision
```

11. Application 명령어

11.1 Conveyor Tracking

11.1.1 set_conveyor(name)

- 기능

UI에서 Conveyor정보를 설정한 경우, 프로그램에서 Conveyor Tracking Application을 시작할 수 있도록 해당 Conveyor 이름으로 ID를 획득하고 작업물 모니터링 명령을 내립니다. 작업물 모니터링은 Conveyor에서 Triggering 되는 작업물에 대해 수행되며 프로그램이 종료될 때까지 계속됩니다.

- 인수

인수명	자료형	기본값	설명
name	string	-	Conveyor 이름

- 리턴

값	설명
Conveyor ID	Conveyor 설정 성공 시 Conveyor ID를 리턴
None	Conveyor 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

```
CONV1 = set_conveyor("conveyor1")
```

- 관련 명령어

get_conveyor_obj(), tracking_conveyor(), untracking_conveyor()

```
11.1.2 set_conveyor_ex(name="", conv_type=0, encoder_channel=1,
triggering_mute_time=0.0, count_per_dist=5000,
conv_coord=posx(0,0,0,0,0), ref=DR_BASE, conv_speed=100.0,
speed_filter_size=500, min_dist=0.0, max_dist=1000.0,
watch_window=100.0, out_tracking_dist=10.0)
```

■ 기능

Conveyor Tracking Application을 시작할 수 있도록 Conveyor를 설정하고 ID를 획득합니다. 명령이 수행된 이후에는 프로그램이 종료될 때까지 설정된 Conveyor에서 Triggering되는 물체들을 모니터링합니다. Conveyor 정보 setting을 UI를 통해서 설정할 수 없어서 파라미터를 직접 설정해야 할 때 사용합니다.

¹⁾M2.4.0 이전 버전 대비 전체 인자의 기본값 추가

²⁾M2.4.0 이전 버전 대비 ref 인자 추가 (world 좌표계 반영)

³⁾M2.4.0 이전 버전 대비 obj_offset_coord 인자 삭제, obj_offset_coord 인자는 11.1.3 get_conveyor_obj() 함수에서만 입력하는 것으로 변경

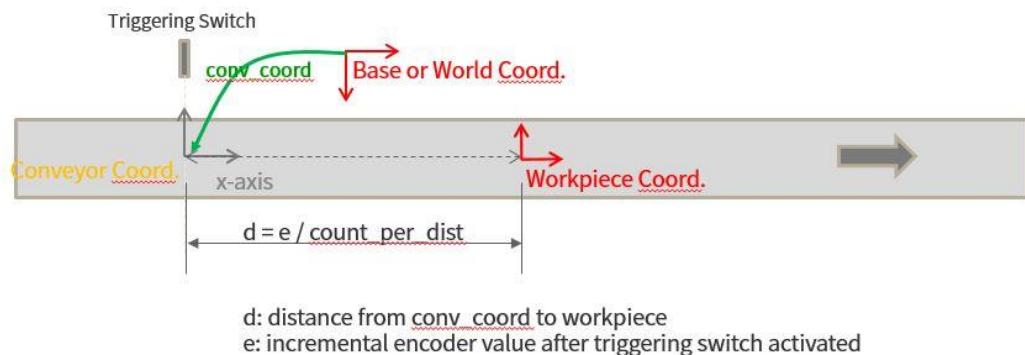
■ 인수

인수명	자료형	기본값	설명
name	string	""	Conveyor 이름
conv_type	int	0	Conveyor 타입(0: Linear, 1: Circular)
encoder_channel	int	1	외부 encoder 채널(1, 2)
triggering_mute_time	float	0.0	Triggering 직후에 triggering 신호가 들어와도 triggering(encoder reset, 작업물 추적 시작)을 수행하지 않는 시간(s).
count_per_dist	int	5000	길이 당 엔코더 카운트 환산값 (Linear: count/m, Circular: count/rad)
conv_coord	posx list (float[6])	posx(0,0,0,0,0,0)	컨베이어 고정 좌표계 (Base/World 좌표계 기준, mm, °)
ref	int	DR_BASE	컨베이어 고정 좌표계의 기준 좌표계(DR_BASE: Base, DR_WORLD: World)

인수명	자료형	기본값	설명
conv_speed	float	100.0	컨베이어 nominal 속도(Linear: mm/s, Circular: °/s)
speed_filter_size	int	500	컨베이어 속도 Filtering 시 Moving Average Filter Size
min_dist	float	0.0	컨베이어 작업 최소 길이(Triggering Switch 기준, Linear: mm, Circular: °)
max_dist	float	1000.0	컨베이어 작업 최대 길이(Triggering Switch 기준, Linear: mm, Circular: °)
watch_window	float	100.0	컨베이어 작업 대기 모니터링 길이(작업 최소 길이 기준, Linear: mm, Circular: °)
out_tracking_dist	float	10.0	컨베이어 Tracking 해제 여유 구간 길이(작업 최대 길이 기준, Linear: mm, Circular: °)

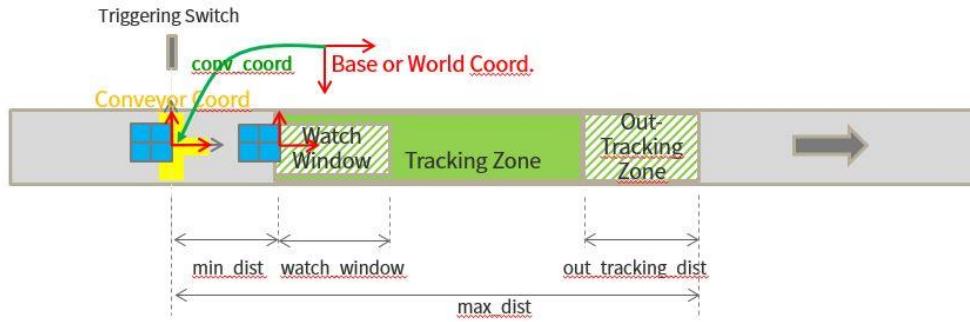
알아두기

- conv_type 인자에서 Circular Conveyor는 현재 지원하지 않습니다!
- set_conveyor() 또는 set_conveyor_ex() 함수 실행 이후 프로그램 종료 전까지, Triggering Switch를 지나는 모든 작업물은 max_dist를 지날 때까지 모니터링 됩니다.
- 단, triggering_mute_time을 설정할 경우, 이전 작업물 감지 이후에 해당 시간 동안은 Triggering Switch가 작동되어도 모니터링 목록에 포함시키지 않습니다. Triggering Switch의 노이즈가 있거나 의도적으로 일정 시간 동안 작업물을 배제하고 싶을 때 사용합니다.
- conv_coord는 컨베이어에 고정된 좌표계로서 베이스 또는 월드 좌표계 기준의 컨베이어 좌표계입니다. 여기서, **conv_coord**의 x축은 컨베이어가 흘러가는 방향을 나타냅니다. 컨베이어 작업물이 Triggering Switch를 작동시키는 순간부터 증가된 엔코더 값은 count_per_dist 인자를 이용하여 작업물이 이동한 길이로 환산할 수 있고, 이 길이를 conv_coord의 x축 방향으로 연장하면 기준 좌표계 대비 작업물의 위치를 추적할 수 있습니다.



[Conveyor 좌표계와 작업물 좌표계]

- conv_speed는 컨베이어의 이동 속도입니다. 엔코더로 센싱되는 컨베이어 속도가 이 속도의 200%를 넘을 경우 Info를 주는 용도로만 사용되므로 TP UI를 통해서 측정할 수 없는 경우 대략적인 값을 입력합니다.
- speed_filter_size는 엔코더로부터 컨베이어 속도를 추정할 때 사용되는 moving-average filter의 사이즈입니다. 크기가 클수록 노이즈를 상쇄시킬 수 있지만, 가감속 시 tracking accuracy가 저하될 수 있습니다.
- 컨베이어 위에 구간은 Watch Window, Tracking Zone, Out-Tracking Zone으로 구분됩니다.
- Watch Window는 Tracking을 위해서 작업물 좌표계 획득을 하려고 할 때, 해당 구간 안에 있는 작업물만 작업 가능하다고 판단하는 구간입니다. get_conveyor_obj() 함수를 호출 했을 때, 이 구간 안에 있지 않으면 함수 리턴을 해주지 않고, 작업물이 있으면 get_conveyor_obj() 함수 옵션(FIFO, LIFO)에 따라서 작업물 좌표계를 리턴해줍니다.
- Tracking Zone은 Conveyor Tracking을 수행하는 구간입니다.
- Out-Tracking Zone은 Tracking을 계속하면 로봇이 고유의 작업 영역 또는 사용자가 설정한 작업 영역을 벗어난다고 판단하여 로봇의 Tracking을 자동으로 종료하게 되는 구간입니다.
- 3개의 구간은 아래 그림과 같이 4개의 길이(min_dist, max_dist, watch_window, out_tracking_dist)로 정의됩니다.



[Conveyor 구간 및 길이 설명]

▪ 리턴

값	설명
Conveyor ID	Conveyor 설정 성공 시 Conveyor ID를 리턴
None	Conveyor 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

```
CONV1 = set_conveyor_ex(name='conveyor_1',
                        conv_type=0,                      # linear
                        encoder_channel=1, triggering_mute_time=0.0,
                        count_per_dist=5000,                # 5000 count/mm)
                        conv_coord=posx(500, 100, 500, 0, -90, 0), ref=DR_BASE,
                        conv_speed=100.0,                  # conveyor speed: 100 mm/s,
                        speed_filter_size=500,             # moving avg. filter size: 500 ms
                        min_dist=100, max_dist=1000, watch_window=200, out_tracking_dist=10)
```

- 관련 명령어

`get_conveyor_obj()`, `tracking_conveyor()`, `untracking_conveyor()`

11.1.3 get_conveyor_obj(conv_id, timeout=None, container_type=DR_FIFO, obj_offset_coord=None)

- **기능**

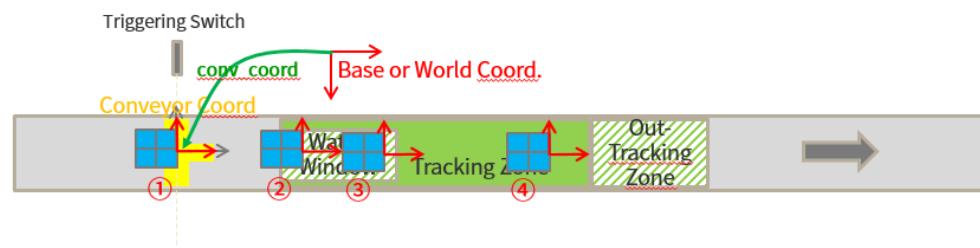
해당 Conveyor로부터 작업 가능한 작업물 좌표계 ID를 리턴합니다. 함수가 불렸을 때, Watch Window에 있는 작업물 중 설정한 container 규칙에 따라 작업물을 한 개씩 반환해줍니다.

- **인수**

인수명	자료형	기본값	설명
conv_id	int	-	Conveyor ID
timeout	float	None	설정한 시간동안 리턴할 작업물이 없으면 대기 종료하고 함수 리턴
container_type	int	DR_FIFO	작업물 container type(DR_FIFO: first-in/first-out, DR_LIFO: last-in/last-out)
obj_offset_coord	posx	None	컨베이어 고정 좌표계 대비 작업물 좌표계(mm, °)
	list (float[6])		

알아두기

- 해당 함수를 호출했을 때, Watch Window에 있는 작업물을 설정한 Container 규칙에 따라 작업물 좌표계 id를 한 개씩 반환해줍니다. 예를 들어, 아래와 같이 작업물이 배치되어 있는 순간 get_conveyor_obj 함수를 호출하면 Watch Window 안에 있는 ②, ③ 작업물이 후보가 됩니다. 이 때, container_type을 DR_FIFO로 설정한 경우 Watch Window에 먼저 들어온 ③ 작업물 좌표계 id를 반환하고, DR_LIFO로 설정한 경우 Watch Window에 나중에 들어온 ② 작업물 좌표계 id를 반환합니다. 만약에 함수 호출 당시에 Watch Window에 작업물이 없으면 최대 timeout 변수에 설정된 시간까지 대기하고 있다가 그 사이에 작업물이 들어오면 해당 id를 반환해줍니다.

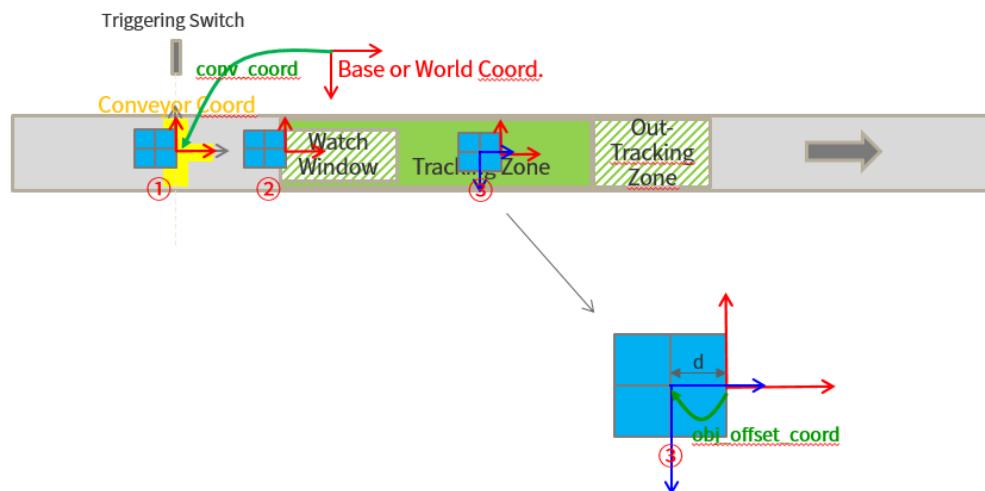


[작업물 좌표계 id 반환 규칙 설명]

- obj_offset_coord는 작업물 좌표계에 offset을 적용하고 싶을 때 사용합니다. 보통 티징점을 손쉽게 입력하기 위해서 사용하거나 외부센서(ex. Vision 센서)와 연동해서 작업물의 자세와 위

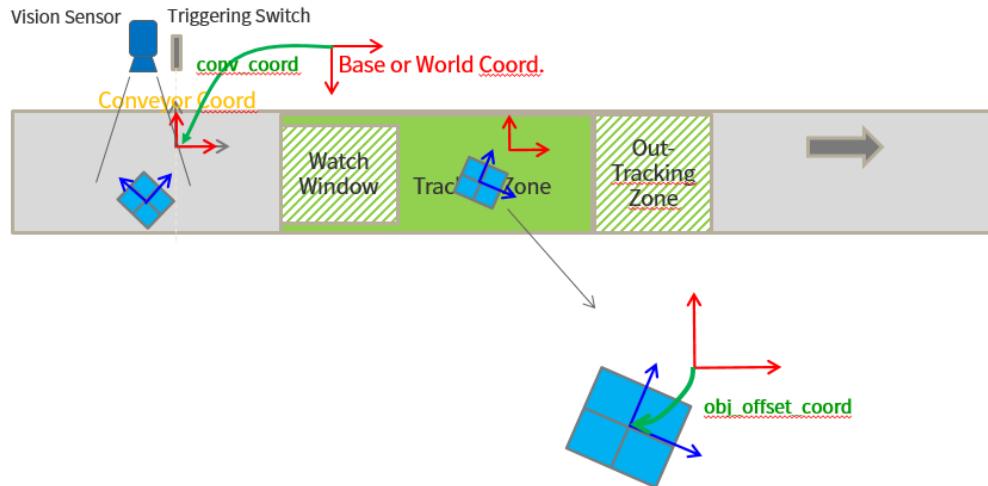
치를 동적으로 바꿔주고 싶을 때 사용합니다. 입력하지 않으면 offset은 0이 적용됩니다.

아래 그림 같은 경우, 작업물 좌표계가 작업물 우측면에 생성되고, Base or World Coord.과 작업물 좌표계가 90도 틀어져있습니다. 이 때, 티칭점의 위치는 작업물 가운데 기준으로, 티칭점의 자세는 Base or World Coord. 기준으로 부여하고 싶은 경우 `obj_offset_coord = posx(-d, 0, 0, -90, 0, 0)`으로 적용하면 됩니다. 이 TP UI를 통해서 티칭점을 획득할 경우 필수적인 부분은 아니지만, drl만을 사용하거나 티칭점을 직접 입력해야 할 경우 필요합니다.



[`obj_offset_coord` 적용 사례 1]

다음으로, 아래 그림 같이 작업물이 컨베이어 진행 방향과 무관한 방향으로 위치가 변하는 경우 또는 작업물의 자세가 바뀌는 경우 엔코더 신호만으로는 작업물의 위치/자세를 알아낼 수 없으므로 비전 센서를 써서 작업물의 추가적인 위치/자세 변화를 감지해야 합니다. 그리고, 여기서 감지한 위치/자세 변화를 동적으로 `obj_offset_coord`에 입력해주면 작업물 좌표계를 이에 맞게 생성합니다.



[obj_offset_coord 적용 사례 2: 비전 센서 사용]

▪ 리턴

값	설명
int	CONV_COORD. Conveyor 유저 좌표계 ID (121~150)
음의 정수	timeout 시간이 지나도 반환할 작업물이 없을 경우

☞ 알아두기

반환할 작업물이 없을 경우, timeout 시간이 지날 때까지 함수를 리턴해주지 않습니다. timeout 시간이 지났지만 해당 작업물이 없으면 -1을 반환합니다. 단, timeout 시간을 입력하지 않은 경우는 계속해서 함수를 리턴해주지 않습니다.

▪ 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

```

## One object in a cycle
CONV1 = set_conveyor('conveyor1')

movel(posx(100, 100, 50, 0, 0, 0), ref=DR_BASE) # waiting position
while True:
    CONV_COORD_1 = get_conveyor_obj(CONV1)
    tracking_conveyor(CONV1)

    # synched motion
    movel(posx(0,0, 50, 0, 0, 0), ref=CONV_COORD_1)
    movel(posx(0,0, 0, 0, 0, 0), ref=CONV_COORD_1)
    set_digital_output(DO_GRIPPER, 1)
    movel(posx(0,0, 50, 0, 0, 0), ref=CONV_COORD_1)

    untracking_conveyor(CONV1)

    movel(posx(100, 100, 50, 0, 0, 0), ref=DR_BASE) # waiting position

## Multi objects in a cycle
CONV1 = set_conveyor('conveyor1')

while True:
    CONV_COORD_1 = get_conveyor_obj(CONV1)
    tracking_conveyor(CONV1)

    # fist object
    movel(posx(0,0, 50, 0, 0, 0), ref=CONV_COORD_1)
    movel(posx(0,0, 0, 0, 0, 0), ref=CONV_COORD_1)
    set_digital_output(DO_GRIPPER, 1)
    movel(posx(0,0, 50, 0, 0, 0), ref=CONV_COORD_1)

    # second object
    CONV_COORD_2 = get_conveyor_obj(CONV1, time_out=10)

```

```
if CONV_COORD_2 > 0: # -1 if no objects available during time_out
    movel(posx(0,0, 50, 0, 0, 0), ref=CONV_COORD_2)
    movel(posx(0,0, 0, 0, 0, 0), ref=CONV_COORD_2)
    set_digital_output(DO_GRIPPER, 1)
    movel(posx(0,0, 50, 0, 0, 0), ref=CONV_COORD_2)

    # first object if you need
    movel(posx(0,0, 50, 0, 0, 0), ref=CONV_COORD_1)

untracking_conveyor(CONV1)

movel(posx(100, 100, 50, 0, 0, 0), ref=DR_BASE)
```

- 관련 명령어

tracking_conveyor(), untracking_conveyor()

11.1.4 tracking_conveyor(conv_id)

- 기능

로봇이 Conveyor Tracking을 시작합니다. 정지 상태에서 가속하여 Conveyor 속도에 균접하면 리턴합니다.

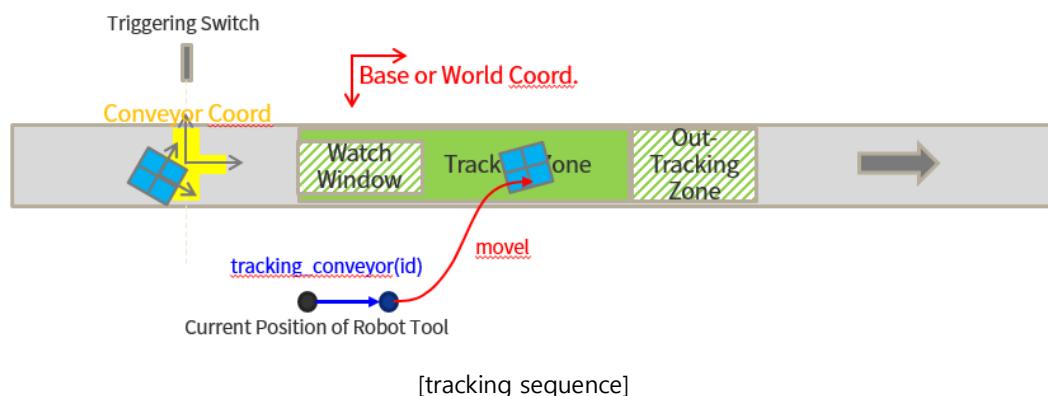
- 인수

인수명	자료형	기본값	설명
conv_id	int	-	Conveyor ID



알아두기

tracking_conveyor 명령이 떨어지면 로봇 현재 위치에서 Conveyor를 tracking하기 시작하고 정지 상태에서 가속하여 Conveyor 속도에 균접하면 함수를 리턴해줍니다. 함수 리턴 후에 task motion 을 수행할 수 있습니다.



- 리턴

값	설명
0	속도 동기화 성공
음의 정수	속도 동기화 과정 중 로봇 작업영역을 벗어날 것으로 예상될 때

- 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

예외	설명
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

```
CONV1 = set_conveyor('conveyor1')

while True:
    CONV_COORD_1 = get_conveyor_obj(CONV1)

    tracking_conveyor(CONV1)      # start moving to track conveyor

    # task on conveyor
    movel(posx(0,0, 50, 0, 0, 0), ref=CONV_COORD_1)
    movel(posx(0,0, 0, 0, 0, 0), ref=CONV_COORD_1)
    set_digital_output(DO_GRIPPER, 1)
    movel(posx(0,0, 50, 0, 0, 0), ref=CONV_COORD_1)

    untracking_conveyor(CONV1)
    obj_count = obj_count + 1
```

■ 관련 명령어

untracking_conveyor()

11.1.5 untracking_conveyor(conv_id, time=None)

▪ 기능

로봇이 Conveyor Tracking을 종료합니다. 종료 모션을 마치고 속도가 0으로 떨어지면 리턴합니다.

▪ 인수

인수명	자료형	기본값	설명
conv_id	int	-	Conveyor ID
time	float	None	Tracking 종료를 위한 감속 시간(sec)



알아두기

- time 이 None 인 경우 Default 로 설정된 속도로 감속합니다. 싸이클 타임 단축이 필요한 경우 감속 시간을 조정합니다. 로봇이 낼 수 있는 최대 감속 시간보다 짧게 입력한 경우에는 시간이 무시되고 최대 감속 시간으로 정지합니다.

▪ 리턴

값	설명
0	속도 동기화 종료 성공
음의 정수	속도 동기화 종료 모션 중 로봇 작업영역을 벗어날 것으로 예상될 때

▪ 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

```
CONV1 = set_conveyor('conveyor1')

while True:
    CONV_COORD_1 = get_conveyor_obj(CONV1)
```

```
tracking_conveyor(CONV1)

# task on conveyor
moveL(posx(0,0, 50, 0, 0, 0), ref=CONV_COORD_1)
moveL(posx(0,0, 0, 0, 0, 0), ref=CONV_COORD_1)
set_digital_output(DO_GRIPPER, 1)
moveL(posx(0,0, 50, 0, 0, 0), ref=CONV_COORD_1)

untracking_conveyor(CONV1, 0.1)
```

▪ 관련 명령어

tracking_conveyor()

11.2 External Encoder 설정 명령어

11.2.1 set_extenc_polarity(channel, polarity_A, polarity_B, polarity_Z, polarity_S)

- **기능**

해당 엔코더 채널의 A, B상 극성과 S, Z상의 트리거 방식을 설정한다.

- **인수**

인수명	자료형	기본값	설명
Channel	int	1	엔코더 채널(1, 2) 1: 채널 1 2: 채널 2
polarity_A	int	0	A상의 극성(0: A상, 1: /A상)
polarity_B	int	0	B상의 극성(0: B상, 1: /B상)
polarity_Z	int	0	Z상의 트리거 방식(0: 하강에지, 1: 상승에지)
polarity_S	int	0	S상의 트리거 방식(0: 하강에지, 1: 상승에지)

- **리턴**

값	설명
N/A	사용안함

- **예외**

예외	설명
N/A	사용안함

- **예제**

```
set_extenc_polarity(1, 0, 1, 0, 1)
# External Encoder channel 1을 A상, /B상, Z상(하강에지), S상(상승에지)로 설정
```

- 관련 명령어

set_extenc_mode

11.2.2 set_extenc_mode(channel, mode_AB, pulse_AZ, mode_Z, mode_S, inverse_cnt)

- **기능**

해당 엔코더 채널의 A, B, Z, S상의 동작 모드를 설정한다.

- **인수**

인수명	자료형	기본값	설명
channel	int	1	엔코더 채널(1, 2) 1: 채널 1 2: 채널 2
mode_AB	int	0	AB상의 사용 Mode(0 ~ 4) 0: 미사용 1: A상 Quadrature 사용 B상 Quadrature 사용 2: A상 Count B상 Direction 사용 3: A상 Up Count 사용 B상 미사용 4: A상 Down Count 사용 B상 미사용
pulse_AZ	int	0	Z펄스당 A펄스의 카운트 (0 ~ 100000)
mode_Z	int	0	Z상의 사용 Mode(0 ~ 1) 0: 미사용 1: A/B Count 오차 보전 2: Encoder Count Clear
mode_S	int	0	S상의 사용 Mode(0 ~ 1) 0: 미사용 1: Strobe Signal 2: Encoder Count Clear
inverse_cnt	int	0	Encoder Count의 방향 반전 여부 0: 정방향 1: 역방향

- 리턴

값	설명
N/A	사용안함

- 예외

예외	설명
N/A	사용안함

- 예제

```
set_extenc_mode(1, 2, 20000, 1, 1, 0)
# External Encoder channel 1의 동작 모드를 아래와 같이 설정
# A상 Count, B상 Direction 사용
# Z펄스당 A펄스 카운트는 20000
# Z상을 여러 카운트 누적 보상 모드로 사용, S상 사용
# Encoder Count의 방향은 정방향으로 설정
```

- 관련 명령어

set_extenc_polarity

11.2.3 get_extenc_count(channel)

- **기능**

해당 엔코더 채널의 현재 카운트 값을 구한다.

- **인수**

인수명	자료형	기본값	설명
channel	int	1	엔코더 채널(1, 2) 1: 채널 1 2: 채널 2

- **리턴**

값	설명
count	해당 채널의 Encoder Count 현재 값

- **예외**

예외	설명
N/A	사용안함

- **예제**

```
enc_cnt = get_extenc_count(1)
# External Encoder channel 1의 현재 count 값을 구함
```

- **관련 명령어**

set_extenc_polarity

set_extenc_mode

clear_extenc_count

11.2.4 clear_extenc_count(channel)

- **기능**

해당 엔코더 채널의 카운트 값을 0으로 초기화한다.

- **인수**

인수명	자료형	기본값	설명
channel	int	1	엔코더 채널(1, 2) 1: 채널 1 2: 채널 2

- **리턴**

값	설명
N/A	사용안함

- **예외**

예외	설명
N/A	사용안함

- **예제**

```
clear_extenc_count(1)
# External Encoder channel 1의 count 값을 0으로 초기화함
```

- **관련 명령어**

get_extenc_count

12. A-Series 전용 명령어

12.1 get_function_input(index)

- 기능

컨트롤러의 디지털 입력 점접에서 신호를 불러오기 위한 명령문으로 디지털 입력 점접 값을 읽습니다.

Process Button 장치의 Function 버튼 상태를 읽기 위한 명령문입니다.

- 인수

인수명	자료형	기본값	설명
index	int	-	Process Button 장치의 Function 버튼 번호이며, 1 ~ 4까지 가능함.

- 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_function_input(1)      #1번 Function 버튼 상태 읽기
in8 = get_function_input(4)      #4번 Function 버튼 상태 읽기
```

```
flange_serial_open(baudrate=115200, bytesize=DR_EIGHTBITS,  
parity=DR_PARITY_NONE, stopbits = DR_STOPBITS_ONE)
```

12.2 flange_serial_open(baudrate=115200, bytesize=DR_EIGHTBITS, parity=DR_PARITY_NONE, stopbits = DR_STOPBITS_ONE)

▪ 기능

Flange Serial 통신 포트를 열기 위한 명령어입니다.

▪ 인수

인수명	자료형	기본값	설명
baudrate	int	115200	Baud rate 2400, 4800, 9600, 19200, 38400, 57600, 115200 etc
bytesize	int	8	데이터 bit 수 • DR_FIVEBITS: 5 • DR_SIXBITS: 6 • DR_SEVENBITS: 7 • DR_EIGHTBITS: 8
parity	str	"N"	Parity checking • DR_PARITY_NONE: "N" • DR_PARITY EVEN: "E" • DR_PARITY ODD: "O"
stopbits	int	1	Stop bit의 수 • DR_STOPBITS_ONE =1 • DR_STOPBITS_TWO = 2

▪ 리턴

값	설명
0	성공
음수값	실패

▪ 예외

예외	설명
DR_Error (DR_ERROR_TYPE)	인수들의 데이터형 오류 시

예외	설명
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

12.3 flange_serial_close()

- **기능**

Flange Serial을 닫기 위한 명령어입니다..

- **리턴**

값	설명
0	성공

- **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

12.4 flange_serial_write(tx_data)

▪ 기능

Flange Serial에 data를 쓰기 위한 명령어입니다.

▪ 인수

인수명	자료형	기본값	설명
tx_data	byte	-	송신할 데이터(최대32byte) • 데이터 타입 byte 형이어야 합니다.

▪ 리턴

값	설명
0	성공

▪ 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

12.5 flange_serial_read()

- **기능**

Flange Serial에서 data를 읽어오기 위한 명령어입니다.

- **리턴**

값	설명
res	수신한 데이터의 바이트 수
rx_data	Read 한 데이터

- **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



Doosan Robotics

www.doosanrobotics.com