Robotic Arm Dataset (RAD) Features Description

1 Command Dataset

The command dataset was collected from six different cyber-physical system (CPS) devices: **(i)** four-axis N9 robot arm from North Robotics; **(ii)** six-axis UR3e robot arm from Universal Robots; **(iii)** C-Mag HS 7 magnetic stirrer and heater from IKA; **(iv)** 100-240V, 50/60Hz Fisherbrand Mini-Centrifuge from Fisher Scientific; **(v)** Cavro XLP 6000 syringe pump from Tecan; and **(vi)** Quantos powder dosing system from Mettler Toledo. These devices have commands, arguments and responses associated with them that we further explain in detail. We do not report any connection-related parameters like IP addresses or port numbers, which are typically part of initialization requests, as they are not relevant.

The centrifuge is the simplest of all modules. It is connected via the C9 controller, and can be directed to start and stop spinning using the `OUTP n` command exposed by the C9 controller, where n denotes the port to which the centrifuge is connected (Table 1).

Command	**Arguments (type) / Responses (type)**	**Description**
OUTP n	Status (b) / -	Directs centrifuge to start and stop spinning

Table 1: Fisherbrand 100-240V 50/60Hz Mini-Centrifuge

Command	**Arguments (type) / Responses (type)**	**Description**
set_home_direction	Direction (i) / -	Sets the home direction
move_z_stage	Steps (i) / - Speed (f) / -	Moves the z stage in specified steps
home_z_stage	- / - *	Homes the z stage

Table 2: Arduino-Controller Stepper Motor

The IKA C-Mag HS 7 magnetic stirrer and heater exposes a simple programmable API to control its motor (which is used for stirring) and heater (Table 3). Commands START_1 (START_4) and STOP_1 (STOP_4) are used to start and stop the heater (motor), respectively. Commands SET_MODE_m (where m = A, B, or D) and RESET allow switching between different operating modes. Commands IN_* and OUT_* are used to get and set different parameters, respectively. While both OUT_WD1 and OUT_WD2 set the watchdog time, they enable different watchdog modes.

Like the IKA module, the Tecan Cavro XLP 6000 syringe pump also exposes a simple API (Table 4). Commands R and T are used to start and stop the pump; w and W are used to home the valves and the plunger; Z is used to set the homing position; g and G are used to start and stop the batch command loop; O and I are used to switch the valve to a set or given position; U, K, V, L, A, P, and D are used to set different parameters; and Q, ?, ?1, ?2, ?3, and ?6 are used to get different parameters.

Command	Arguments (type) / Responses (type)	Description
init	- / -	Initializes the Magnetic Stirrer
IN_NAME	- / Device Name (s)	Read device name
START_1	- / -	Start the heater
STOP_1	- / -	Stop the heater
START_4	- / -	Start the motor
STOP_4	- / -	Stop the motor
SET_MODE_A	- / -	Set operating mode A
SET_MODE_B	- / -	Set operating mode B
SET_MODE_D	- / -	Set operating mode D
RESET	- / -	Switch to normal operating mode
OUT_SP_1	Temperature (f) / -	Adjust the set temperature value
OUT_SP_4	Speed (f) / -	Adjust the set speed value
OUT_SP_12	Temperature safety limit (f) / -	Setting WD safety limit temperature with set value echo
OUT_SP_42	Speed safety limit (f) / -	Setting WD safety limit speed with set value echo
OUT_WD1	Watchdog time (f) / -	Setting the Watchdog mode 1
OUT_WD2	Watchdog time (f) / -	Setting the Watchdog mode 2
IN_PV_1	- / Sensor value (f)	Read actual external sensor value
IN_PV_2	- / Hotplate sensor value (f)	Read actual hotplate sensor value
IN_PV_4	- / Stirring speed (f)	Read stirring speed value
IN_PV_5	- / Viscosity trend (f)	Read viscosity trend value
IN_SP_1	- / Rated temperature (f)	Read rated temperature value
IN_SP_3	- / Rated safety temperature (f)	Read rated set safety temperature value
IN_SP_4	- / Rated speed (f)	Read rated speed value

Table 3: IKA C-Mag HS 7 Magnetic Stirrer and Heater

The *ArduinoAugmentedQuantos* class controls both the Mettler Toledo Quantos system for powder dosing and the Aurduino-controlled stepper motor that is tasked with its z-axis control. Unlike the IKA and Tecan modules above, the Quantos module can be configured in many different ways, as evident from the commands shown in Tables 2 and 6. The commands are self-explanatory (a positive consequence of tracing a higher-level class), and each command maps to a unique device-level command. Note that we omit a range of bookkeeping parameters such as user and sample IDs, and calibration and expiry dates; while these are returned as part of head_data and sample_data APIs, they are not currently being used by RosyChem Lab. RATracer's data processing module therefore also filters these out.

Command	Arguments (type) / Responses (type)	Description
init	Syringe volume (f) / - Counts per stroke (i) / - Velocity scale (i) / - Dead volume (i) / - Total valve position (i) / - Distribution valve (b) / - Slope code (i) / - Wait timeout (f) / -	Initializes the syringe pump
R	- / -	Start the pump
T	- / -	Stop the pump
w	- / -	Home the valves
W	Plunger home speed (i) / -	Home the plunger
Z	Plunger home speed (i) / -	Set the homing position
g	- / -	Start the batch command loop
G	Iteration count (i) / -	Stop the batch command loop
O	- / -	Set the valve to a set position
I	Position (i) / -	Set the valve to a given position
U	Pump Configuration (i) / -	Set pump configuration
K	Dead volume (i) / -	Set dead volume of pump
V	Velocity (i) / -	Set velocity of pump
L	Slope code (i) / -	Set slope code of pump
A	Position in counts (i) / -	Set position in counts
P	Distance in counts (i) / -	Set distance in counts
D	Distance (opp. direction) in counts (i) / -	Set distance in counts when moving in opposite direction
Q	- / Pump status (i)	Get pump status
?	- / Position in counts (i)	Get the current position in counts
?1	- / Start speed (i)	Get the current start speed
?2	- / Default speed in counts/s (i)	Get the default velocity in counts/s
?3	- / Cutoff speed (i)	Get the current cutoff speed
?6	- / Valve position (i)	Get the current valve position

Table 4: Tecan Cavro XLP 6000 Syringe Pump

Next, we present the robot arm APIs. We start with the N9, which is a four-axis robot arm from North Robotics. The N9 is controlled via its C9 controller. Table 5 lists a subset of the C9 APIs that is used to control the N9. Many commands are applied separately to each axis and therefore take a list of target axes as an argument. Commands BIAS, JLEN, and SPED are used to set different parameters; SPED also returns the value of parameters set by the robot. POS gets the position of the gripper or the end effector. ARM is the main command used to move the robot arm to the specified position; it also returns the final position after movement. Note that a position defines a point in the 3D space using a vector

of three values.

Command	Arguments (type) / Responses (type)	Description
init	Use joystick (b) / -	Initializes the N9 robot arm
HOME	Axes (i $\times$ 4) / - Home only if needed (b) / - Skip actual homing (b) / -	Homes the N9 robot
HALT	Axes (i $\times$ 4) / -	Halts the given axes
BIAS	Elbow bias (i) / -	Sets the elbow bias
CURR	Axis (i) / Max current (i) / - Max (b) / -	Returns the actual or max current for the given axis, setting the max current if given
JLEN	Length (f) / -	Sets the elbow length
SPED	Default velocity (i) / Default velocity (i) Default acceleration (i) / Default acceleration (i)	Returns the velocity and acceleration of the N9 robot
POS	- / Position (f $\times$ 3)	Gets the position of the gripper or the end effector
MVNG	Axes (i $\times$ 4) / Moving States (b $\times$ 4)	Returns a list of moving statuses for the given axes
ARM	Position (f $\times$ 3) / Position (f $\times$ 3) Gripper position (f) / - Relative move (b) / - Velocity (f) / - Acceleration (f) / - Elbow bias (i) / -	Moves the robot arm to the specified position

Table 5: North Robotics N9 Robot

Command	Arguments (type) / Responses (type)	Description
init	Logging level (i) / - Perfect counts (i) / -	Initializes the quantos
start_dosing	- / -	Starts dosing of the solid
stop_dosing	- / -	Stops dosing of the solid
lock_dosing_pin_pos	- / -	Locks the dosing head to the dosing unit [sic]
unlock_dosing_pin_pos	- / -	Unlocks the dosing head from the dosing unit [sic]
set_pan_empty	- / -	Tells the Quantos that the pan is empty
tap_before_dosing	Yes/no (b) / -	Tap before dosing enabled

tap_while_dosing	Yes/no (b) / -	Tap during dosing enabled
tapper_intensity	Intensity (i) / -	Sets tapper intensity
tapper_duration	Seconds (i) / -	Sets tapper duration in seconds
target_mass	Milligrams (f) / -	Sets target mass in milligrams
tolerance_value	% (f) / -	Sets the tolerance value for powder dosing in percent
tolerance_mode	+/- or 0/+ (b) / -	Sets tolerance mode (+/- or 0/+)
dosing_algorithm	M/P/H (i) / -	Powder dosign algorithm (standard or advanced)
antistatic_enabled	Yes/no (b) / -	AntiStatic kit enabled
front_door_position	- / Position (i)	Returns the front door position
front_door_position	Position (i) / -	Sets the front door position
sampler_position	- / Position (i)	Get sampler position
sampler_position	Position (i) / -	Sets sampler position
sampler_status	- / On/off (b)	Returns sampler status
weigh_pan_status	- / Empty or not (b)	Returns the status of the weighing pan
zero	- / -	Zeros the balance reading
head_data	- / Leveled (b) / Dose limit (i) / No. of doses (i) / Remaining quantity (f)	Returns the contents of the RFID chip of the dose head
sample_data	- / Quantity (f) / Target quantity (f) / Tolerance % (f) / Validity (b) / Accuracy % (f) / Seconds (f) / Tap while dosing (b) / Tap intensity (i) / Dosing mode (b) / Leveled (b) / Dose limit (i) / No. of doses (i) / Quantity left (f)	Returns the results data of the last dispense

Table 6: Mettler Toledo Quantos Powder Dosing System

Command	**Arguments (type) / Responses (type)**	**Description**
init	Velocity (f) / - Position units (s) / - Max. velocity (f) / - Joint velocity (f) / - Max. joint velocity (f) / - Gripper velocity (f) / - Gripper force (f) / -	Initializes the robot arm
default_joint_velocity	Joint velocity (f) / Joint velocity (f)	Sets and gets the default joint velocity
pose	- / TCP position (f $\times$ 6)	Gets the pose of the robot arm
joint_positions	- / Joint positions (f $\times$ 6)	Gets the joint positions of the robot arm
joint_count	- / Joint count (i)	Gets the joint count
tool_offset	Tool offset location (f $\times$ 6)/ -	Sets the tool offset
tool_mass	Payload (f) / -	Sets the tool mass
gripper_position	- / Gripper position (f)	Gets the gripper position
stop	- / -	Stops the robot arm
emergency_stop	- / -	Emergency stop of the robot arm
wait	Timeout (f) / - Wait for running (b) / -	Waits for the robot arm to start or stop and timeouts after a while
move_to_location	Location (f $\times$ 6) / - Base reference frame (f $\times$ 6)/ - Velocity (f) / - Acceleration (f) / - Relative (b) / - Wait (b) / - Timeout (f) / -	Move robot arm to the specified location
move_joints	Joint positions (f $\times$ 6) / - Velocity (f) / - Acceleration (f) / - Relative (b) / - Wait (b) / - Timeout (f) / -	Moves the joints to the specified location
move_circular	Midpoint location (f $\times$ 6) / - End location (f $\times$ 6) / - Base reference frame (f $\times$ 6)/ - Velocity (f) / - Accelration (f) / -	Moves the arm in a circular direction

	Wait (b) / - Timeout (f) / -	
open_gripper	Position (f) / - Force (f) / - Velocity (f) / - Wait (b) / - Timeout (f) / -	Opens the gripper
close_gripper	Position (f) / - Force (f) / - Velocity (f) / - Wait (b) / - Timeout (f) / -	Closes the gripper
wait_for_gripper_stop	Timeout (f) / -	Waits for the gripper to stop and timeouts after a while

Table 7: Universal Robots UR3e Robot Arm

Unlike the N9, the UR3e is connected over TCP, via a third-party Python package *urx*. As a result, RATracer traces the methods exposed by RosyChem Lab's *class UR3Arm(RobotArm)* rather than the device-level commands that are exposed by the UR3e robot itself. Nonetheless, the UR3e dataset (Table 7) resembles the N9 dataset in terms of the basic actuation APIs and is therefore equally useful. For convenience, the UR3e APIs use both cartesian and polar coordinate systems. Each *frame* or *location* in the space is therefore represented using two tuples: (x, y, z) and (rx, ry, rz), i.e., , six floats in total. This representation is used for denoting both joint positions and tool center point (TCP). RATracer's data processing filters out all connection related parameters that happen to be a part of the argument list for many methods in *class UR3Arm(RobotArm)*.

2 Power Dataset

Table 8 shows a list of physical properties that are collected through the UR3 robot arm for the power monitoring data.

Name	**Description**
target_q	Target joint positions
target_qd	Target joint velocities
target_qdd	Target joint accelerations
target_current	Target joint currents
target_moment	Target joint moments
target_TCP_pose	Target coordinates
target_TCP_speed	Target speed
actual_execution_time	thread execution time
actual_tool_accelerometer	X, Y, Z accelerometer values
speed_scaling	Speed scaling of trajectory limiter

Table 8: Power Monitoring Features