UC5 (robot) - OPC UA tests

Brief description of the robot HMI

- The robot control system (including HMI) is running on *industrial PC B&R* (the same HW as in the case of the real robot model - only the real robot prototype is not connected and only virtual model of the robot is available via HMI visualization)
- The robot control system and HMI was configured for *specific Non Destructive Testing (NDT) application* (one example of the teached NDT probe trajectory)
- The robot HMI was reduced for "operator" user (task reduction especially for NDT, more service functions are restricted):
  - available on public IP address via web browser[1]: **147.228.47.212**
  - login: **operator**
  - password: **charmuc5**

Note: If the HMI website is not displayed correctly please refresh it (F5, ctrl+F5)!
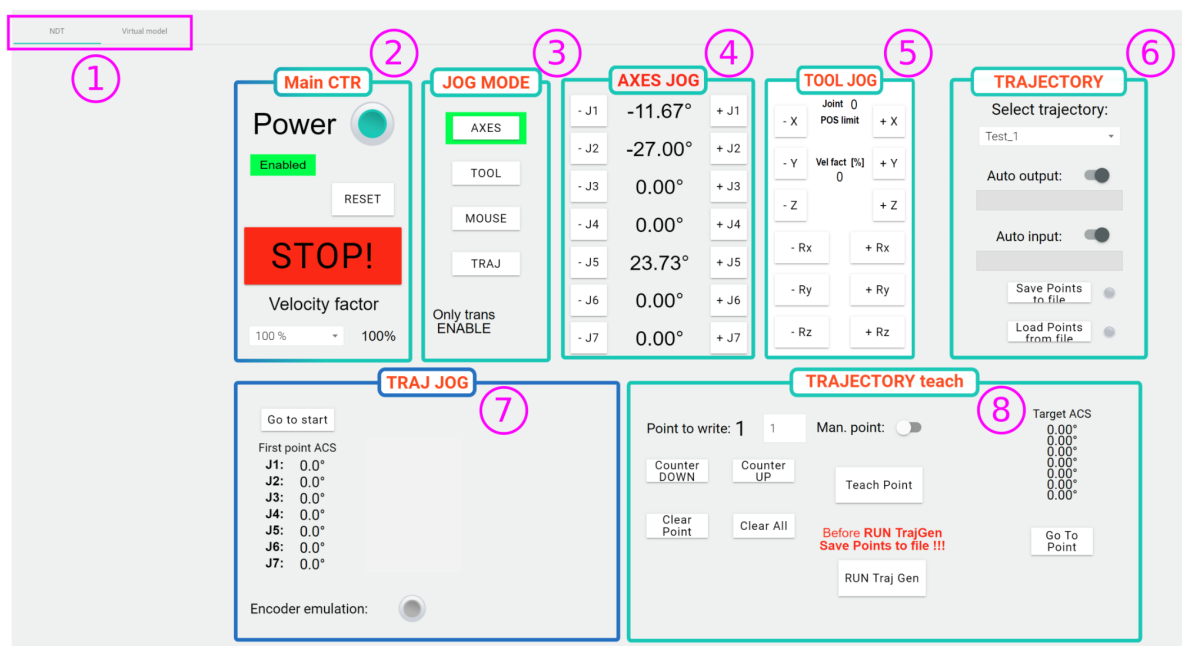

Fig 1: HMI (control panel) of the robot (NDT operator mode)

1. Switch between *Control panel* (NDT) and *Robot visualization panel* (Virtual model), the website can be duplicated in the browser to see both of them, see Fig. 2.
2. Main robot control
3. Robot JOG mode selection:
   a. AXES - robot is moving in the individually axes
   b. TOOL - robot is moving according to NDT probe (tool) frame (coordinated motion)

---

[1] Google Chrome was used for testing.

<ol type="a" start="3">
MOUSE - robot is controlled by 3D mouse sensor => hand-guidance (not relevant for virtual robot model)
TRAJ - robot is moving along teached NDT probe trajectory (relevant for the following example, see bellow)

<ol start="4">
AXES JOG - robot axes control
TOOL JOG - robot coordinated (TOOL) control, joint POS limit and robot singularity occurrence are indicated
TRAJECTORY - trajectory selector for teaching trajectory (output to the trajectory generator) and movement along the trajectory in auto mode (input from trajectory generator)[2]
TRAJ JOG - control of the movement of the NDT probe (auto, manual jog) along teached trajectory (relevant for the following example, see bellow)
TRAJECTORY teach - control panel for teaching the NDT probe trajectory by an operator especially through 3D mouse sensor (rough teaching, final correction, etc.) - not relevant for the following example
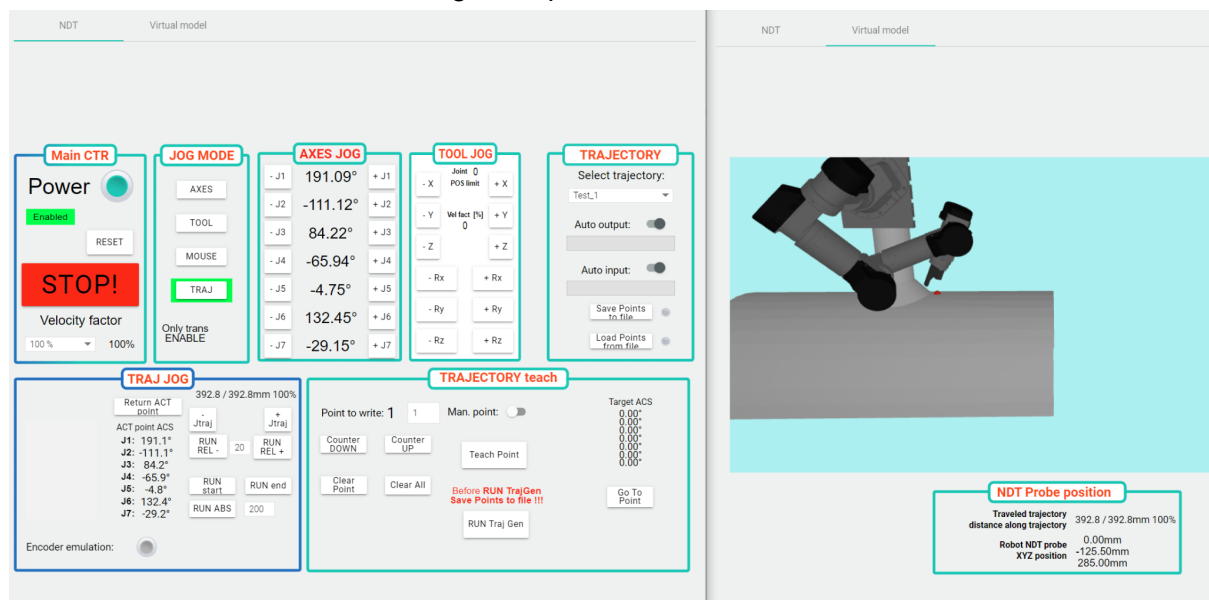

Fig. 2: Both panels (*Control panel* and *Robot visualization panel*)

---

[2] Trajectory generator is implemented as a Matlab script which communicates with the robot control system periodically via REST and FTP protocol. Matlab script has to run on HOST PC.
Trajectory generator itself is not relevant for the OPC UA test. An example of one trajectory has been generated (as *Test_1* example, see *Selected trajectory* menu).

Example of the robot auto movement along teached trajectory

The following example has been selected for the OPC UA interface test (more details are introduced in the following chapter).

The next steps describe the test example:

1. Main CTR: *RESET* errors and turn on the robot (*Power)* => the robot state changes to *Enabled*
2. TRAJECTORY: Selected (teached) trajectory is *Test_1* (*Auto input* has to be enabled)
3. TRAJ JOG: Load the robot 1st trajectory point (in ACS[3]) - push the *Go to start* button => robot is ready to run to the 1st point
4. TRAJ JOG: Push the *Go to start* button again, the robot starts to move to the 1st trajectory point
   Note: The view of the 3D visualization can be adjusted via mouse.
5. JOG MODE: Switch the JOG MODE to *TRAJ* => trajectory control buttons appear in TRAJ JOG
6. TRAJ JOG: The robot movement along the trajectory can be controlled in various ways:
   a. *-Jtrans, +Jtrans*: Manual JOG along trajectory
   b. *RUN REL -, RUN REL +*: Relative robot movement along trajectory
   c. *RUN start, RUN end*: Robot moves to the start and to the end of the teached trajectory
   d. *RUN ABS*: Robot moves to the defined position along trajectory

Note: The robot overall speed can be adjusted via *Velocity factor* menu (Main CTR).
Note: The button *Return ACT point* returns the robot to the actual position along the. trajectory, e.g. after the robot has been moved via *AXES*/*TOOL* JOG MODE out of the trajectory.
Note: All numeric/text values which are inserted have to be confirmed by "Enter".

[Example of robot control (video)](#)

---

[3] ACS - Axes Coordinate System

OPC UA configuration (data interface)

For OPC UA test the robot control system implements O*PC UA Server Endpoint* with the following settings:

- **Endpoint URL:** opc.tcp://147.228.47.212:4885
- **SecurityPolicy:** Basic256Sha256
- **MessageSecurityMode:** Sign&Encrypt
- **CredentialsAuthentication:** observer:charmuc5

We need to add you as user :
- **Your certificate (public key)**

All data are under Namespace: 4 = urn:Rex:Exec:ZCU:CHARM:Robot:RexCore

The data under consideration is under **Namespace 4** as follows:

1. **Robot NDT probe position** (center of universal joint of the NDT probe holder - red dot in Fig. 3)

Identifier	Meaning	Unit
$.main.VTOR_NDTprobeXYZposition.y1	**X position** of the center of the NDT probe holder with respect to the reference frame depicted in Fig. 3	m
$.main.VTOR_NDTprobeXYZposition.y2	**Y position** of the center of the NDT probe holder with respect to the reference frame depicted in Fig. 3	m
$.main.VTOR_NDTprobeXYZposition.y3	**Z position** of the center of the NDT probe holder with respect to the reference frame depicted in Fig. 3	m

2. **The distance traveled along NDT probe trajectory**

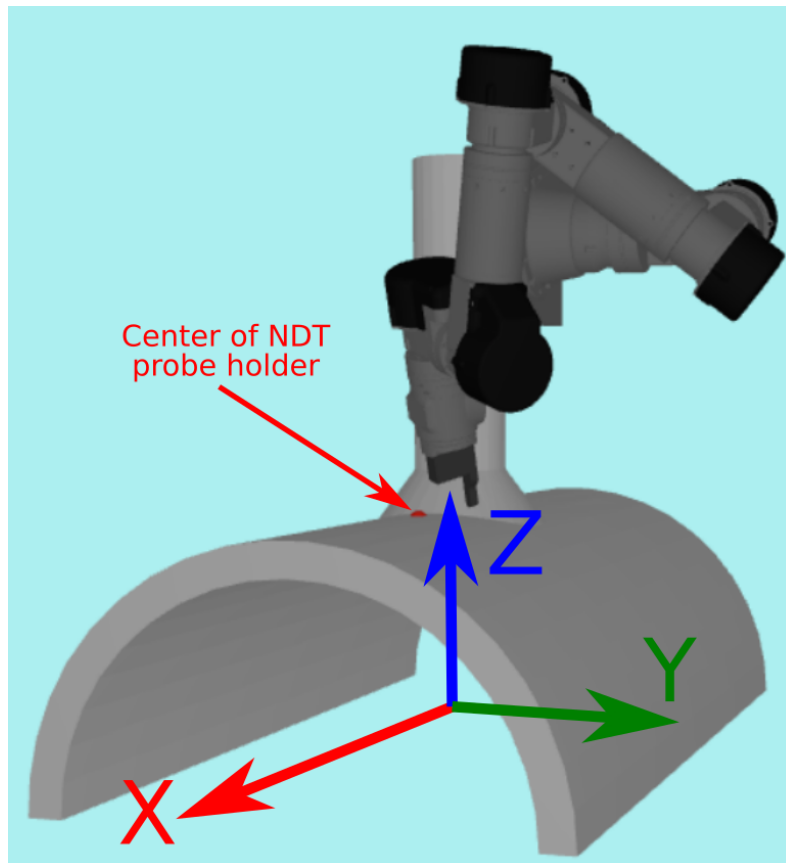| Identifier	Meaning	Unit
$.main.DIV_AxisS_actPathState.u1	Absolute instantaneous distance traveled by the NDT probe along the trajectory	mm
$.main.DIV_AxisS_actPathState.u2	Length of the trajectory (max. distance)	mm
$.main.DIV_AxisS_actPathState.y	Relative instantaneous distance traveled by the NDT probe along the trajectory	0..1

Fig. 3: NDT probe position and corresponding reference frame

UaExpert was used for testing of OPC UA settings and the configuration file *.uap* can be downloaded here.