XHand Control ROS Documentation

1.1.1	2024.10.25

CONTENTS

Release Notes

releases	dates	edit a record
1.1.0	2024.10.25	first edition
1.1.1	2024.10.25	Updating the ros package

Instructions for use

present (sb for a job etc)

XHandControlROS is a ROS package for EtherCAT communication with XHand robots, controlling their joints and reading sensor data. The package provides a full control interface to the XHand through ROS topics, services and callback functions.

Download:

xhand_control_ros_v1.1.1_20241025_release.zip

XHandControlROS Instructions for Use

present (sb for a job etc)

XHandControlROS is a ROS package for EtherCAT communication with XHand robots, controlling their joints and reading sensor data. The package provides a full control interface to the XHand through ROS topics, services and callback functions.

compiling

Ensure that the following dependencies have been installed:

- ROS (supports ROS noetic)

```Bash
 sudo apt install libcrypto++-dev
```

Creating a workspace

```Bash
 mkdir -p ~/catkin_ws/src
```

will ros package to your catkin_ws/src workspace and compile it:

```Bash
 cd ~/catkin_ws
 catkin_make
source devel/setup.bash
```

nodal

xhand_control_ros node

This node is responsible for establishing a connection with the XHand device and processing the hand's status and commands.

boot node

You can start the node with `the root user` by running the following command:

```Bash
 source devel/setup.bash
 roslaunch xhand_control_ros xhand_control.launch
```

Alternatively, you can start the node with a `normal user` by running the following command:

- Add permissions to the executable after each compilation of the code:

```Bash
 sudo chown root:root
```

```
devel/lib/xhand_control_ros/xhand_control_ros_node
 sudo chmod u+s devel/lib/xhand_control_ros/xhand_control_ros_node
```

- Then run the node:

```
Bash
 source devel/setup.bash
 roslaunch xhand_control_ros xhand_control.launch
```

subject (of a talk or conversation)

Posted topics

- `/xhand_control/xhand_state`: Publish the joint state and sensor data of XHand, refer to the joint information of the hand Joint information
    - Message type: `xhand_control_ros::XHandStateArray`
    - XHandStateArray The content of the message:
        - `header`: the message header.
        - `hand_id[]`: hand ID.
        - `hand_name[]`: hand name.
        - `hand_states[]`: joint states for each hand.
            - `name[]`: the name of the joint.
            - `position[]`: position of the joint.
            - `effort[]`: joint moment.
            - `temperature[]`: temperature of the joint.
            - `error_code[]`: joint error code.
        - `sensor_states[]`: sensor states for each hand.
            - `finger_sensor_states[]`: sensor states for individual hands.
                - `location`: Sensor location.
                - `calc_force[]`: combined sensor force
                - `raw_force[]`: sensor raw force data.
                - `calc_temperature[]`: sensor torque.
                - `raw_temperature[]`: sensor raw torque data.

View Sample Message

```
Bash
 rostopic echo /xhand_control/xhand_state
```

Subscribed topics

- `/xhand_control/xhand_command`: subscribe to XHand control commands sent by the user.
    - Message type: `xhand_control_ros::XHandCommand`
    - XHandCommand The content of the message:
        - `hand_id`: specifies the ID of the hand to be controlled.
        - `name[]`: Specifies a list of joint names to be controlled.
        - `position[]`: target position of each joint.
        - `kp[]`, `kd[]`, `ki[]`: PID control parameters.
        - `effort_limit[]`: Torque limit.
        - `mode`: control mode.

Example of posting a message

```
Bash
 rostopic pub -1 /xhand_control/xhand_command
xhand_control_ros/XHandCommand '{hand_id: 0, name:
["thumb_bend_joint", "index_bend_joint"], position: [0.5, 0.18],
kp: [100, 100], kd: [0, 0], ki: [0, 0], effort_limit: [350, 350],
mode: 3}'
```

service

Services provided

- `/xhand_control/read_hand_info`: read XHand information.
    - Service Type: `xhand_control_ros::ReadHandInfo`
    - Request Fields:
        - `hand_id`: the ID of the hand to be queried.
        - `info_type`: type of query (e.g. hand type, serial number, version, etc.).
    - Response Fields:

        - success: whether the operation was successful.
        - `info`: information about the returned hand.

- `/xhand_control/set_hand_id`: set the ID of XHand.
    - Service Type: `xhand_control_ros::SetHandId`
    - Request Fields:
        - `current_id`: current hand ID.
        - `new_id`: the new ID to be set.
    - Response Fields:
        - success: whether the operation is successful or not, need to restart the node after success.

- `/xhand_control/set_hand_name`: Set the name of the XHand.
    - Service Type: `xhand_control_ros::SetHandName`
    - Request Fields:
        - `hand_id`: ID of the hand to be set.
        - `hand_name`: the new name to be set.
    - Response Fields:
        - success: whether the operation is successful or not, need to restart the node after success.

- `/xhand_control/reset_hand_sensor`: reset XHand's sensor.
    - Service type: `xhand_control_ros::ResetSensor`
    - Request Fields:
        - `hand_id`: ID of the hand to be reset.
        - `sensor_id`: The sensor ID (17 to 21) to be reset.
    - Response Fields:
        - success: whether the operation was successful.

Service Call Example

```Bash
 rosservice call /xhand_control/read_hand_info '{hand_id: 0,
info_type: 0}'
 rosservice call /xhand_control/set_hand_name '{hand_id: 0,
hand_name: 'test_hand'}'
 rosservice call /xhand_control/set_hand_id '{current_id: 1,
new_id: 2}'
 rosservice call /xhand_control/reset_hand_sensor '{hand_id: 0,
sensor_id: 17}'
```

Joint Names and Limits

joint name	lower limit of position	Position Limit
thumb_bend_joint	0	1.57
thumb_rota_joint1	-1.05	1.57
thumb_rota_joint2	0	1.57
index_bend_joint	-0.09	0.3
index_joint1	0	1.92
index_joint2	0	1.92
mid_joint1	0	1.92
mid_joint2	0	1.92
ring_joint1	0	1.92
ring_joint2	0	1.92
pinky_joint1	0	1.92
pinky_joint2	0	1.92

common problems

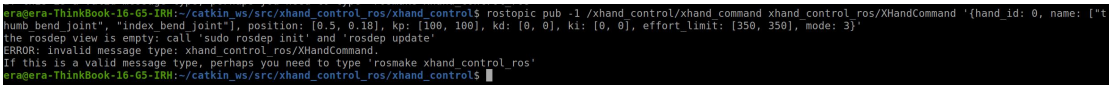
1. Unable to detect device

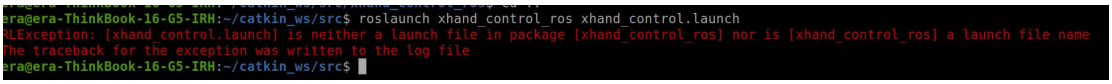
Make sure that the EtherCAT device is connected properly and that the interface name is configured correctly.

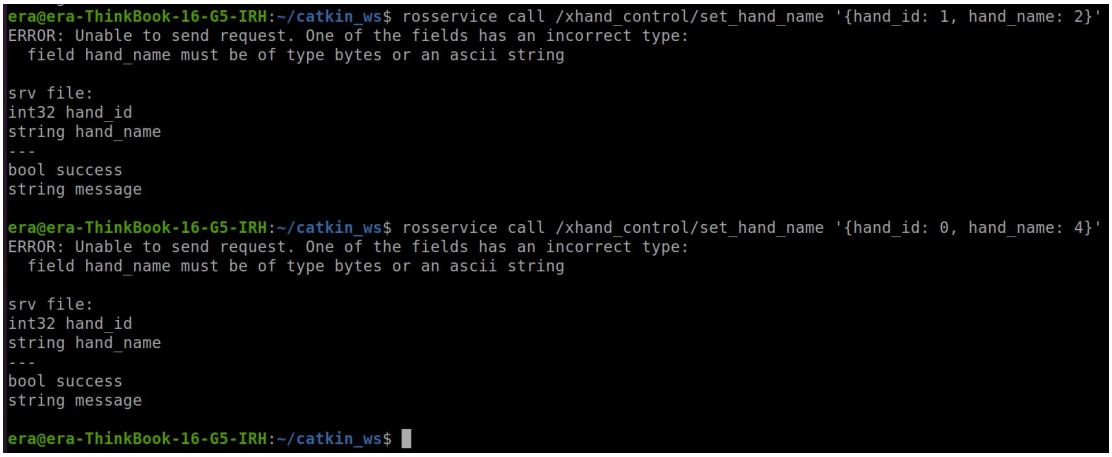
2. Unable to post or subscribe to messages

Make sure you have started the node correctly and that the topic naming is consistent with the actual subscription or publication.

3. No environment configured

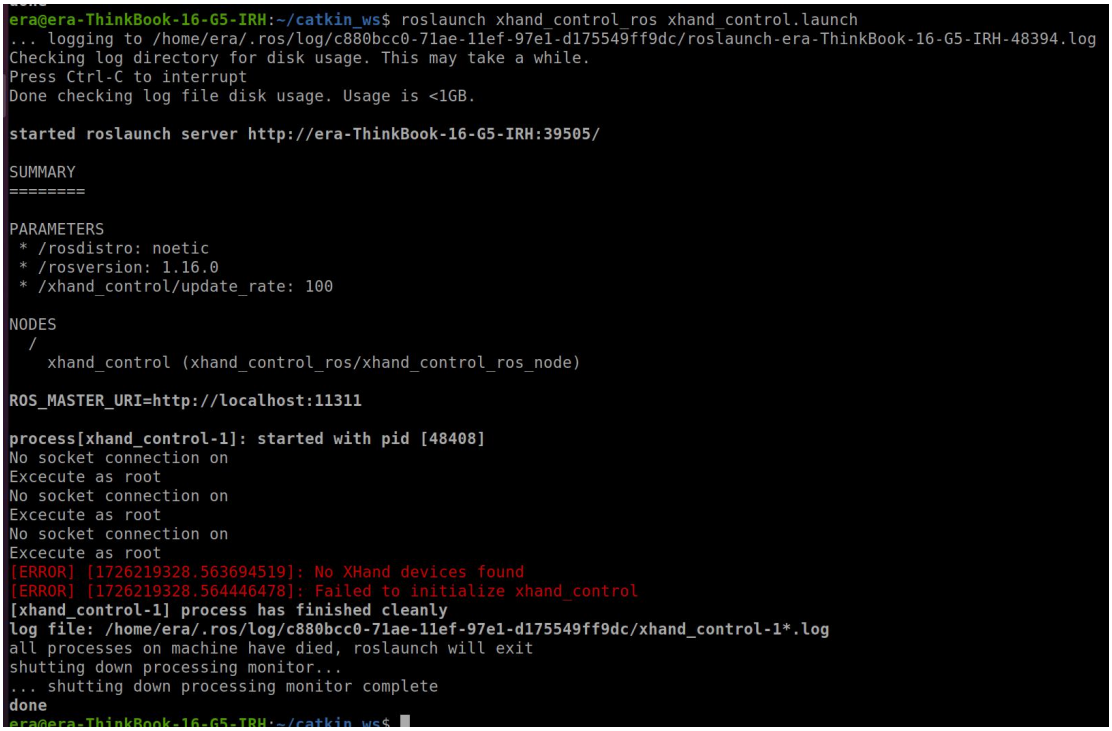






Need to execute `source devel/setup.bash` in the workspace

4. insufficient authority



Root privileges are required, or refer to Startup node (normal user)

5. How about switching between zero torque mode and PD bit control mode via ROS?

Just change mode to 0 for zero torque mode, and 3 for pd bit control.

6. What will happen if the effort limit is given to exceed the maximum force out of the joint?

Limit to the maximum value, input value support 0-400, more than 400 value according to 400 processing

7. Meaning of control mode mode in xhand_control/xhand_command?

Mode A value of 0 is zero torque mode, a value of 3 is torque mode.