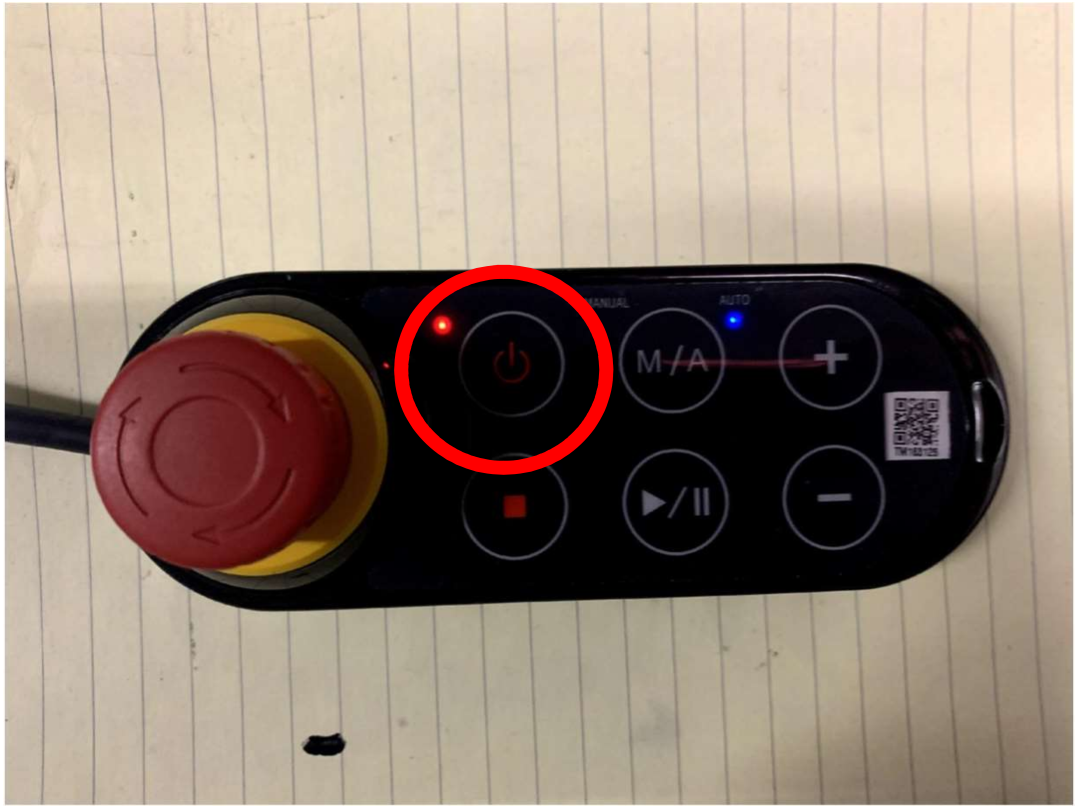
HW4 Tutorial

Hao-Yun Chen

2020/11/23

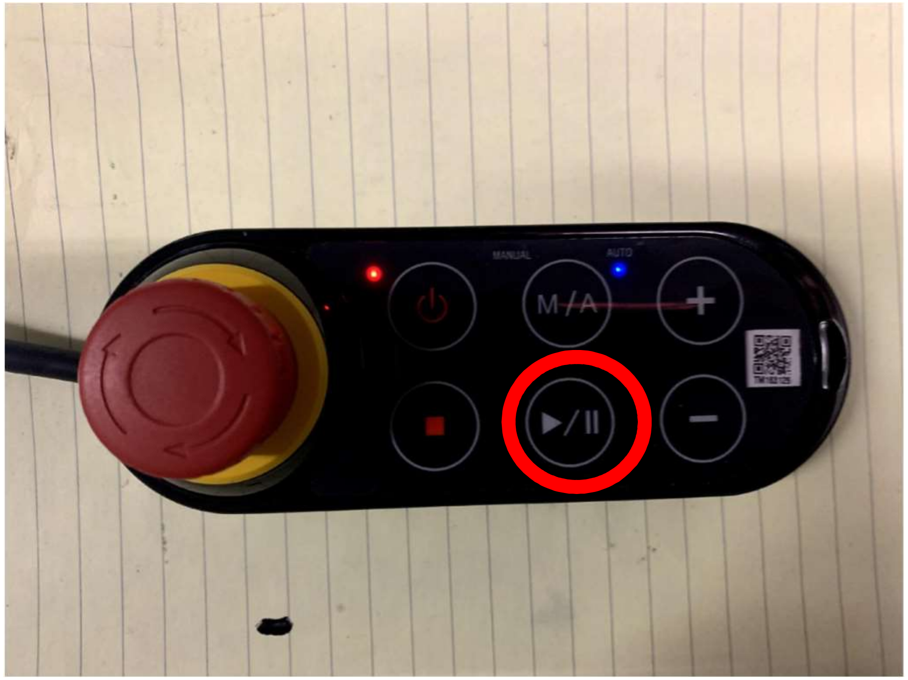
Server (Robot Arm)

Step 1 Turn the robot arm on

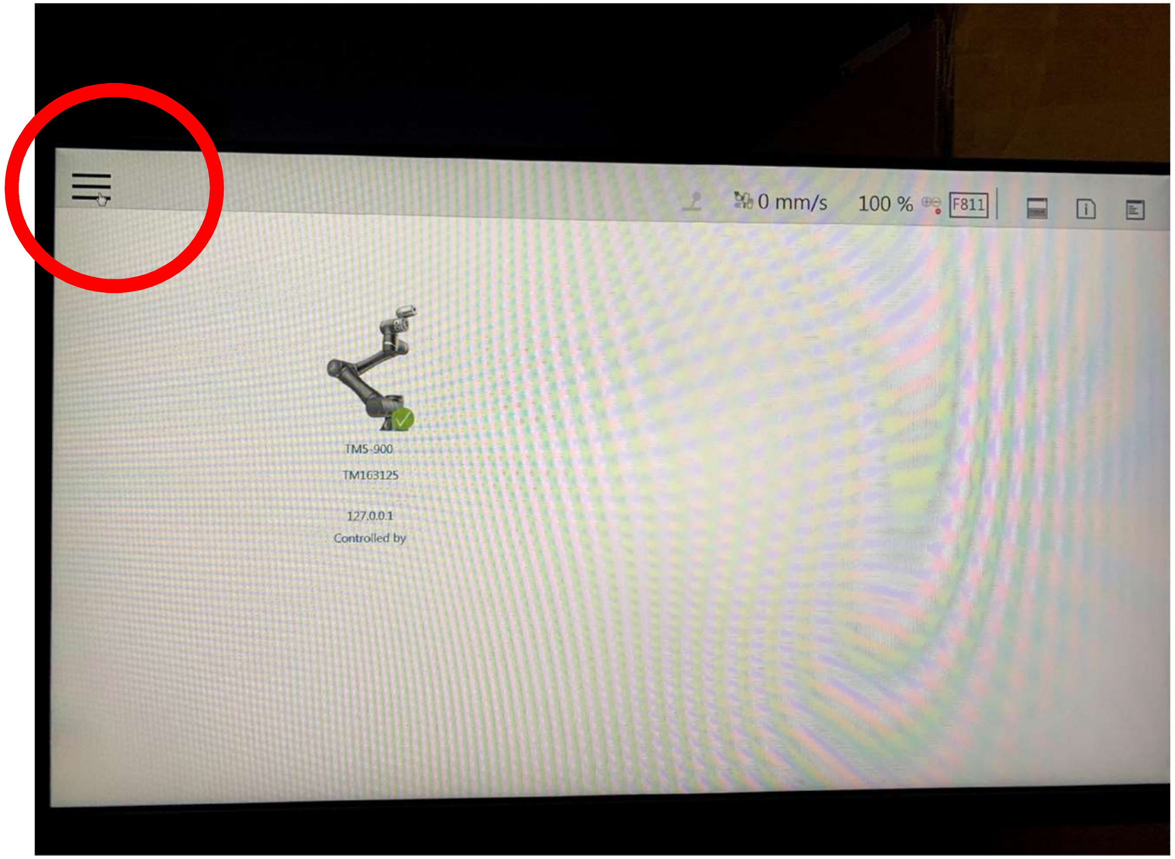


Step 2 Directly Run the Project

Press the play button directly, if everything work well, you should be able to hear the beep sound and the robot's end effector's light turn blue, then you can go to the "Client (Computer)" part in page 14, otherwise, continue to the next page's instruction.

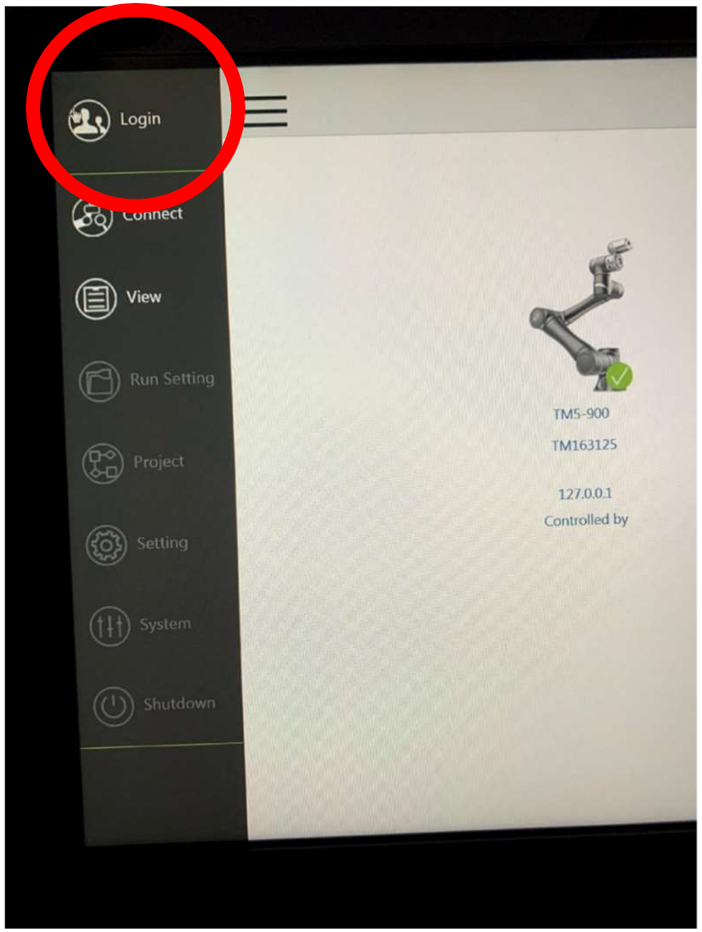


Step 2 Login and get control



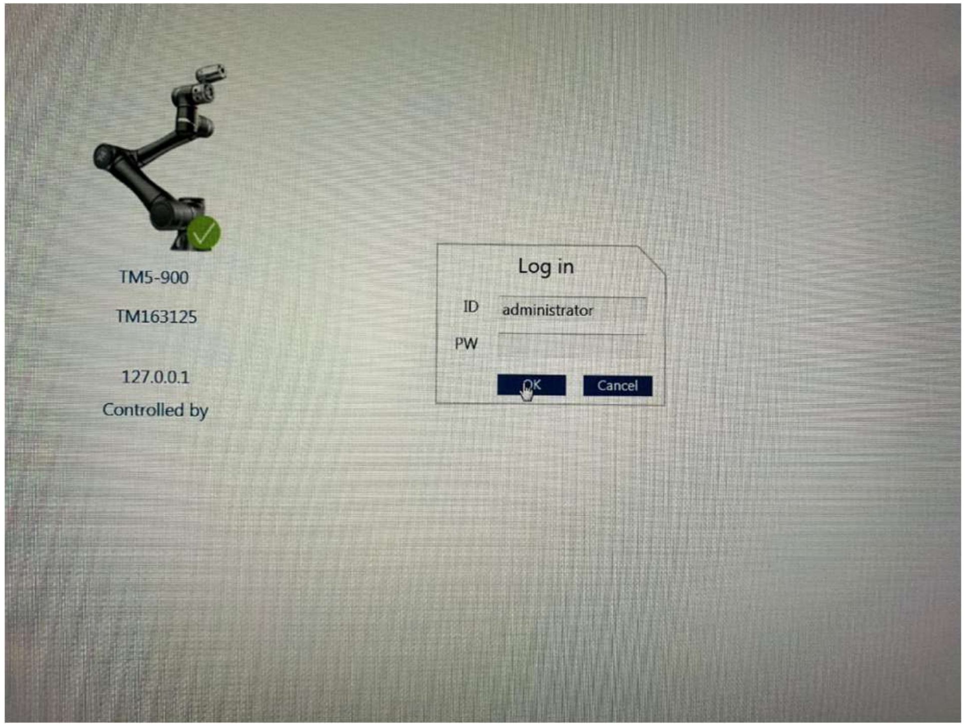
Step 2 Login and get control

- Click the login icon



Step 2 Login and get control

- No need to enter password, just click ok



TM5-900

TM163125

127.0.0.1

Controlled by

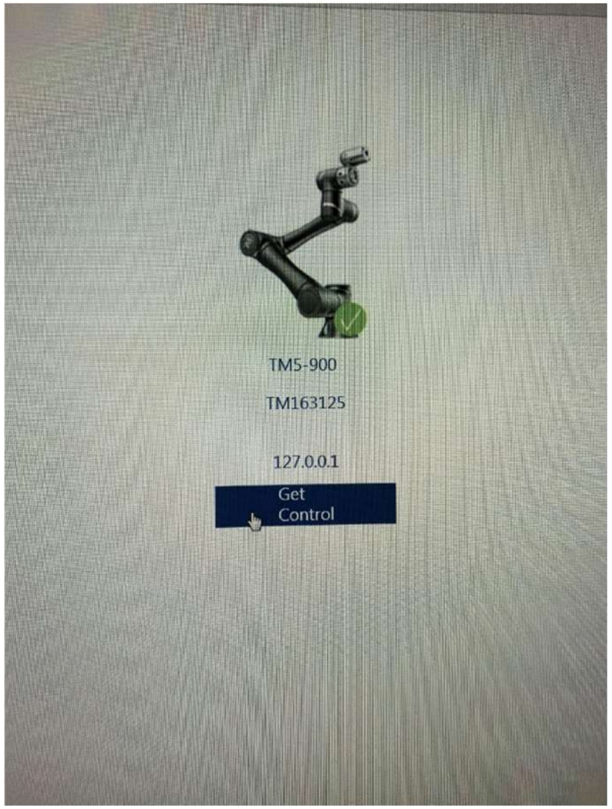
Log in

ID administrator

PW

Step 2 Login and get control

- The “Get Control” button should appear, click it.

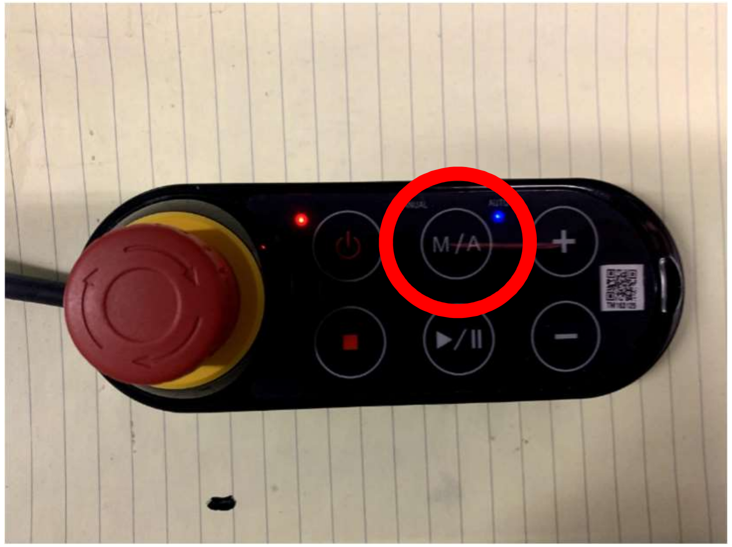


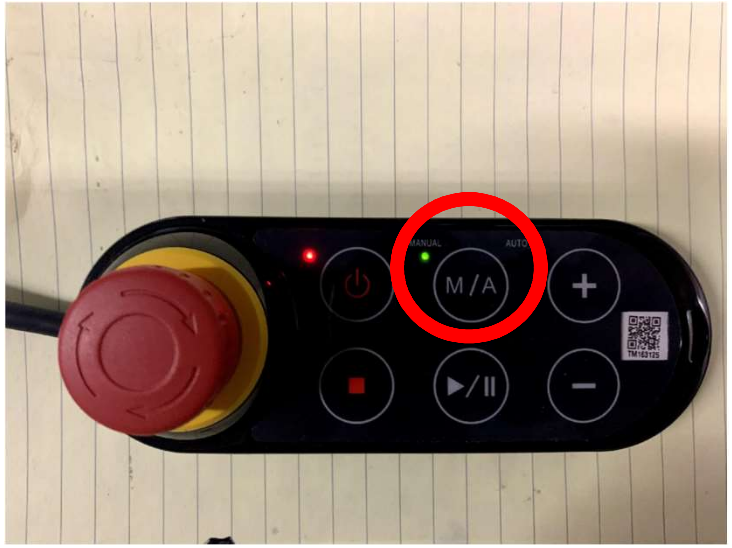
TM5-900

TM163125

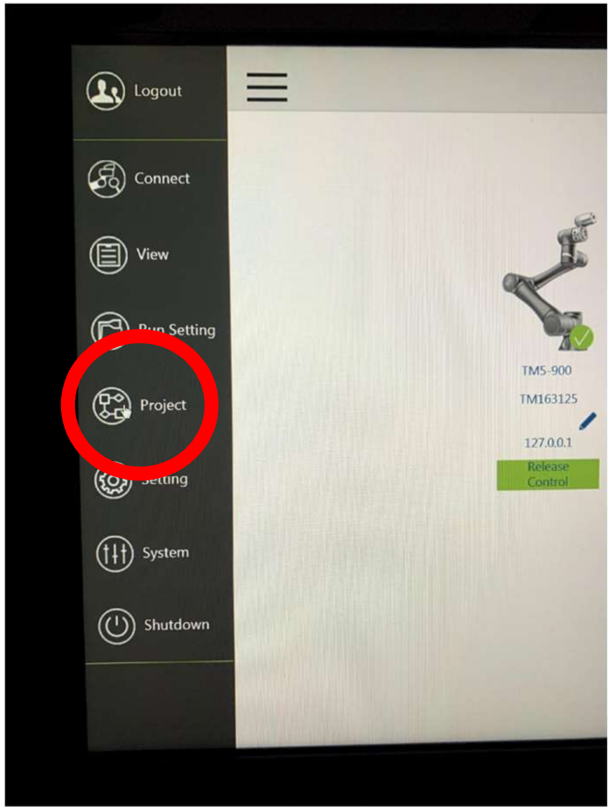
127.0.0.1

Step 3 Change to manual model

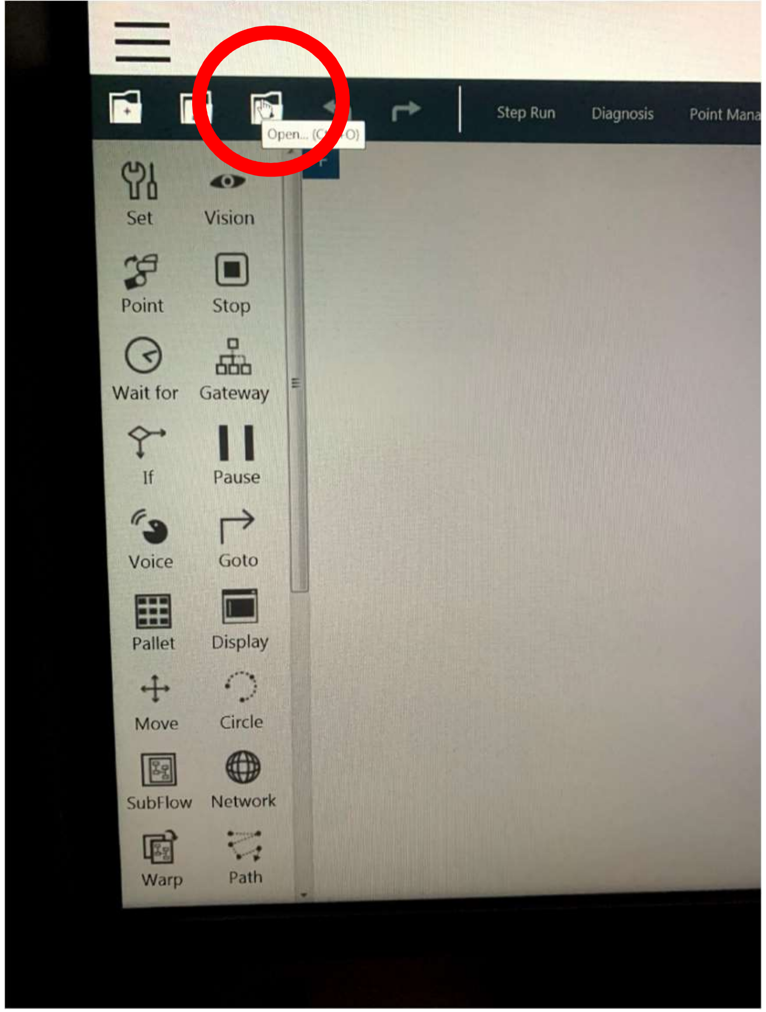




Step 4 Open the listener project

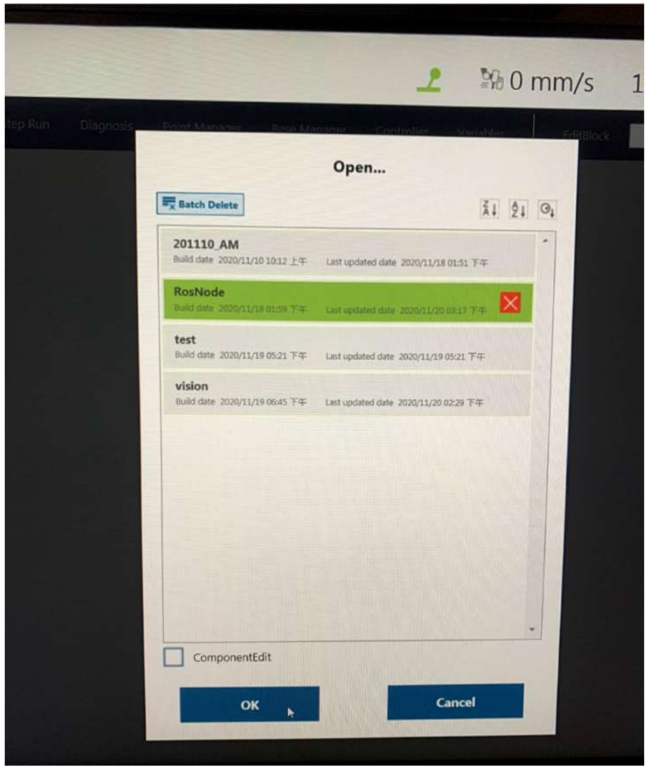


Step 4 Open the listener project



Step 4 Open the listener project

- Choose project “RosNode”



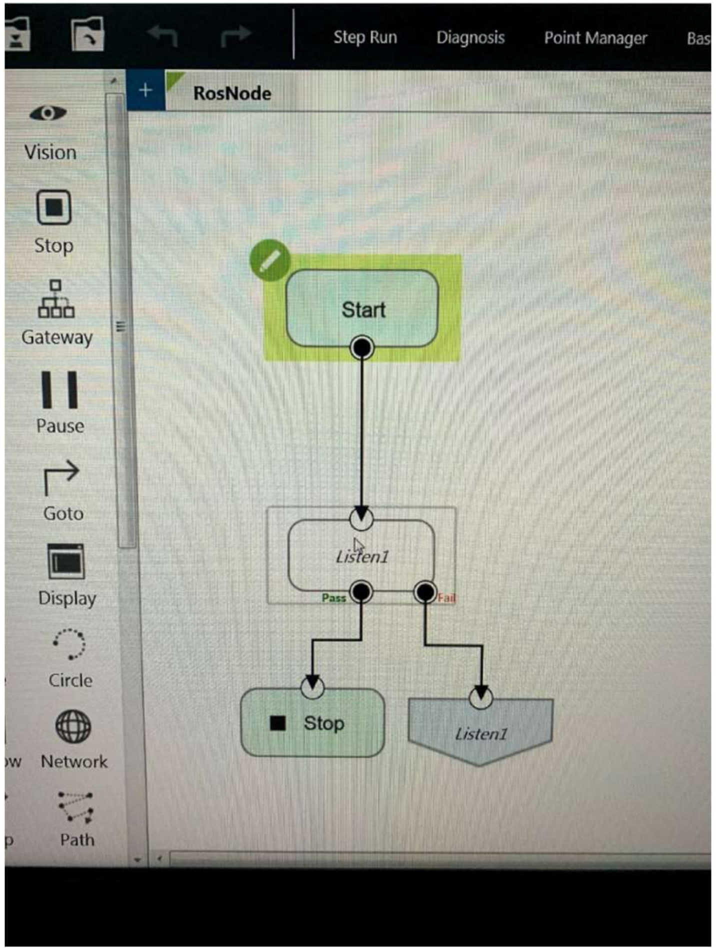
201110_AM
Build date 2020/11/10 10:12 上午 Last updated date 2020/11/18 01:51 下午

RosNode
Build date 2020/11/18 01:59 下午 Last updated date 2020/11/20 03:17 下午

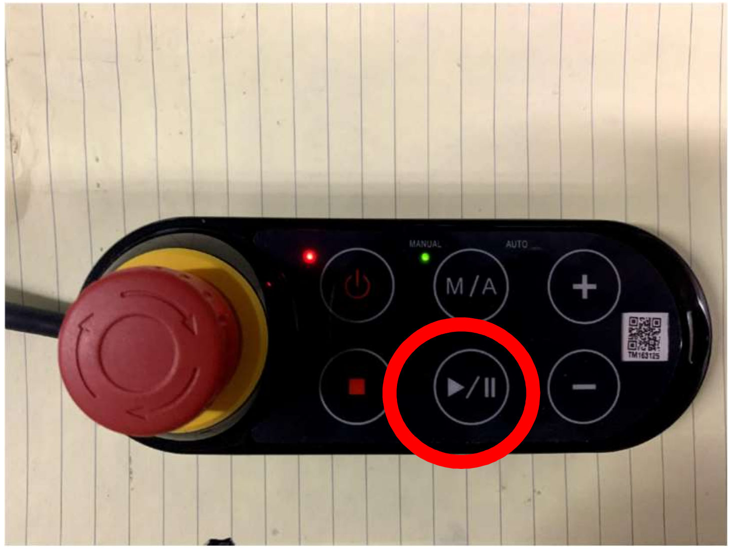
test
Build date 2020/11/19 05:21 下午 Last updated date 2020/11/19 05:21 下午

vision
Build date 2020/11/19 06:45 下午 Last updated date 2020/11/20 02:29 下午

☐ ComponentEdit



Step 5 Run the listener project



Client (Computer)

Step 1 Create your team’s own workspace

- Create your ROS workspace

```
$ mkdir -p ~/workspaces/teamX_ws/src
$ cd ~/workspaces/teamX_ws
```

- You can still compile with empty workspace

```
$ catkin_make
```

  You should have a 'build' and 'devel' folder under 'teamX_ws’

Step 2 Create your team’s own package

- Create a package

```
$ cd ~/workspaces/teamX_ws/src
$ catkin_create_pkg <package_name> std_msgs rospy roscpp
```

- Compile

```
$ cd ~/workspaces/teamX_ws
$ catkin_make
```

- Source your environment

```
$ source devel/setup.bash
```

- Enter your package folder

```
$ roscd <package_name>
```

Step 3 Create Your Team's Own Python Environment

- Create your Pyenv Environment

```
$ pyenv virtualenv 2.7.17 teamX
```

- Set this environment as your workspace's default python environment

```
$ cd ~/workspaces/teamX_ws
$ pyenv local teamX
```

- Install the necessary package

```
$ cp ~/workspaces/TAs_ws/requirements.txt .
$ pip install –r requirements.txt
```

- IMPORTANT: Do NOT mess up with either system's or other group's python environment

Step 4 Get the Python template

- Copy the template to your team’s workspace

```
$ mkdir  ~/workspaces/teamX_ws/src/<package_name>/scripts
$ cp ~/workspaces/TAs_ws/src/send_script/scripts/send_script.py
~/workspaces/ teamX_ws/src/<package_name>/scripts
```

- Compile your files

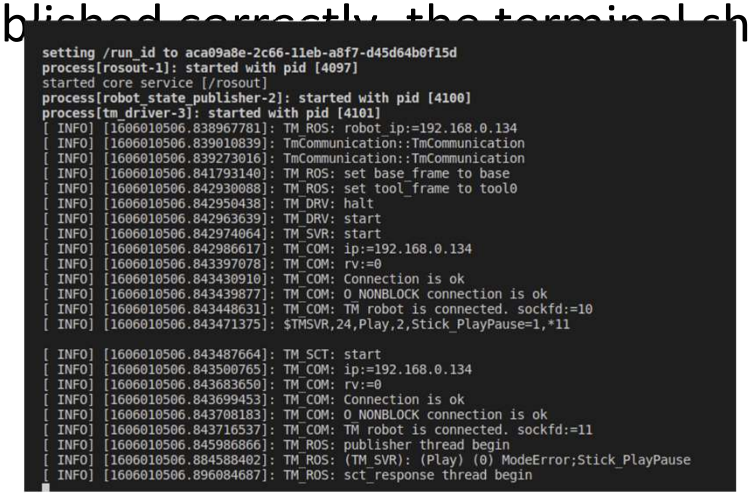
```
$ cd ~/workspaces/teamX_ws && catkin_make
```

- Source your environment

```
$ source devel/setup.bash
```

Step 4 Run the robot arm’s driver

- Run the driver
  ```
  $ cd ~/catkin_ws
  $ pyenv shell system #use the system environment to run driver
  $ source ~/catkin_ws/devel/setup.bash
  $ roslaunch tm_driver tm5_900_bringup.launch robot_ip:=<robot_ip>
  ```
  IP of the arm close to the door is 192.168.0.134, the other is 192.168.0.119
- If the connection established correctly, the terminal should look like the following figure

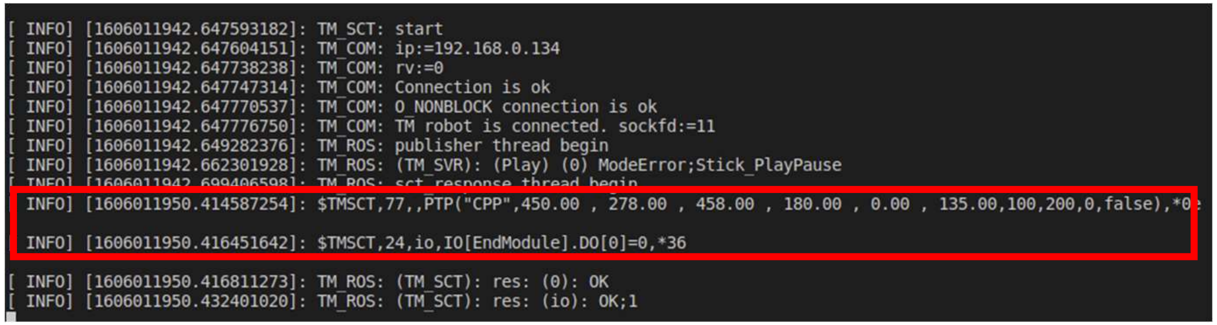


```
setting /run_id to aca09a8e-2c66-11eb-a8f7-d45d64b0f15d
process[rosout-1]: started with pid [4097]
started core service [/rosout]
process[robot_state_publisher-2]: started with pid [4100]
process[tm_driver-3]: started with pid [4101]
[ INFO] [1606010506.838967781]: TM_ROS: robot_ip:=192.168.0.134
[ INFO] [1606010506.839010839]: TmCommunication::TmCommunication
[ INFO] [1606010506.839273016]: TmCommunication::TmCommunication
[ INFO] [1606010506.841793140]: TM_ROS: set base_frame to base
[ INFO] [1606010506.842930088]: TM_ROS: set tool_frame to tool0
[ INFO] [1606010506.842950438]: TM_DRV: halt
[ INFO] [1606010506.842963639]: TM_DRV: start
[ INFO] [1606010506.842974064]: TM_SVR: start
[ INFO] [1606010506.842986617]: TM_COM: ip:=192.168.0.134
[ INFO] [1606010506.843397078]: TM_COM: rv:=0
[ INFO] [1606010506.843430910]: TM_COM: Connection is ok
[ INFO] [1606010506.843439877]: TM_COM: O_NONBLOCK connection is ok
[ INFO] [1606010506.843448631]: TM_COM: TM robot is connected. sockfd:=10
[ INFO] [1606010506.843471375]: $TMSVR,24,Play,2,Stick_PlayPause=1,*11

[ INFO] [1606010506.843487664]: TM_SCT: start
[ INFO] [1606010506.843500765]: TM_COM: ip:=192.168.0.134
[ INFO] [1606010506.843683650]: TM_COM: rv:=0
[ INFO] [1606010506.843699453]: TM_COM: Connection is ok
[ INFO] [1606010506.843708183]: TM_COM: O_NONBLOCK connection is ok
[ INFO] [1606010506.843716537]: TM_COM: TM robot is connected. sockfd:=11
[ INFO] [1606010506.845986866]: TM_ROS: publisher thread begin
[ INFO] [1606010506.884588402]: TM_ROS: (TM_SVR): (Play) (0) ModeError;Stick_PlayPause
[ INFO] [1606010506.896084687]: TM_ROS: sct_response thread begin
```

Step 4 Run the Python template

- Source your environment
  ```
  $ source ~/workspaces/teamX_ws/devel/setup.bash
  ```
- Run the script
  ```
  $ rosrun <package_name> send_script.py
  ```
- The robot should move to the predefined pose and gripper will close
- You can see the command send from the computer in the driver terminal



```
[ INFO] [1606011942.647593182]: TM_SCT: start
[ INFO] [1606011942.647604151]: TM_COM: ip:=192.168.0.134
[ INFO] [1606011942.647738238]: TM_COM: rv:=0
[ INFO] [1606011942.647747314]: TM_COM: Connection is ok
[ INFO] [1606011942.647770537]: TM_COM: O_NONBLOCK connection is ok
[ INFO] [1606011942.647776750]: TM_COM: TM robot is connected. sockfd:=11
[ INFO] [1606011942.649282376]: TM_ROS: publisher thread begin
[ INFO] [1606011942.662301928]: TM_ROS: (TM_SVR): (Play) (0) ModeError;Stick_PlayPause
[ INFO] [1606011942.699406598]: TM_ROS: sct_response thread begin
  INFO] [1606011950.414587254]: $TMSCT,77,,PTP("CPP",450.00 , 278.00 , 458.00 , 180.00 , 0.00 , 135.00,100,200,0,false),*0

  INFO] [1606011950.416451642]: $TMSCT,24,io,IO[EndModule].DO[0]=0,*36

[ INFO] [1606011950.416811273]: TM_ROS: (TM_SCT): res: (0): OK
[ INFO] [1606011950.432401020]: TM_ROS: (TM_SCT): res: (io): OK;1
```