Control UR5 via ROS
environment setting up

Liujiang Yan, July 2 2016

1. Prerequisites

   In order to complete this tutorial, you should have access to the following:
   - A computer with ROS installed (using indigo in this tutorial)
   - ROS-Industrial's universal_robot package

2. Installing the neccesary software

   The pre-defined ur package like below is out-of date.

   ```
   $ sudo apt-get ros-indigo-universal-robot
   ```

   In order to get the up-to-date package from github, we need to do two things:

   a. creating a workspace for catkin

   Assumes that you have installed catkin and sourced your environment.

   First of all, create a catkin workspace

   ```
   $ mkdir -p ~/[your-space-name]/src
   $ cd ~/[your-space-name]/src
   $ catkin_init_workspace
   ```

   Secondly, build the workplace

   ```
   $ cd ~/[your-space-name]/
   $ catkin_make
   ```

   Sourcing the devel/setup.bash file will overlay this workspace on top of your enviorment. So before continuing this process as well as everytime to establish the control, you need source the setup.sh file like below

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

   b. clone the package from github

   Clone the main ur package from github

   ```
   $ git clone --branch indigo-devel
   https://github.com/ros-industrial/universal_robot/tree/indigo-devel
   /[your-space-name]
   ```

   Substitute the /[you-work-space]/src/universal_robot/ur_driver/src/ur_driver folder with following

   https://github.com/LiujiangYan/universal-robot-v32/tree/master/ur_driver

   cd the workspace, sourcing the setup.bash and catkin_make

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

3. Connecting the robot UR5

   The pc can connect the Universal Robot via Ethernet. In order to enable the connection, we need to setup the network via the teach-pendant through

   ```
   Setup Robot -> Setup Network Menu -> DHCP -> Apply -> Update
   ```

   Oncehaving the valid IP address, test the conncetion through ping command

   ```
   $ ping [your-robot-IPaddress]
   ```

   If there’s feedback and no error return, the conncection is set successfully.

4. Testing the robot

   cd to the workspace, make sure the connection is already established then,

```
        $ roscore
```

   start a new terminal

```
        $ roslaunch ur_bringup ur5_bringup.launch robot_ip:=[your-robot-IPaddress]
[reverse_port:=REVERSE_PORT]
```

   start another new terminal

```
        $ rosrun ur_driver test_move.py
```

5. De-Bug advice

   The controller version from the robot is up-to-date so that the existing software avaliable from github is no longer suitable and needs some debuging. Though we have done some debugging and successfully tested the control from ros, we have no way to guarantee the validity of the process and the software.

   Here we presents some debug advice. May they be helpful.

   cd the folder and python some files below, you may need to correct the IP address in the py files,

```
        $ cd /[you-work-space]/src/universal_robot/ur_driver/src/ur_driver
        $ python test_comm.py
        $ python test_RTcomm.py
```

   From the error returned, you may see the error towards the communication between PC and robot.

```
        $ roslaunch ur_bringup ur5_bringup.launch robot_ip:=[your-robot-IPaddress]
[reverse_port:=REVERSE_PORT]
```

   From the error returned, you may see the error towards the driver.py.

   Also, you may find the client-interface and related parameters from the following excel helpful.

   https://github.com/LiujiangYan/universal-robot-v32/blob/master/Client_Interface.xlsx