Connecting to the hardware in Robolab

Thorbjørn Mosekjær Iversen (thmi@mmmi.sdu.dk) and
Stefan-Daniel Suvei (stdasu@mmmi.sdu.dk)

March 1, 2017

Contents

1 Introduction

The following guide explains both how to connect to the pc's in Robolab using the provided virtual machine, and how to setup your own laptop with a native Ubuntu 16.04 installation. If you are using the virtual machine you will not have to install anything, however read through the guide anyway so you know how the virtual machine has been configured.

The network connection requires the manual definition of a number of IP-addresses. In order to avoid groups having to change the IP-addresses to their own configuration, we have chosen the following ip-addresses for the hardware. Please use these when manually configuring you own system.

```
127.0.0.1 localhost
192.168.100.50 laptop-VM
192.168.100.60 laptop-HOST
192.168.100.51 Cell1
192.168.100.52 Cell2
192.168.100.53 Cell3
192.168.100.3 UR1
192.168.100.2 UR2
192.168.100.4 UR3
192.168.100.1 laptop-UR5Simulator
192.168.100.10 WSG
```

The relevant lines of the above should be added to the file `/etc/hosts` in Ubuntu. This has already been done for the Robolab machines and the virtual machine.

The Robot cells are numbered as follows:

- Cell 1: Closet to the door
- Cell 2: In the middle;
- Cell 3: Closet to the windows

2 Configure network

First you need to configure the Ethernet connection on each computer connected to the network. To do this do the following:

- Turn off your wifi and plug the network cable into your laptop
  - The wifi sometime creates problems, so turn it off if you don't need it.
- Configure a new Ethernet connection
  - Go to *Network connections* and press *Add*, select *ethernet* and press *create*.
  - Give your new connection a name, fx. *myRoviConnection*
  - Select the *IPv4 Settings* tab and select *Method: Manual*. Press *Add* to manually set your ip-address. The Address must be called `192.168.100.X` where `X` is a number of your choice[1]. The *Netmask* must be set to `255.255.255.0` while the *Gateway* is left blank.
  - Press *Save...* to save the settings. Be aware that sometimes you need to delete blank address lines or click in a different text field to make the *Save...* button click-able.
- Disconnect from you current default Ethernet connection and connect using the newly configured network
- You can see if your computer uses the correct ip using the terminal commands `ifconfig` and `hostname -I`

[1] Each computer on the network must have a different **X** so make sure the Robolab machine, your laptop, the robot, and your virtual machine (if you have one) use a different **X**. Preferably use the ip-addresses written in the introduction.

3 Using a virtual machine

On Black Board(section *Project*) you will find a guide on how to download the Virtual Machine. The Virtual Machine was tested with VMware Workstation 12 Player. Before the virtual machine is launched, configure the network settings in VMware so that the *Network Adapter* is set to *Bridged: Connected directly to the physical network* and set a check mark in *Replicate physical network connection state*. This will allow you to connect to the hardware. It is important that the WiFi is disabled OR that you connect to the network created in 2. If you need internet, you should set the *Network Adapter* to *NAT* - but this will not allow you to connect to the hardware (grippers, robot, etc.).

The virtual machine's network has been configured as described in 2. The IP used for this connection is `192.168.100.50`. Make sure that it is different from the one used for the host (your native OS).

4 Setup ROS network connection to the Robolab machines

Communicating over a wired network in ROS is a two step process. First the network must be configured such that the Robolab machine and your laptop can ping each other. Second ROS must be configured to support the network connection. The following assumes you are using a native installation `Ubuntu 16.04` or the virtual machine with networking of the virtual machine set to "bridged".

It should be noted that the Robolab machines are already configured correctly. It is however a good idea to check that they are setup according to the guide if you encounter connection problems. Another group might have been messing around with the settings.

From now on we will refer to the manually set ip-addresses of the Robolab machine and your laptop as `<robolab>` and `<laptop>`. To check that the connection is properly configured, run:

`ping <laptop>` from the Robolab machine
`ping <robolab>` from your laptop.

Make sure you can ping both ways before continuing this guide.

If ping-ing is succesfull, you will get messages like:

```
64 bytes from <laptop>/<robolab>: icmp_seq=1 ttl=64 time=0.027 ms.
```

4.1 Configure ROS for the network

To launch nodes on the Robolab PC using your own laptop, you first have to configure ROS for the network communication. Do the following:

- Open a new terminal and start a `roscore` on the Robolab machine. The terminal will output something similar to[2]:
  `ROS_MASTER_URI=http://rovi2-pc:11311/`
- In this case *rovi2-pc* is the name the Robolab computer uses to reference itself, while 11311 is the port where the communication is taking place. However *rovi2-pc* is the local ip-address so each terminal on computers in the network have to specify the real 192.168.100.X ip-address corresponding to `<robolab>`. So on each computer in the network (including the Robolab computer), run the following command in a terminal:

  `export ROS_MASTER_URI=http://<robolab>:11311`
  (remember `<robolab>` referes to the correct 192.168.100.X ip-adress).
- Each computer on the network also has to tell ROS it's own ip-address:

[2]When using roslaunch a roscore is automatically initiated. However the ROS_MASER_URI will still be printed to the terminal output

`export ROS_IP=<robolab>` from the Robolab machine
`export ROS_IP=<laptop>` from your laptop

- Note that this must be done for **every terminal** that you use to run a ROS command. If you are unable to connect to the master it likely that you opened a new terminal and forgot to set *ROS_MASTER_URI* and *ROS_IP* You can check if you have set the values correctly by running:

  `echo $ROS_MASTER_URI`
  and
  `echo $ROS_IP`

All the computers on the network should now be able to communicate with the Robolab roscore. You can check this by connecting a vision sensor to the Robolab computer and running the appropriate launch file. Your laptop should now be able to see and grab data from the published topics.

If you can see the topics, but no messages are published on them, check that you can ping between the computers both ways.

To avoid having to write the above commands every time a new terminal is opened, we have provided a script on the Robolab pcs called *setup_ros_ips.sh*. Simply go to Documents and run
`. ./setup_ros_ips.sh`
on the robolab pc. To create a similar script for your virtual machine, you can copy the script from the robolab machines and change the line:
`export ROS_IP=192.168.100.X`
to
`export ROS_IP=192.168.100.50`

5 Connecting to the depth sensors

5.1 Directly to your laptop

The optimal method for connecting to the depth sensors *Carmine* and *XtionPRO live* is though a usb cable directly to your own laptops using Openni2 drivers. However due to the variety of setups on your personal computers, we cannot provide you technical support for this method. The following is therefore an 'unofficial' guide to setting the system up on your own computers. This setup works fairly consistently outside a virtual machine.

- Install `Ubuntu 16.04` on your laptop
- `sudo apt-get install ros-kinetic-desktop-full`
- `sudo apt-get install ros-kinetic-openni2-launch`

5.2 Using virtual machine

The provided virtual machine already have the above components installed. Depending on the operating system there are unfortunately some problems connecting the depth sensors to the virtual machine. The following list shown which configurations have been found to work and which have not. If you try out a configuration that is not on the list, please send an email to `thmi@mmmi.sdu.dk` with the results so we can update the list.

- **Works**
  - `Windows 7` with `VMware Workstation 12 Player`
- **Doesn't work**
  - `Ubuntu 14.04` with `VMware Workstation 12 Player`
  - `Ubuntu 16.04` with `VMware Workstation 12 Player`

5.3 Using the machines in Robolab

The machines in Robolab have been setup as described above. You can launch the drive nodes on a Robolab machine and then connect to the nodes over the wired network. See the section on configuring the network.

5.4 Running the Depth sensor node

The *Carmine* and *XtionPRO live* are run using the *openni2-launch* node. We have provided a launch file which provides the necessary arguments. To launch the provided launch file do the following:

```
cd ~/Documents
roslaunch rgbd.launch
```

The openni2-launch node now publishes a large number of topics.

5.5 Trouble shooting

5.5.1 Check that information is being published to the topics

To check that your system works you can run:
`rosrun rqt_image_view rqt_image_view`
Try to select the following topics:

```
/rgbd/rgb/image_rect_color
/rgbd/depth/image
/rgbd/ir/image
```

If the outputs are an rgb image, a depth image and a black image, the depth sensor is running as it should.

The node is also publishing the reconstructed 3D points, both registered and non-registered. You can check that these are being published by running `rviz` in the terminal. In the graphical user interface do the follwing:

- Add PointCloud2
- Under *Global Options* select *Fixed Frame* to be fx. `/camera_rgb_optical_frame`
- Under the PointCloud2 tab, select the topic `/rgbd/depth_registered/points`

Rviz will now show the published registered point.

5.5.2 Check if the pc can detect the depth sensors

The following linux commands can help you analyze connection problems with the depth sensors:

- `lsusb -v` (list usb connections verbosely)
- `tail -f /var/log/syslog` (continuesly view the last lines of the syslog file. This will show messages when you plug in a sensor).

The output might give a "not enough bandwith" warning, or "unlikely big volume range" warning. These warnings can appear even on a working system.

6 Bumble Bee 2

The Bumble Bee 2 has to be connected over firewire. Since most laptops don't have firewire, you will most likely have to run the driver node on the Robolab machines and connect to it over wired network.

- The Bumble Bee 2 depends on the `ros-kinetic-pointgrey-camera-driver` package.

- To start the camera node in ROS use the `bumblebee.launch` file which comes with the package. So in a terminal run:

  ```
  roslaunch pointgrey_camera_driver bumblebee.launch
  ```

- If you have a firewire connection on your laptop you can install the driver on you own machine. To do this you can just run: `sudo apt-get install ros-kinetic-pointgrey-camera-driver`

6.1 Trouble shooting

- Run `coriander` in the terminal to check if there is a connection to the Bumble Bee.
  - If there is no connection, try to unplug the firewire cable and plug it into a different firewire port. Note that it might take a few seconds for the pc to detect the bumble bee.
- Check that you have both the left and the right image. Run for instance `rqt` and add visualizations (plugins → visualization → Image view for `camera/left/image_color` and `camera/right/image_color`)
  - If one image is shown correctly and the other is greyish
    - unplug the bumblebee, shutdown ROS, plugin the bumblebee and relaunch the bumblebee driver.
  - if the images are green it is probably a bug which causes the Bumblebee to not update the white balance. To fix this do the following:
    - While the bumblebee driver and rqt image viewers are running run `rosrun rqt_reconfigure rqt_reconfigre` in a new terminal. This allows you to change parameters dynamicly.
    - Select the bumblebee camera. You will now see a lot of different camera parameters. Find the tickbox `autowhitebalance`. Untick it and tick it again. This should prompt the bumblebee camera to recalibrate the whitebalance and should fix the color issue.

7 Connecting to the UR5 robot

7.1 Directly to your laptop

Before attempting to operate the robot, make sure you have installed RobWork, RobWorkStudio and RobWorkHardware. A guide on how to install these can be found on Black Board. You also have to install ROS Kinetic (follow the guide on `http://wiki.ros.org`) and create a catkin workspace. Once ROS is installed, creating the catkin workspace is done by following these steps:

```
mkdir -p ~/catkin_ws/src
cd ~/catkin_ws/src
catkin_init_workspace
cd ~/catkin_ws
catkin_make
```

To overlay this workspace on top of your environment, you have to source the `setup.bash` file from `catkin_ws/devel` . The easiest thing to do is to add the source in your `.bashrc` file. To do this, go to your home folder, press `Ctrl+h` to show the hidden files, open `.bashrc` and add the following line at the end of the file:

```
source /home/USERNAME/catkin_ws/devel/setup.bash.
```

The next step is to download and build the CAROS package in your catkin workspace. This can be done by following these steps:

```
cd ~/catkin_ws/src
git clone https://gitlab.com/caro-sdu/caros.git
cd ~/catkin_ws
catkin_make
```

Once CAROS is built, you need to replace a header file inside the *caros_universalrobot* package, in order for the collision checker to function correctly. The new header file can be found on BlackBoard. Download it and then replace the old one from:
`~/catkin_ws/src/caros/hwcomponents/caros_universalrobot/include/caros/test`

To work with the robot, follow these steps:

- Connect to the robot using an Ethernet cable and connect to the network connection created in 2.
- Change the IP address of the robot[3] and of the PC in the parameters .xml file in `catkin_ws\src\caros\hwcomponents\caros_universalrobot\launch`

  The *device_ip* is the robot's IP address and the *callback_ip* is your laptop's own ip address. You can find this one by opening a terminal and running `ifconfig` (check the *inet addr* from *eth0*)
- Ping the robot to check if the connection is established
- Open a terminal and run the *caros_universalrobot* launch file with:
  `roslaunch caros_universalrobot simple_demo_using_move_ptp.test`

You will see the robot move to a specific waypoint and backwards.
ATENTION! ALWAYS KEEP AN EYE ON THE ROBOT AND HAVE THE CONTROLLER STOP BUTTON CLOSE!

7.2 Using the UR5 simulator

Assuming you have installed ROS, RobWork, RobWorkStudio, RobWorkHardware and CAROS (with the header file change noted in the previous section), the next step is to download the UR5 simulator. Go to `https://www.universal-robots.com/download/?option=16594` and select your preferred simulator version and download. We recommend version 3.1. You might have to play around with the Java version to install the simulator.

Once installed, the UR5 controller simulator can be launched by clicking the `ursim-3.X UR5` Desktop icon, or by using the terminal command

```
cd workspace\ursim-3.X
./start-ursim.sh
```

where X is the number of the controller.

Once the controller simulator starts, click `OK` on `Go to initialization screen`, than go to `Run Program -> Move` and then click on `Home` to move the robot joints into their Home position. After that follow these steps:

- Change the IP address of the robot and of the PC in the parameters .xml file in `catkin_ws\src\caros\hwcomponents\caros_universalrobot\launch`

  The *device_ip* is the robot's IP address and the *callback_ip* is your laptop's own ip address. When running with the simulator, *device_ip* and the *callback_ip* should be the same. You can find the correct address by opening a terminal and running `ifconfig` (check the *inet addr* from *lo*).
- Open a terminal and run the *caros_universalrobot* launch file with:
  `roslaunch caros_universalrobot simple_demo_using_move_ptp.test`

You should see the robot move in the UR5 controller window.

[3]The IP address of the robot can be checked on the controller, in the NETWORK SETUP section

7.3 Using the virtual machine with the UR5 simulator

Because the URSimulator does not work in the *RoVi2_2017* virtual machine, you are provided with a second virtual machine (titled *URSimulator*) running Ubuntu 14.04 and the UR5 Simulator. To use the simulator in this configuration you will need two PC's: one running the *RoVi2_2017* virtual machine and a second one for the *URSimulator* virtual machine. The two PC's have to be connected via an Ethernet cable.

To test the functionality, you have to first launch the UR5 controller simulator in the *URSimulator* virtual machine. You can do this either by clicking the `ursim-3.1 UR5` Desktop icon, or by using the terminal command

```
cd workspace\ursim-3.1
./start-ursim.sh
```

Once the controller simulator starts, click `OK` on `Go to initialization screen`, than go to `Run Program -> Move` and then click on `Home` to move the robot joints into their Home position. After that follow the steps presented in the previous section to see the robot move to the new configuration.

In the *RoVi2_2017* virtual machine, follow these steps:

- Change the IP address of the robot and of the PC in the parameters .xml file in `catkin_ws\src\caros\hwcomponents\caros_universalrobot\launch`

  The *device_ip* is the IP address of the *URSimulator* virtual machine and the *callback_ip* is the *RoVi2_2017* virtual machine's ip address. You can find both of these by opening a terminal and running `ifconfig` (check the *inet addr* from *ess30*)

- Ping the *URSimulator* virtual machine to check if the connection is established

- Open a terminal and run the *caros_universalrobot* launch file with:
  `roslaunch caros_universalrobot simple_demo_using_move_ptp.test`

You should see the robot move in the UR5 controller window, in the *URSimulator* virtual machine.

8 Connecting to WSG-50 Schunk Gripper

8.1 Directly to your laptop

Before downloading and building the WSG-50 ROS package, you have to create a catkin ROS workspace. Assuming you already downloaded ROS, creating the catkin workspace is done by following these steps:

```
mkdir -p ~/catkin_ws/src
cd ~/catkin_ws/src
catkin_init_workspace
cd ~/catkin_ws
catkin_make
```

To overlay this workspace on top of your environment, you have to source the `setup.bash` file from `catkin_ws/devel` . The easiest thing to do is to add the source in your `.bashrc` file. To do this, go to your home folder, press `Ctrl+h` to show the hidden files, open `.bashrc` and add the following line at the end of the file:

```
source /home/USERNAME/catkin_ws/devel/setup.bash.
```

To download and build the WSG-50 package, follow these steps:

```
cd ~/catkin_ws/src
```

```
git clone https://github.com/nalt/wsg50-ros-pkg.git
cd ~/catkin_ws
catkin_make
```

The next step is changing the IP address of the gripper in the launch file. This is done by going to `cd ~/cd catkin_ws/src/wsg50-ros-pkg/wsg_50_driver/launch/` opening the `wsg_50_tcp.launch` and change the IP value to `192.168.100.10`.

Now you are ready to operate the gripper. There are two important steps in connecting the WSG-50 gripper:

- Power up the gripper. Notice that the LED lights blue - this means it is powering up.Then it starts fast blinking purple - this means it is reading the internal configuration files and then it starts slowly blinking purple - this means it is ready to be used.
- Connect the Ethernet to your PC [4] and use the correct network connection (the one you set up in 2)

To make sure that you are properly connected to the gripper, you can run:

`ping 192.160.100.10`.

To launch the gripper, in the terminal write the following:

```
roslaunch wsg_50_driver wsg_50_tcp.launch
```

The gripper should move its fingers.

8.2 Using the virtual machine

In the virtual machine the WSG-50 package is already built and configured. The only thing you need to take care of is the two steps in connecting to the gripper. You can launch the gripper with the following command:

```
roslaunch wsg\_50\_driver wsg\_50\_tcp.launch
```

9 Connecting to PG70 Schunk gripper

To operate the PG70 Schunk gripper, you first have to power it up and connect it to the PC. To launch the PG70 node, open the terminal and type the command:

```
roslaunch caros_schunkpg70 caros_schunkpg70.launch
```

You should now be able to hear a clicking sound. This means that the gripper is ready to be used.

The procedure is the same for both the virtual machine and your own laptop.

[4] Only do this if the LED is slowly blinking purple