Robotnik

TurtleBot™

Open Source Robotics Foundation

YUJIN ROBOT

TURTLEBOT II

User's Manual

Robotnik Automation, S.L.L.
Kinetic v1

Index

1. Robot description

Kobuki Turtlebot 2 is a low-cost mobile research base designed for education and research on state of art robotics. With continuous operation in mind, Kobuki provides power supplies for an external computer as well as additional sensors and actuators. Its highly accurate odometry, amended by our factory calibrated gyroscope, enables precise navigation.

This robot is a compatible replacement for the iRobot Create. Although most of the accessories are fully compatible, be careful when plugging any device from the iRobot version to Kobuki version, because the connectors pin-out could be different.

1.1 Functional Specification

- Maximum translational velocity: 65 cm/s
- Maximum rotational velocity: 3.14 rad/s
- Payload: 5 kg (hard floor), 4 kg (carpet)
- Cliff: will not drive off a cliff with a depth greater than 5 cm
- Threshold Climbing: climbs thresholds of 12 mm or lower
- Rug Climbing: climbs rugs of 12 mm or lower
- Expected Operating Time: 3/7 hours (small/large battery)
- Expected Charging Time: 1.5/2.6 hours (small/large battery)
- Docking: can perform docking within a 2mx5m area in front of the docking station

1.2 Hardware Specification

- PC Connection: usb or via RX/TX pins on the parallel port
- Motor Overload Detection : disables power to motors on detecting high current
- Odometry: 25718.16 ticks/revolution, 11.7 ticks/mm
- Gyro: factory calibrated, 1 axis (100 deg/s)
- Bumpers: left, center, right
- Cliff sensors: left, center, right
- Wheel drop sensor: left, right
- Power connectors: 5V/1A, 12V/1.5A, 12V/5A
- Docking recharging connector: 19V/2.1A- Expansion pins: 3.3V/1A, 5V/1A, 4 x analog in, 4 x digital in, 4 x digital out
- Audio : several programmable beep sequences
- Programmable LED: 2 x two-coloured LED
- State LED: 1 x two coloured LED [blinking - charging, Green - high level, Orange - low level]
- Buttons: 3 x touch buttons
- Battery: 14.8 V lithium-Ion 2200 mAh (small) 4400 mAh (large)

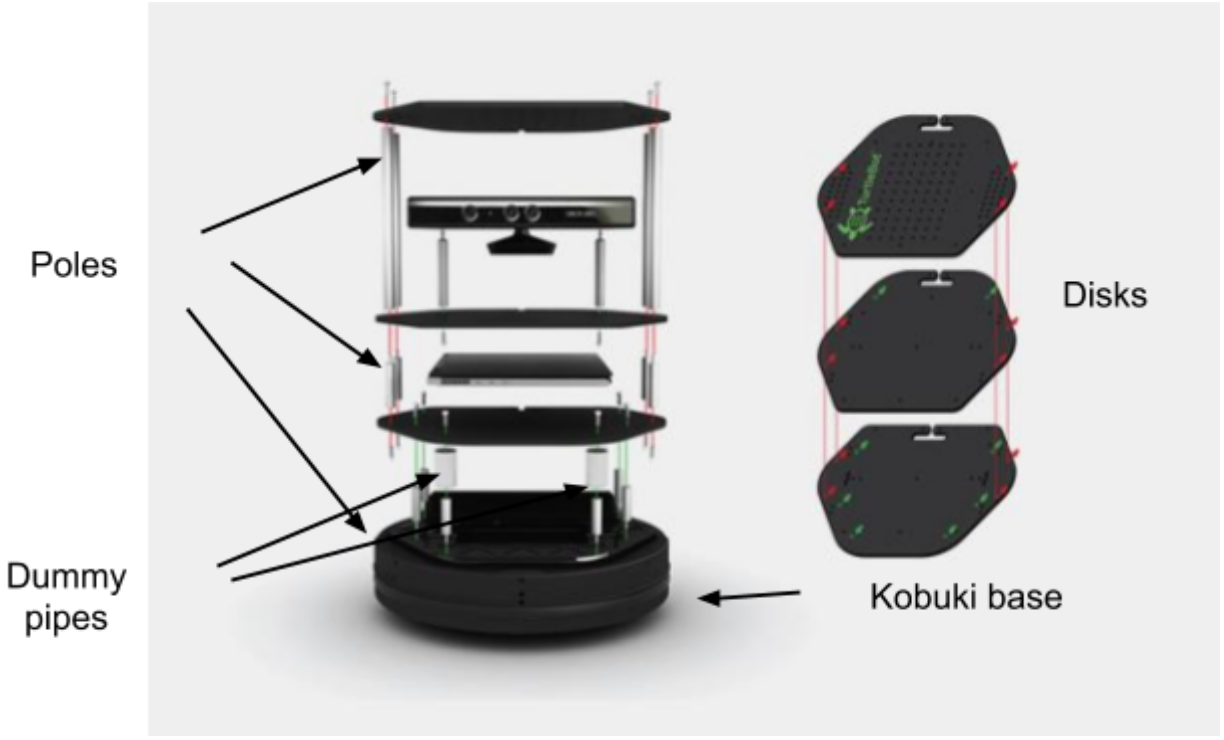
- Firmware upgradeable: via usb
- Sensor Data Rate: 50Hz
- Recharging Adapter: Input: 100-240V AC, 50/60Hz, 1.5A max; Output: 19V DC, 3.16A
- Netbook recharging connector (only enabled when robot is recharging): 19V/2.1A DC
- Docking IR Receiver: left, centre, right

1.3 Software Specification

- Kobuki drivers for ROS and non-ROS in C++
- Kobuki and Turtlebot simulator (Gazebo)
- Kobuki and Turtlebot apps

2. Components list

2.1 Basic components



Kobuki base

Mobile base.

Kobuki Hardware

It's the hardware mounted over the base.
It includes:

- 16 Poles
- 3 Disks
- 2 Dummy pipes

Battery

The 4S1P battery 2200 mAh. It provides 3 hours of operational time.

Kinect cable

Cable to power the Kinect sensor directly from the robot's battery.

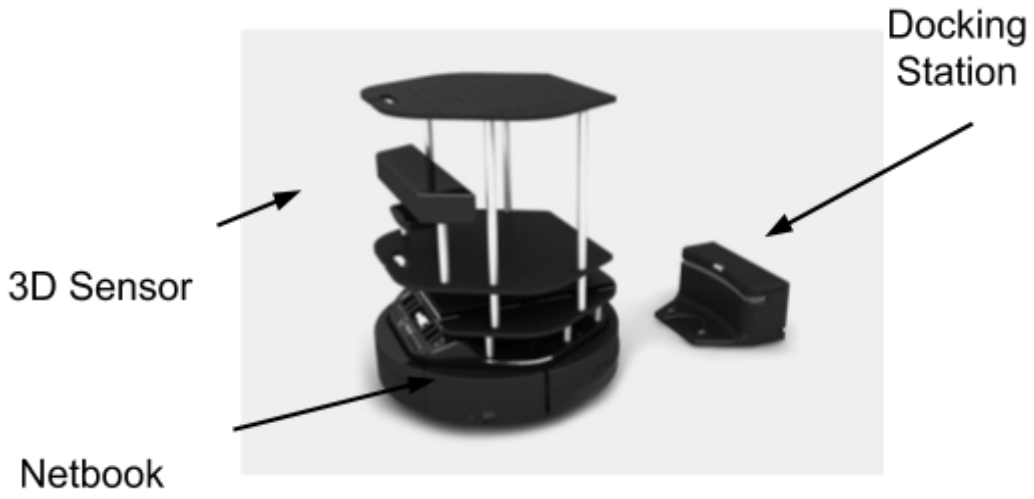
Usb communication cable

Cable to control the Kobuki base from a PC.

Recharging adapter

Standard adapter to charge the battery manually.

2.2 Additional components



3D sensor

The Turtlebot 2 is intended to use a 3D sensor in order to navigate autonomously and map. By default, the Kinect sensor is provided with the robot, but any other sensor can be attached.

Docking station

Battery charger intended to carry out an autonomous battery charge.

Large battery

The 4S2P battery 4400 mAh. It provides 7 hours of operational time.

Netbook

Netbook installed and configured to control the robot and all of its accessories.
It is also possible to use other controllers or pcs.

Turtlebot's USB stick

USB stick that contains all the manuals, and a recovery image of the provided netbook.

3. Assembly instructions

In this section will be described how to assembly the robot and some other useful information. This guide is based on the information available on the official site http://kobuki.yujinrobot.com.

3.1 Assembling the robot

Please refer to the provided assembly guide.

3.2 Connectors

Power

5V@1A Molex PN : 43650-0218 - for custom boards
12V@1.5A : Molex PN : 43045-0224 - specially supporting the kinect
12V@5A : Molex PN : 5566-02B2 - for high powered accessories (e.g. robotic arm)
19V@2A : Molex PN : 3928-9048 - for recharging netbooks

⚠ WARNING: This pin-out can differ from the Turtlebot 1. Please be careful when connecting accessories.
⚠ WARNING: Please verify the pinout or the provided connector before connecting any external device.

Cable

Note, if you click on the preceding links for the power connectors, under the heading Mates with Part(s) you can find the compatible connector to use with each power source. The most important one being of course:
12V@1.5A : Molex PN : 43025-0200 - specially supporting the kinect

Battery

- 4S1P, 2200 mAh. Default Battery Pack Connector
- 4S2P, 4400 mAh. Extra Battery Pack Connector

IO Port

DB25 pin D-SUB Female connector that provides the following functionality.
4 x Digital Input
4 x Digital Output
4 x Analog Input
RX/TX
3.3V
5V

3.3 Swapping batteries

Parts
1 x 4S1P Lithium-Ion battery - default
1 x 4S2P Lithium-Ion battery - optional (longer life)

Default - 4S1P Battery Pack

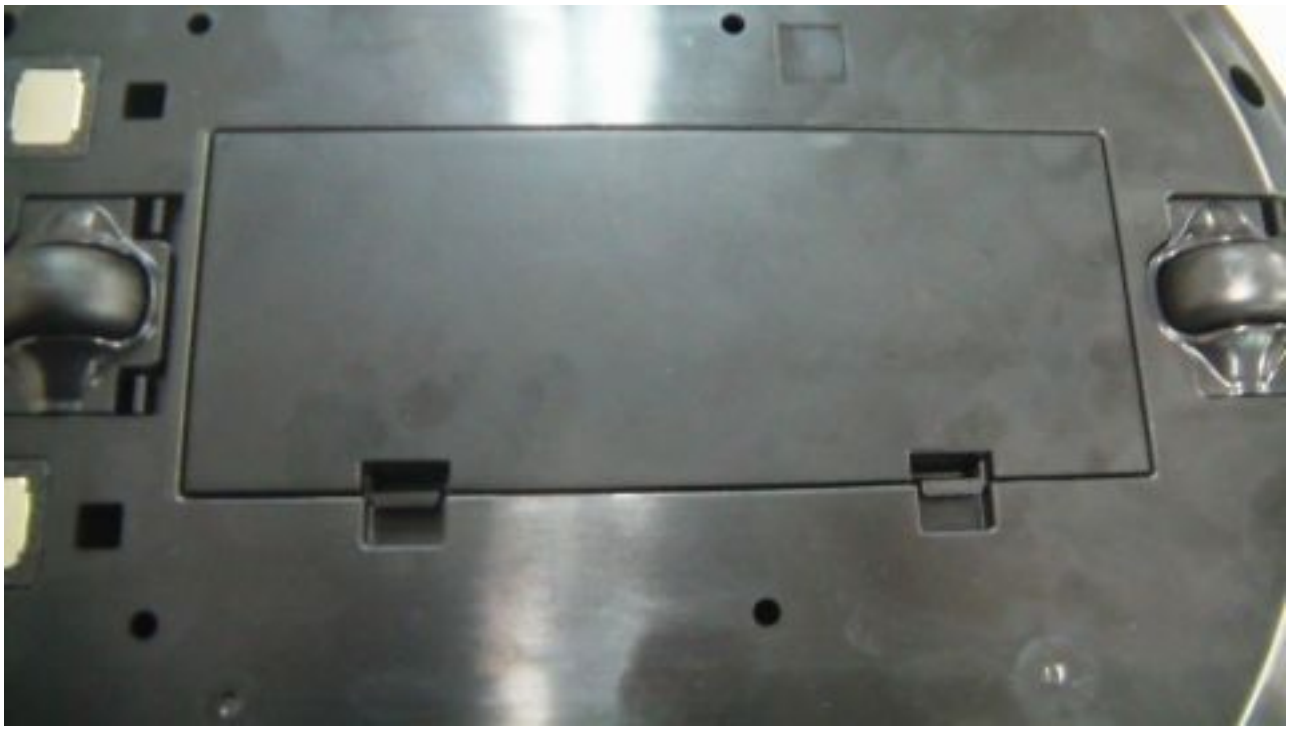
By default, Kobuki comes with a 4S1P battery pack that is stored underneath robot behind a cover at the middle of the base.

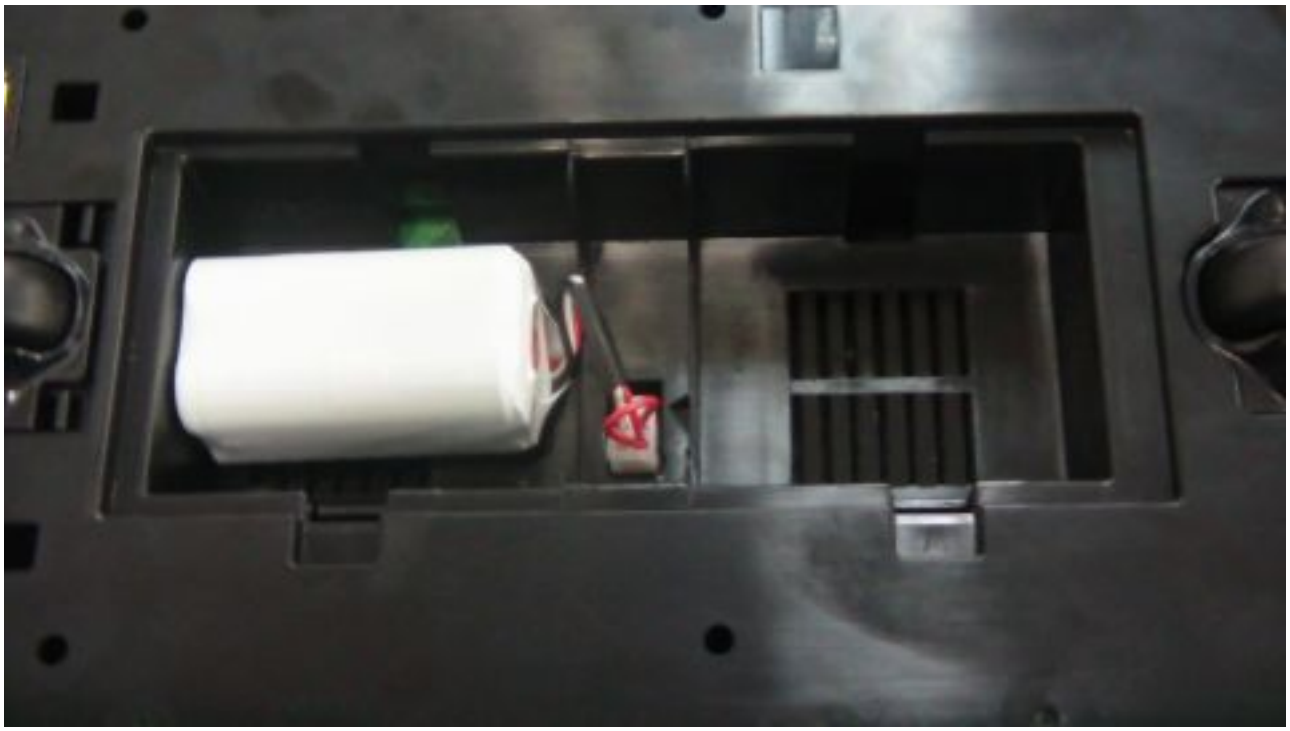
Optional - 4s2p Battery Pack

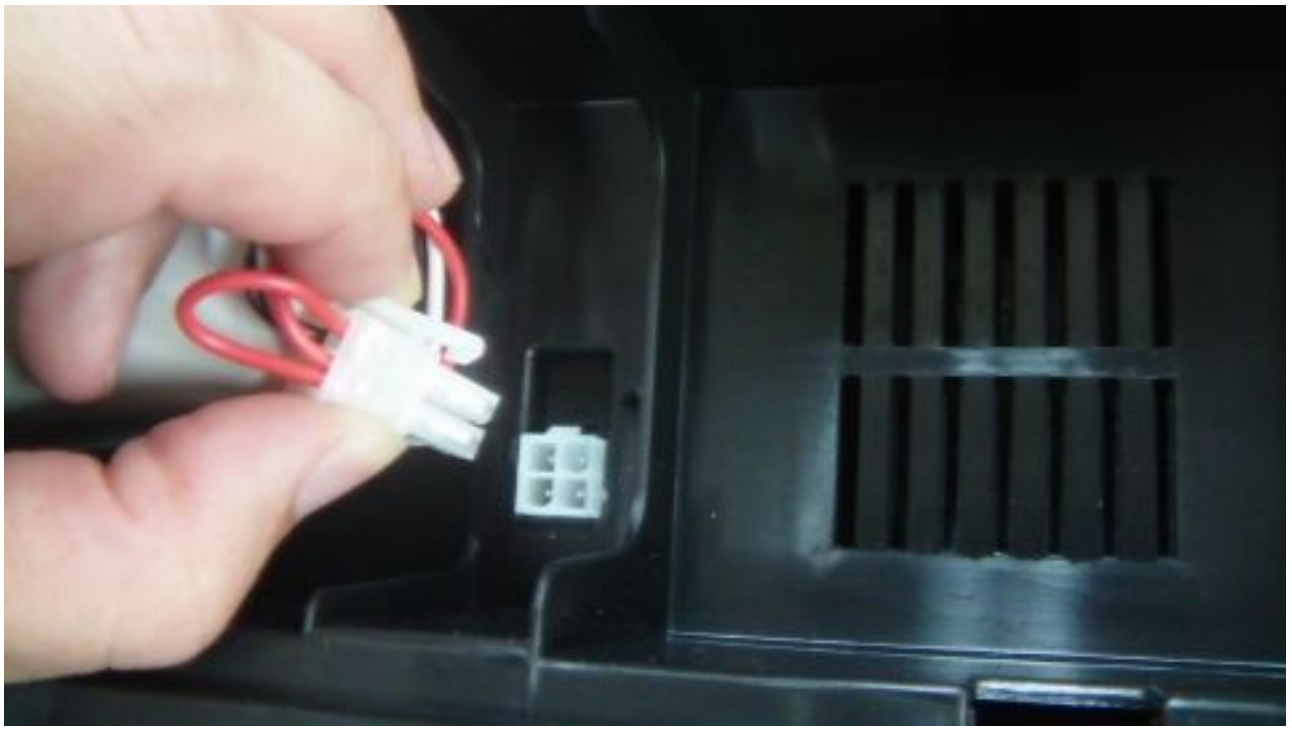
For longer life, the 4S2P pack can be used.
⚠ You cannot use 4S1P and 4S2P packs at the same time!
Remove the 4S1P battery from the base compartment.

Optional - 2x4s2p Battery Pack in parallel

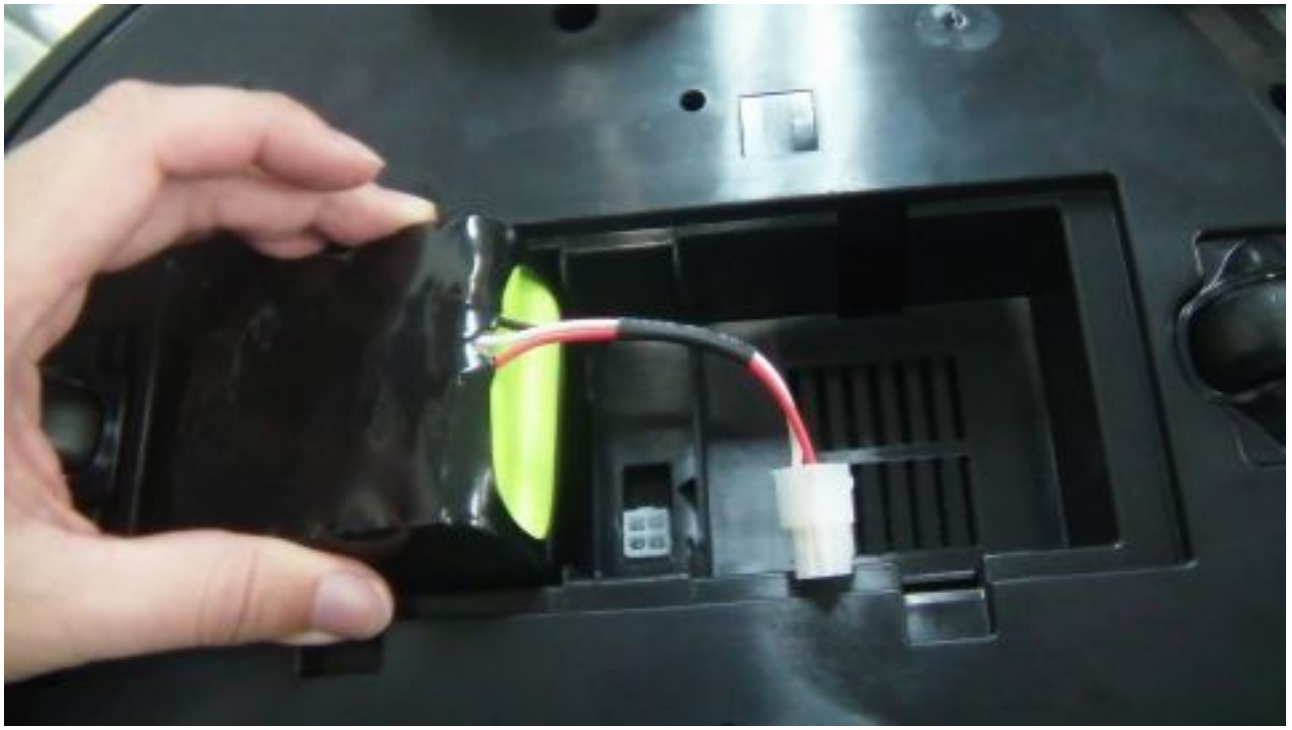
For configurations working with NUC controllers, it is recommended the use of a custom battery pack made from two standard 4S2P batteries.

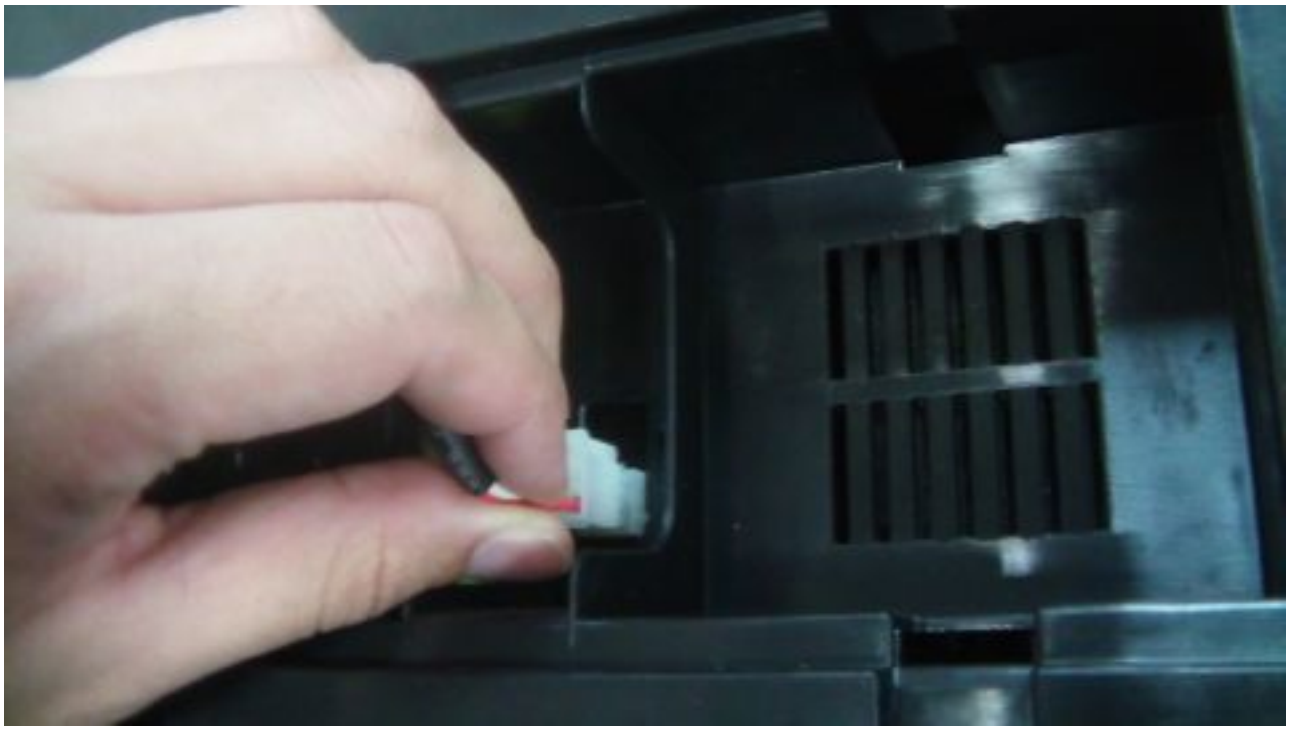


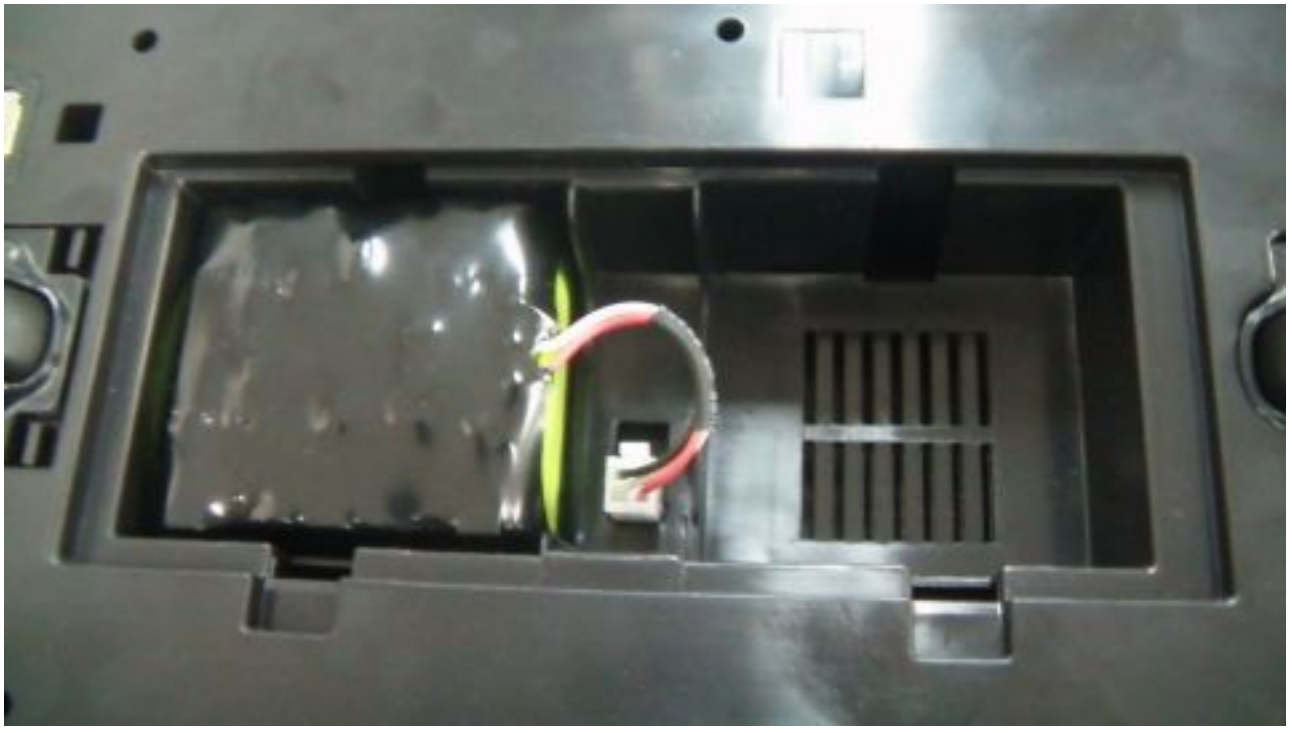


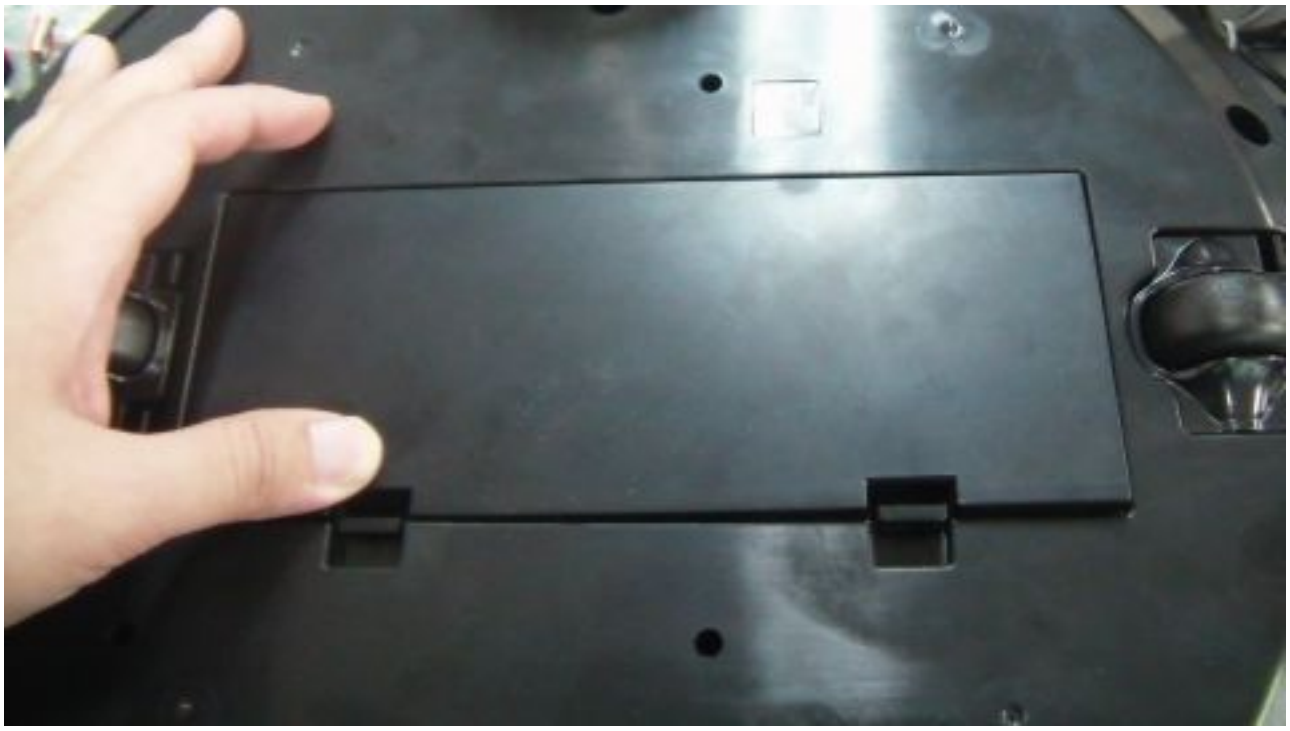


Plug the 4S2P battery into the connector.









Offline Charging

There is also an accessory for those who'd like to offline charge the robot (a pseudo continuous operation mode). Just directly connect... For both online/offline charging, it should take approximately 2 hours to recharge the 4s1p battery fully, and over 3 hours for the 4s2p one.

4. Installation and Configuration

In this section will be described the installation process of the netbook recovery image and the installation of the Turtlebot packages from the official repository.

4.1 Default User & Password

User: **turtlebot**
Password: **ros**

4.2 Installing from Robotnik's USB Installer

In this section will be described the installation of the system by using the USB installer provided by Robotnik.
The provided usb contains either a Ubuntu 14.04 Installer with ROS and the Turtlebot 2 packages, or an image of the system provided (netbook or pc).

4.3 ROS package installation

In this section will be described the ROS package installation from scratch. For further information about this process or about ROS, please refer to www.ros.org.
These packages are supported for Kinetic (current version) version.

Install ROS bases

Install ROS in Ubuntu 16.04 (Trusty):

```
$> sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu
$(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
$> sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80
--recv-key 421C365BD9FF1F717815A3895523BAEEB01FA116
$> sudo apt-get update
$> sudo apt-get install ros-kinetic-desktop-full
$> sudo apt-get install ros-kinetic-kobuki*
$> sudo apt-get install ros-kinetic-turtlebot*
$> sudo apt-get install ros-kinetic-openni* (if using rgbd compatible
camera)
```

```
$> sudo apt-get install ros-kinetic-astra* (for the camera astra)
$> sudo apt-get install ros-kinetic-urg-node (for the hokuyo lasers)
```

Rosdep

Install all dependencies of kobuki_node

```
$> sudo rosdep init
$> rosdep update
```

ROS environment variables

```
$> echo "source /opt/ros/kinetic/setup.bash" >> ~/.bashrc
$> source ~/.bashrc
```

Set udev Rule

Copy across a udev rule for */dev/kobuki* and add the user to the dialout group (you may need to logout and log back in). See *kobuki_ftdi* for more details.

```
$> roscd kobuki_ftdi
$> rosrun kobuki_ftdi create_udev_rules
$> sudo udevadm trigger
```

For the Astra cameras

It is possible to copy or following the installation instructions at https://github.com/tfoote/ros_astra_camera.

```
$> cd  /etc/udev/rules.d/
```

Create a new file as sudo:

```
$> sudo gedit 56-orbbec.rules
```

Copy the following lines:

```
SUBSYSTEM=="usb", ATTR{idProduct}=="0400", ATTR{idVendor}=="2bc5", MODE:="0666", OWNER:="root",
GROUP:="video"
SUBSYSTEM=="usb", ATTR{idProduct}=="0401", ATTR{idVendor}=="2bc5", MODE:="0666", OWNER:="root",
GROUP:="video"
SUBSYSTEM=="usb", ATTR{idProduct}=="0402", ATTR{idVendor}=="2bc5", MODE:="0666", OWNER:="root",
GROUP:="video"
SUBSYSTEM=="usb", ATTR{idProduct}=="0403", ATTR{idVendor}=="2bc5", MODE:="0666", OWNER:="root",
GROUP:="video"
SUBSYSTEM=="usb", ATTR{idProduct}=="0404", ATTR{idVendor}=="2bc5", MODE:="0666", OWNER:="root",
```

```
GROUP:="video"
SUBSYSTEM=="usb", ATTR{idProduct}=="0405", ATTR{idVendor}=="2bc5", MODE:="0666", OWNER:="root",
GROUP:="video"
SUBSYSTEM=="usb", ATTR{idProduct}=="0406", ATTR{idVendor}=="2bc5", MODE:="0666", OWNER:="root",
GROUP:="video"
SUBSYSTEM=="usb", ATTR{idProduct}=="0407", ATTR{idVendor}=="2bc5", MODE:="0666", OWNER:="root",
GROUP:="video"
SUBSYSTEM=="usb", ATTR{idProduct}=="0408", ATTR{idVendor}=="2bc5", MODE:="0666", OWNER:="root",
GROUP:="video"
SUBSYSTEM=="usb", ATTR{idProduct}=="0409", ATTR{idVendor}=="2bc5", MODE:="0666", OWNER:="root",
GROUP:="video"
SUBSYSTEM=="usb", ATTR{idProduct}=="040a", ATTR{idVendor}=="2bc5", MODE:="0666", OWNER:="root",
GROUP:="video"
```

Restart the udev system service:

```
$> sudo service udev reload
$> sudo service udev restart
```

Unplug the camera and plug it again.

Keyboard Teleoperation

If you followed the instruction well with no error pop-up, you should be able to telop your kobuki around with your keyboard!

```
$> roslaunch kobuki_node minimal.launch
$> roslaunch kobuki_keyop keyop.launch
```

Install other useful packages

```
$> sudo apt-get install ssh
```

4.4 Upgrading the firmware

One of the nice features of Kobuki is that the user can update the firmware released by Yujin themselves. When a new firmware version is uploaded, this can be manually flashed onto the robot via usb without the need for special equipment like JTAG devices.
Follow the instructions matching the operating system you will connect to the robot for the flashing operation. For further information, please refer to the official site http://kobuki.yujinrobot.com.
**All robots are provided with the last firmware by default.*

4.5 Additional components

The installation of additional components like 2D lasers, arms, ... requires some additional configuration efforts.

1. Create a catkin workspace.
2. Installation of Robotnik alternative Turtlebot repository
    a. https://github.com/RobotnikAutomation/turtlebot
    b. Clone the repository into the workspace (catkin_ws/src) and compile it
    c. Add the following into the .bashrc
        i. source ~/catkin_ws/devel/setup.bash
3. Other dependencies to install:
    a. https://github.com/RobotnikAutomation/widowx_arm
    b. https://github.com/RobotnikAutomation/phantomx_reactor_arm

4. catkin_make desde la carpeta ~/catkin_ws.

4.5.1 RPLIDAR

Dependencies:
c d
*`ros-kinetic-rplidar-ros`

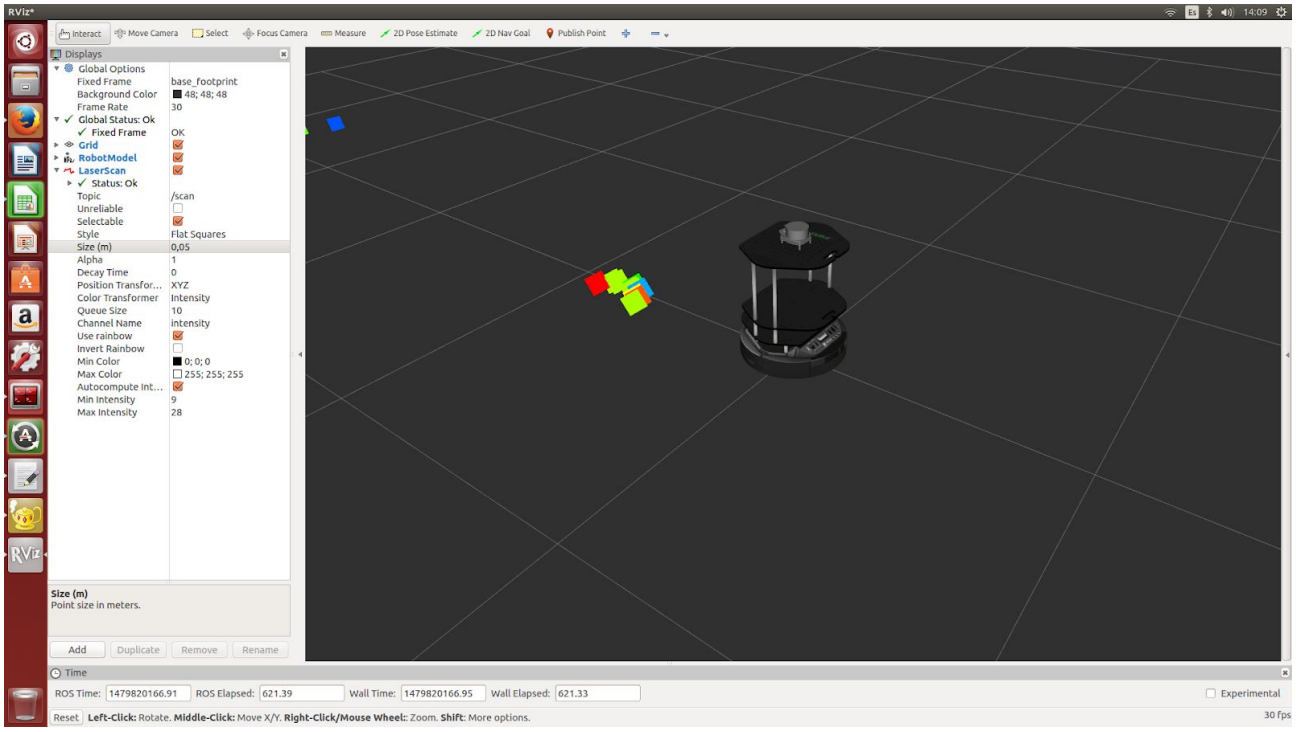
Add the following variable environments into the .bashrc and reload it:

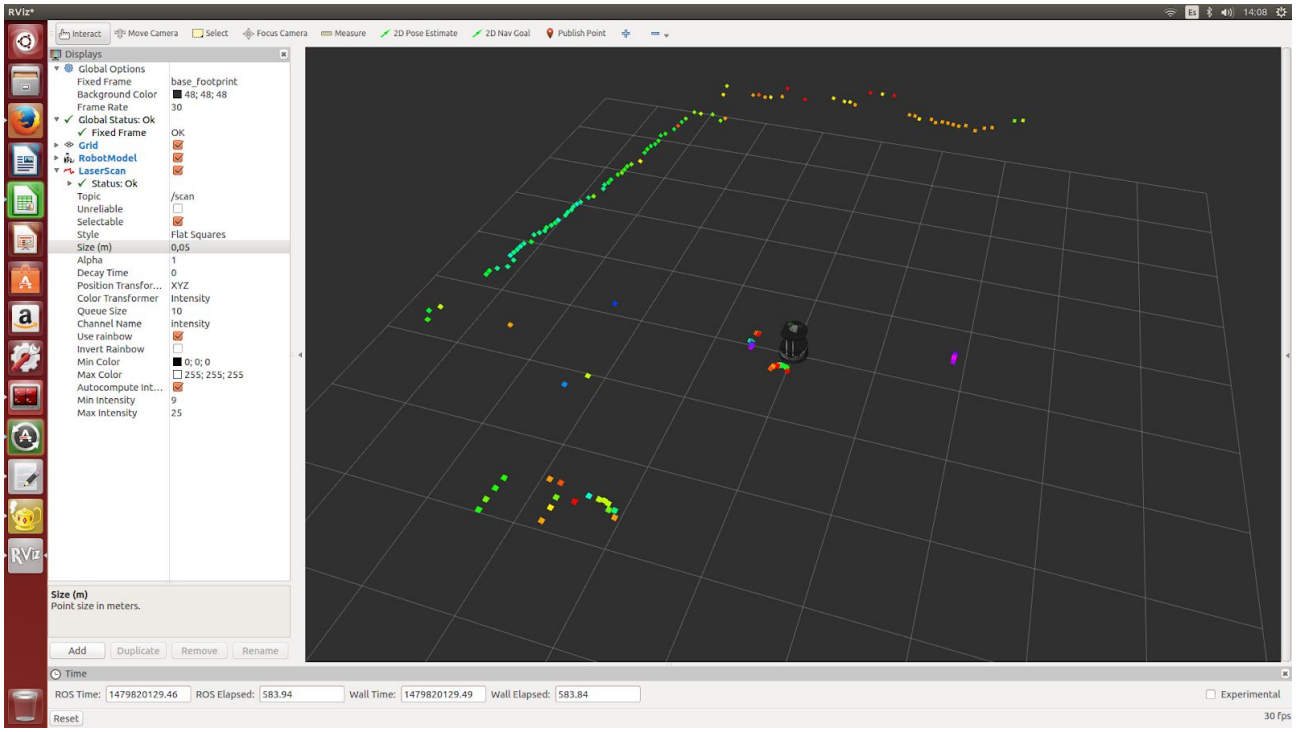
```
export TURTLEBOT_TOP_PLATE_DEVICE=rplidar
export TURTLEBOT_3D_SENSOR=no3d
```

1. Create the udev rules:
    a. `> roscd turtlebot_scripts`

2. Run the script contained in the scripts folder:
    a. `> ./scripts/create_rplidar_udev_rules`
    b. The device should be linked in /dev/rplidar

3. To launch the sensor:
    a. `> roslaunch turtlebot_bringup rplidar.launch`





4.5.2 Hokuyo URG04LX

Dependencies:

```
*ros-kinetic-urg-node
```

Set the user permissions:

```
> sudo usermod -a -G dialout turtlebot
```

Add the following variable environments into the .bashrc and reload it:

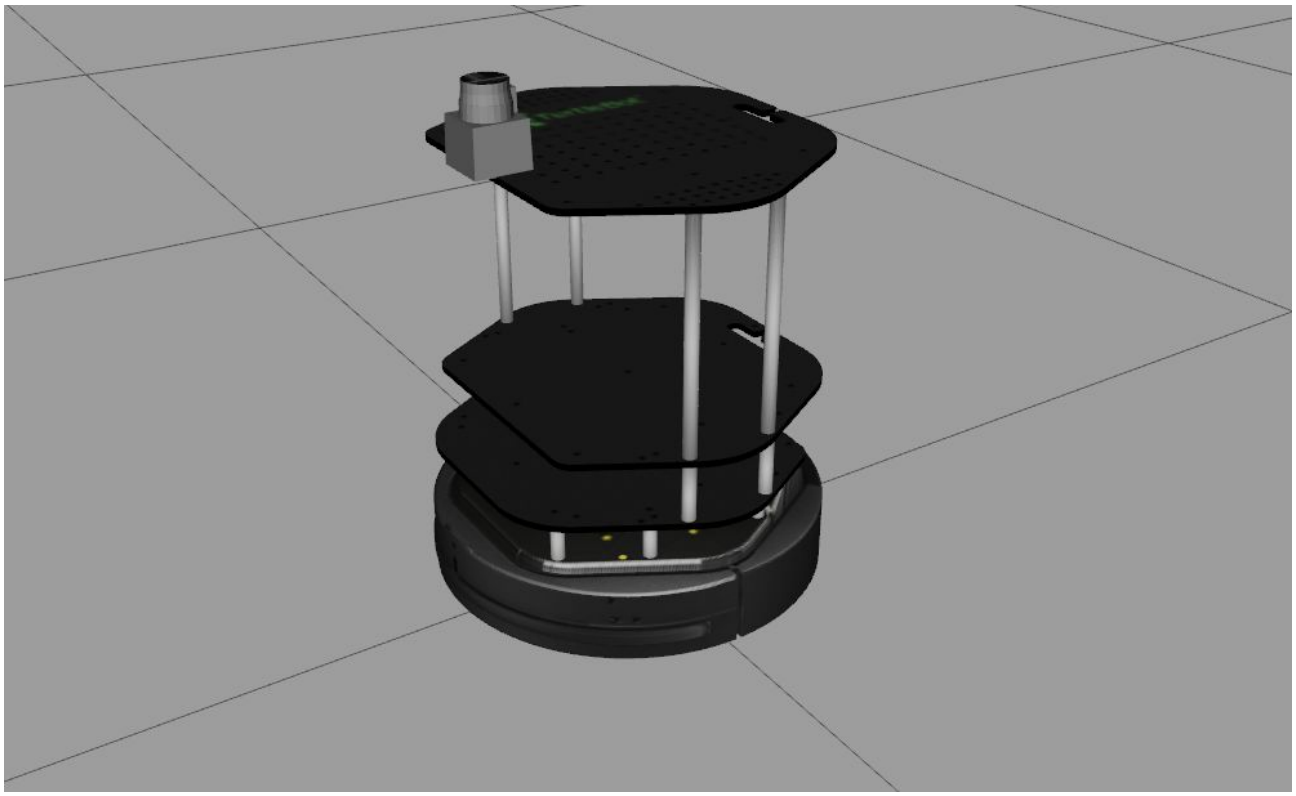
Configuration with the sensor on the **top** plate:

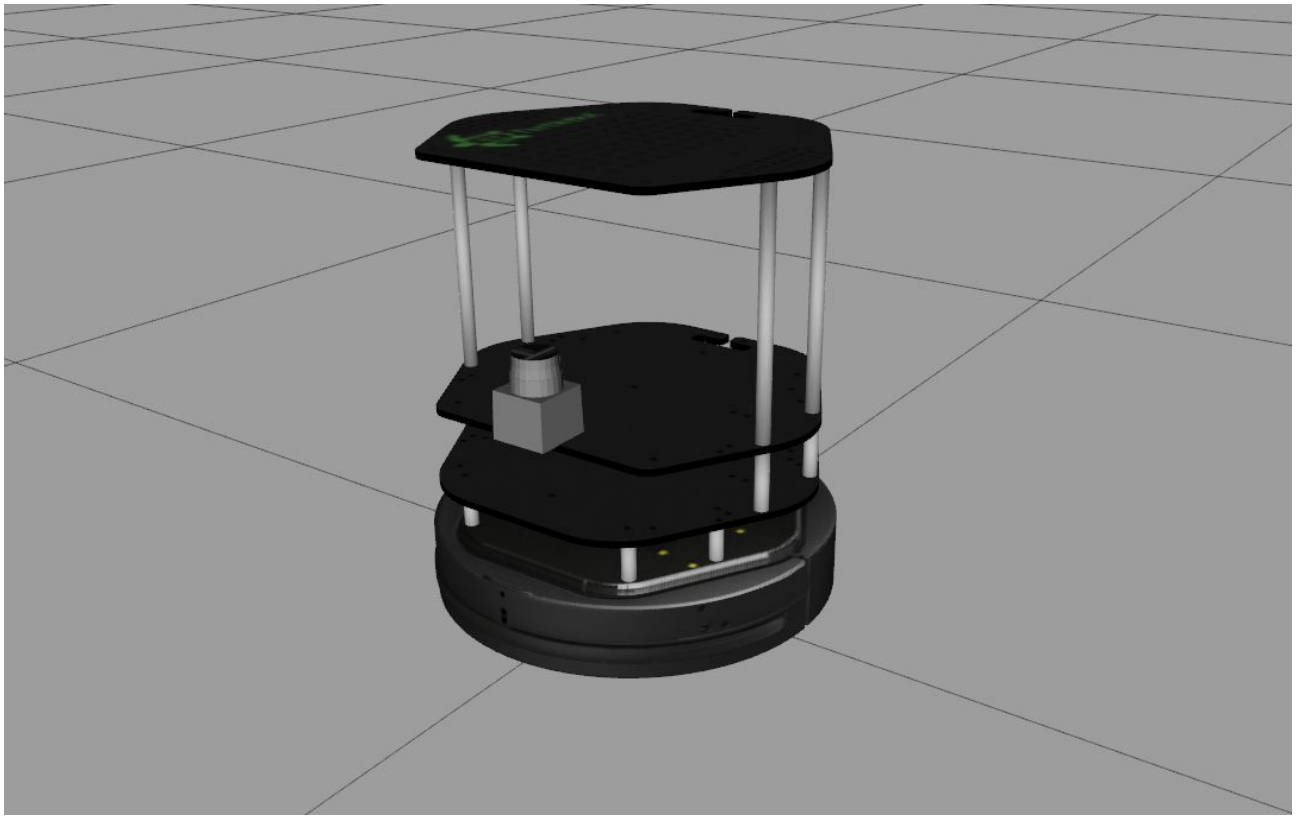
```
export TURTLEBOT_TOP_PLATE_DEVICE=urg04lx
export TURTLEBOT_3D_SENSOR=no3d
```

Configuration with the sensor on the **middle** plate:

```
export TURTLEBOT_TOP_PLATE_DEVICE=notop
export TURTLEBOT_3D_SENSOR=urg04lx
```

1. To launch the sensor:
    a. `> roslaunch turtlebot_bringup minimal.launch`
    b. `> roslaunch turtlebot_bringup hokuyo_urg04lx.launch`





4.5.3 WidowX Arm

Dependencies:

```
*ros-kinetic-arbotix*
```

Set the user permissions:

```
> sudo usermod -a -G dialout turtlebot
```

Add the following variable environments into the .bashrc and reload it:

export TURTLEBOT_TOP_PLATE_DEVICE=widowx

Create the udev rule:

```
> udevadm info -a -n /dev/ttyUSBXX (device connected to the arm)
```

Get the information of the attribute ATTRS{serial}=="A904NK42".

Create a new rule in /etc/udev/rules.d/58-widowx.rules with the following data:

```
SUBSYSTEM=="tty", ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6001",
ATTRS{serial}=="A904NK42", MODE:="0666", GROUP
:="dialout", SYMLINK+="ttyUSB_WIDOWX"
```

Run the following commands to restart the udev daemon:

```
> sudo restart udev
> udevadm trigger
```

See Appendix 1 - Setting up Arbotix board to configure the Arbotix.

Then start the arm controller with:

```
> roslaunch widowx_arm_controller widowx_arm_controller.launch
```

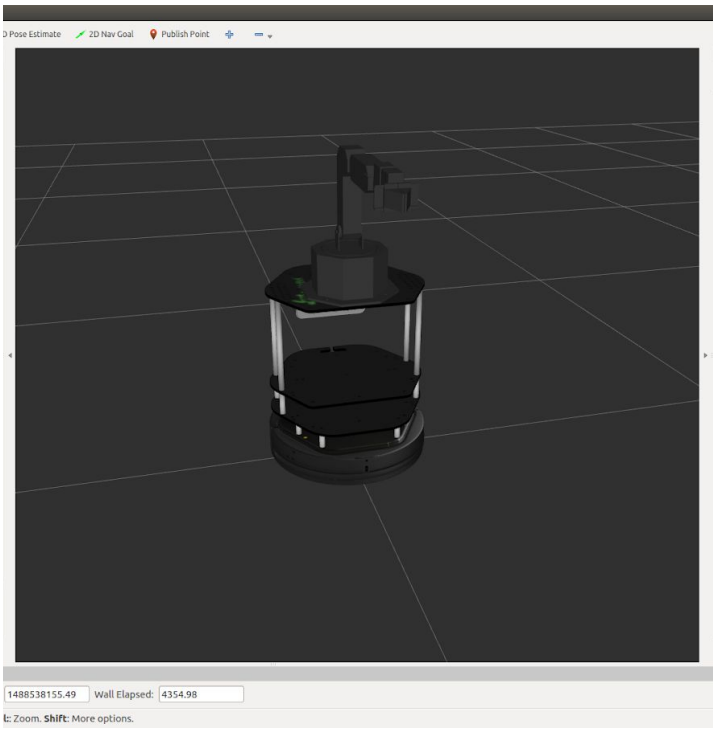
In the widowx_arm_controller.launch are specified two nodes. The first node is the arbotix_driver node and the second is a node to control the gripper.

To control the aperture of the gripper in linear mode, the range allowed is from 0 to 0.03 m.

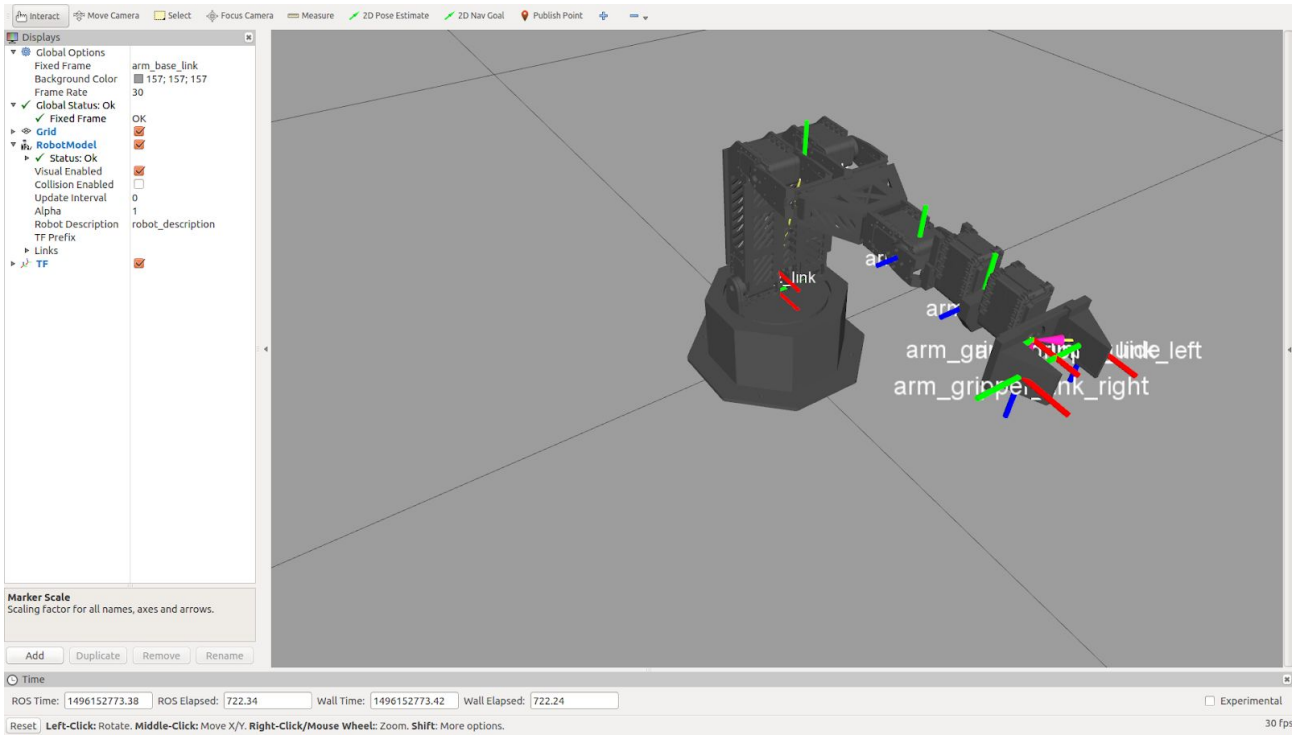
You could use the following commands:

`rostopic pub /gripper_prismatic_joint/command std_msgs/Float64 "data=0.00"` to close the gripper

`rostopic pub /gripper_prismatic_joint/command std_msgs/Float64 "data=0.03"` to open the gripper completely.



4.5.4 PhantomX Reactor Arm



Dependencies:

```
*ros-kinetic-arbotix*
```

Set the user permissions:

```
> sudo usermod -a -G dialout turtlebot
```

Add the following variable environments into the .bashrc and reload it:

export TURTLEBOT_TOP_PLATE_DEVICE=reactor_wrist

Create the udev rule:

```
> udevadm info -a -n /dev/ttyUSBXX (device connected to the arm)
```

Get the information of the attribute ATTRS{serial}=="A904NK42".

Create a new rule in /etc/udev/rules.d/59-reactor.rules with the following data:

```
SUBSYSTEM=="tty", ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6001",
ATTRS{serial}=="A904NK42", MODE:="0666", GROUP
:="dialout", SYMLINK+="ttyUSB_REACTOR"
```

Run the following commands to restart the udev daemon:

```
> sudo restart udev
> udevadm trigger
```

See Appendix 1 - Setting up Arbotix board to configure the Arbotix.

Then start the arm controller with:

```
>roslaunch turtlebot_bringup
arbotix_phantomx_reactor_arm_wrist.launch

SUMMARY
========

PARAMETERS
 * /arbotix/joints/arm_1_joint/id: 1
 * /arbotix/joints/arm_1_joint/max_speed: 75.0
 * /arbotix/joints/arm_2_1_joint/id: 3
 * /arbotix/joints/arm_2_1_joint/max_speed: 75.0
 * /arbotix/joints/arm_2_joint/id: 2
 * /arbotix/joints/arm_2_joint/max_speed: 75.0
 * /arbotix/joints/arm_3_1_joint/id: 5
 * /arbotix/joints/arm_3_1_joint/max_speed: 75.0
 * /arbotix/joints/arm_3_joint/id: 4
 * /arbotix/joints/arm_3_joint/max_speed: 75.0
 * /arbotix/joints/arm_4_joint/id: 6
 * /arbotix/joints/arm_4_joint/max_speed: 75.0
```

```
 * /arbotix/joints/arm_5_joint/id: 7
 * /arbotix/joints/arm_5_joint/max_speed: 75.0
 * /arbotix/joints/arm_gripper_joint/id: 8
 * /arbotix/joints/arm_gripper_joint/max_angle: 0
 * /arbotix/joints/arm_gripper_joint/max_speed: 114.0
 * /arbotix/joints/arm_gripper_joint/min_angle: -90.0
 * /arbotix/joints/arm_gripper_joint/range: 180
 * /arbotix/port: /dev/ttyUSB0
 * /arbotix/rate: 100
 * /robot_description: <?xml version="1....
 * /rosdistro: kinetic
 * /rosversion: 1.11.21

NODES
  /
    arbotix (arbotix_python/arbotix_driver)
    phantomx_reactor_controller
(phantomx_reactor_arm_controller/phantomx_reactor_parallel_motor_j
oints.py)
    robot_state_publisher (robot_state_publisher/state_publisher)
    rviz (rviz/rviz)

ROS_MASTER_URI=http://robotnik-GE70:11311

core service [/rosout] found
process[arbotix-1]: started with pid [16444]
process[phantomx_reactor_controller-2]: started with pid [16445]
process[robot_state_publisher-3]: started with pid [16446]
process[rviz-4]: started with pid [16447]
[INFO] [WallTime: 1496152050.354436] Started ArbotiX connection on
port /dev/ttyUSB0.
[INFO] [WallTime: 1496152050.481723] ArbotiX connected.
```

Note: In the phantomx_reactor_arm_controller.launch are specified two nodes. The first node is the arbotix_driver node and the second is a node to control the the parallel joints.

Check that the joint values are being published:

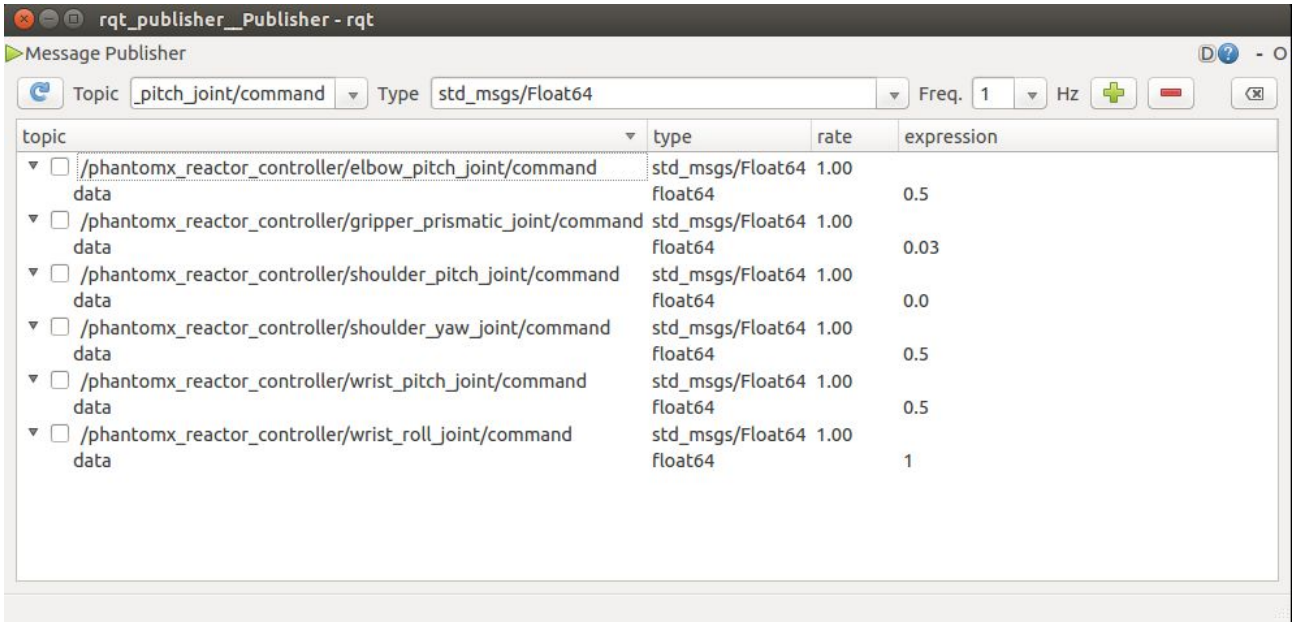
```
>rostopic echo /joint_states

header:
  seq: 3766
  stamp:
    secs: 1496152501
    nsecs: 951384067
  frame_id: ''
name: ['arm_1_joint', 'arm_2_1_joint', 'arm_2_joint',
'arm_3_1_joint', 'arm_3_joint', 'arm_gripper_joint',
```

```
'arm_4_joint', 'arm_5_joint']
position: [0.0051132692929521375, -0.02556634646476069,
0.02556634646476069, 0.0051132692929521375, -0.02556634646476069,
-0.006135923151542565, -0.010226538585904275,
-0.010226538585904275]
velocity: [-0.042620525427073536, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0]
effort: []
```

Run the rqt_publisher tool to set the position of the joints:

```
> rosrun rqt_publisher rqt_publisher
```



4.5.5 Hokuyo UST10LX

Dependencies:

```
*ros-kinetic-urg-node
```

Set the user permissions:

```
> sudo usermod -a -G dialout turtlebot
```

To launch the sensor:

```
>roslaunch turtlebot_bringup hokuyo_ust10lx.launch
```

4.6 Additional WiFi router

IP: 192.168.0.1
User/Pass: admin/R0b0tn1K
SSID pass: R0b0tn1K

5. First steps

In this section will be described the procedure to start and test the main parts of the robot.

5.1 Launching Kobuki

First, start minimal.launch to bring up Kobuki's basic software (bootstrap layer). We use the argument --screen to get verbose output in the terminal.

```
$> roslaunch kobuki_node minimal.launch --screen
```

This launch file starts a nodelet manager and loads the Kobuki nodelet (the ROS wrapper around Kobuki's driver).

```
$> roslaunch kobuki_keyop keyop.launch
```

This launch file starts the control of the robot using the keyboard keys.

```
$> roslaunch kobuki_node kobuki_state_publisher.launch
```

This launch file starts a node that publish the TF transformation using the robot's *.urdf file.

5.2 Testing the hardware

The kobuki_testsuite package provides a bunch of scripts to thoroughly test specific kobuki hardware components.

Testing events

```
$> rosrun kobuki_testsuite test_events.py
```

Testing digital ouputs

```
$> rosrun kobuki_testsuite test_digital_output.py
```

Testing analog inputs

```
$> rosrun kobuki_testsuite test_analog_input.py
```

Testing battery voltage

```
$> rosrun kobuki_testsuite test_battery_voltage.py
```

Testing the gyro

```
$> rosrun kobuki_testsuite test_gyro.py
```

Testing the LEDs

```
$> rosrun kobuki_testsuite test_led_array.py
```

Testing sounds

```
$> rosrun kobuki_testsuite test_sounds.py
```

5.3 Launching the RGBD camera

The openni_kinnect is the stack that contains the Kinect's drivers.

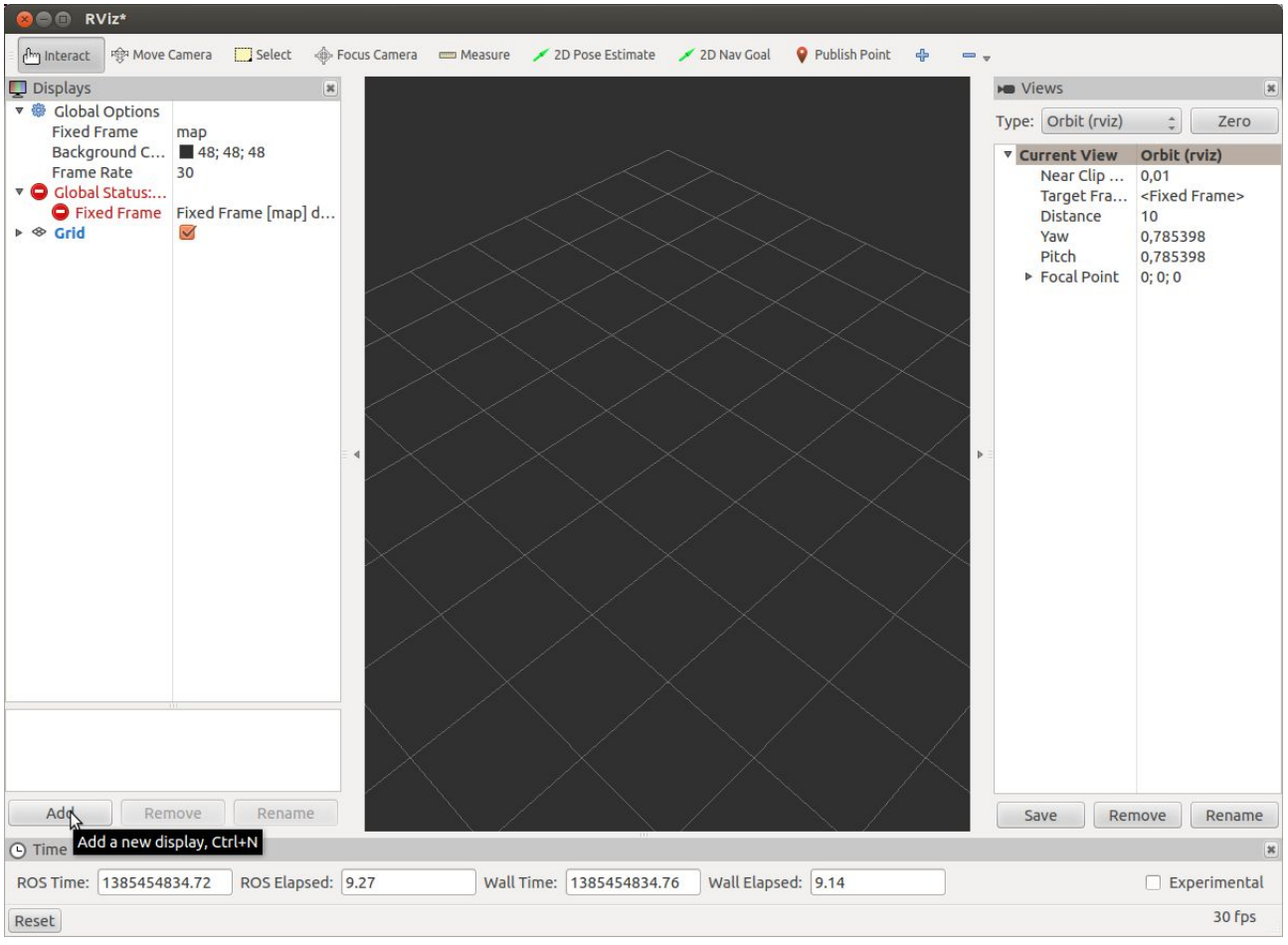
⚠ It is recommended to connect the Kinect's USB into a USB 2.0 slot.

Open a terminal and launch the openni node.

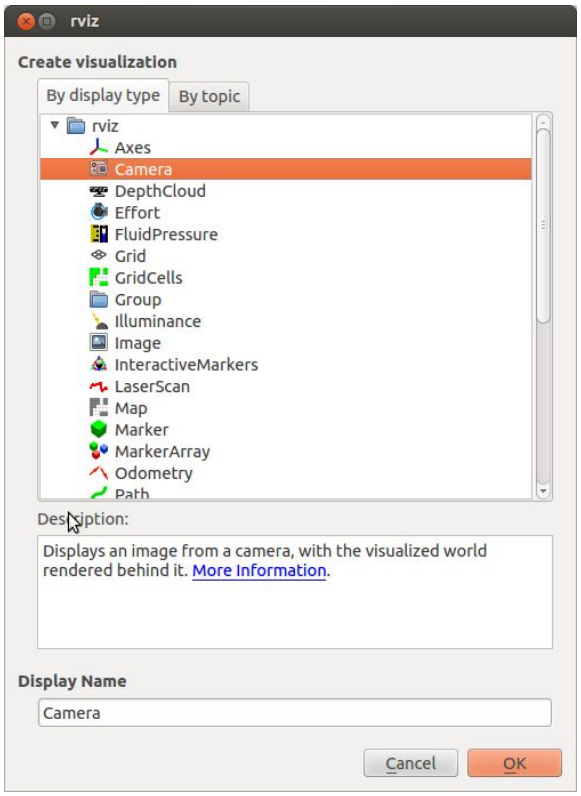
```
$> roslaunch openni_launch openni.launch
or
$> roslaunch astra_launch astra.launch
Or
$> roslaunch turtlebot_bringup 3dsensor.launch 3dsensor:=astra
```

Testing the kinect with rviz

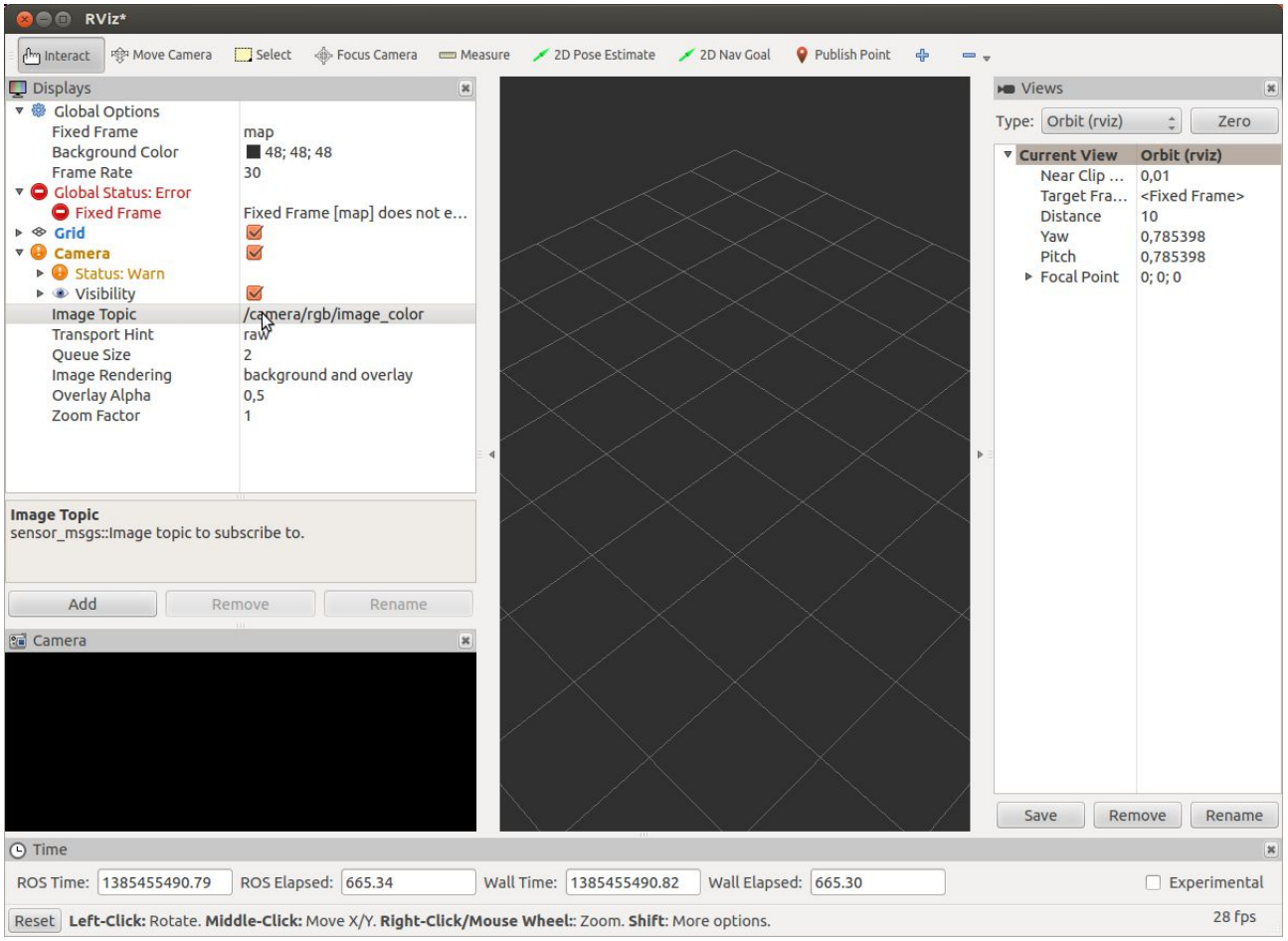
```
$> rosrun rviz rviz
```



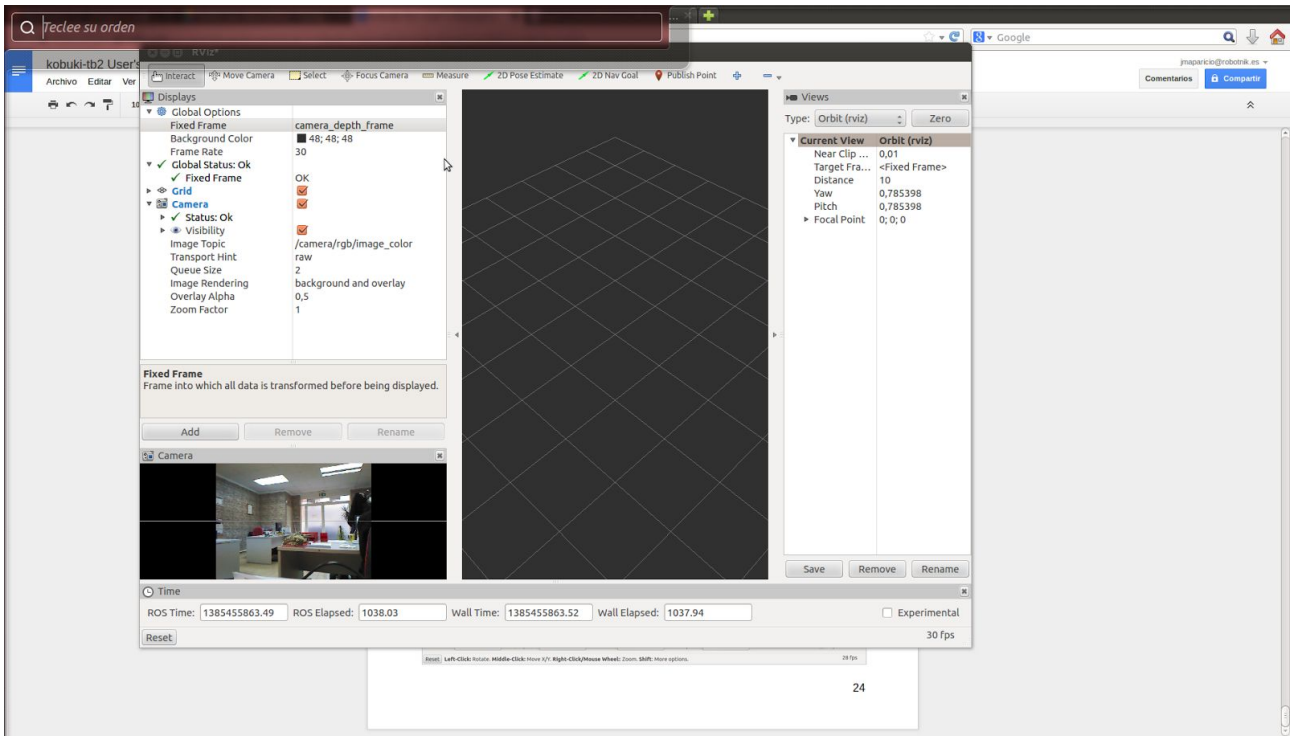
Press Add Button and add a Camera:



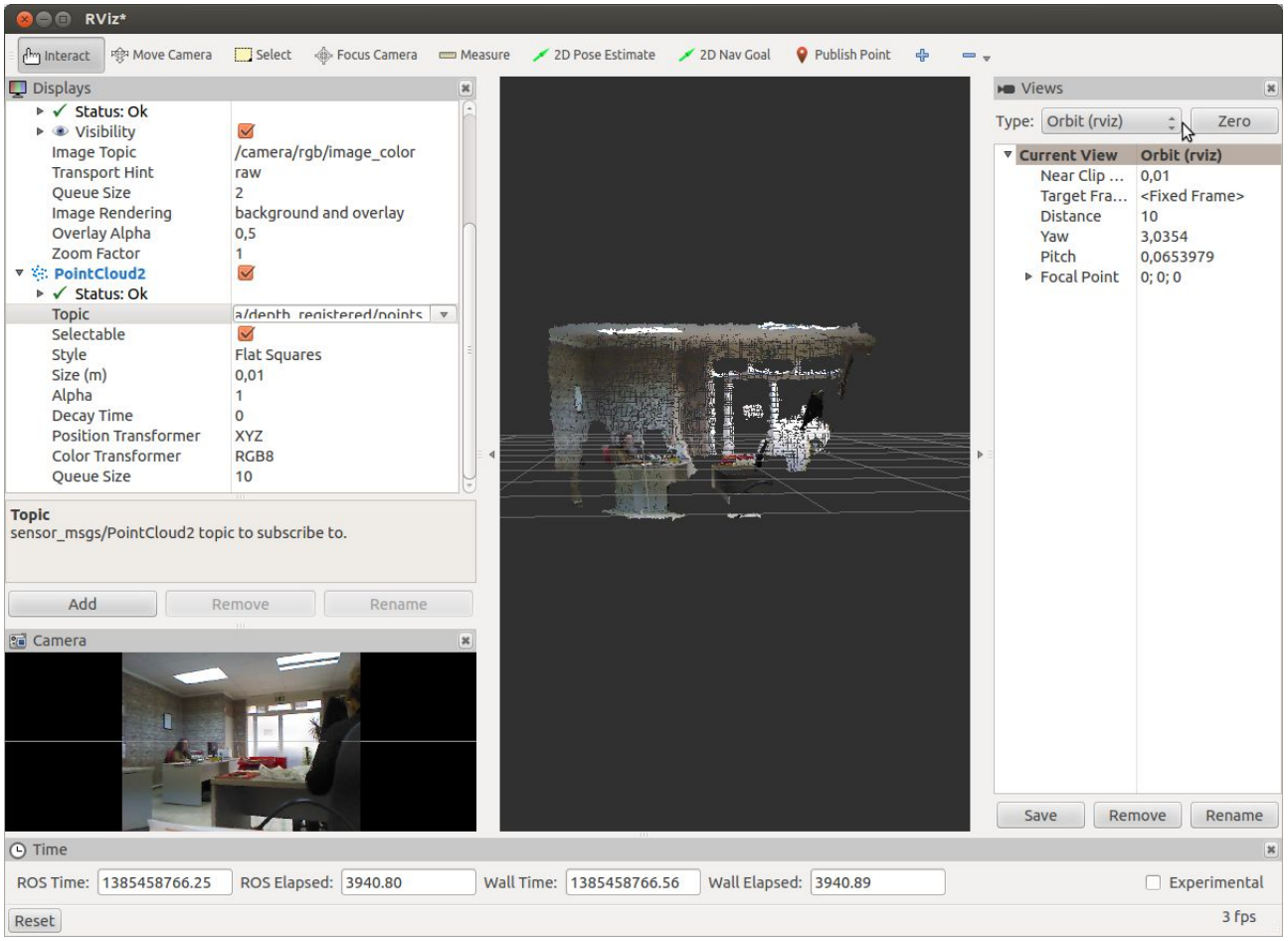
It is necessary to select a topic, for example: /camera/rgb/image_color



And select in Fixed Frame for example: camera_depth_frame



In order to see a point cloud, we have to add a pointcloud2 and select as topic: camera_depth_registered_points



5.3 Controlling the Kobuki robot with a joystick

Install joystick drivers and turtlebot teleop:

```
$> sudo apt-get install ros-kinetic-joystick-drivers
$> sudo apt-get install ros-kinetic-ps3joy
$> sudo apt-get install ros-kinetic-turtlebot-teleop
$> sudo apt-get install sixad
```

5.3.1 PS3 gamepad

It's necessary to make some changes in launch file:

In /opt/ros/kinetic/share/turtlebot_teleop/launch/ps3_teleop.launch

Change the line remap cmd_vel with the following line:

```
<remap from="turtlebot_teleop_joystick/cmd_vel"
to="mobile_base/commands/velocity"/>
```

Start kobuki node:

```
$> roslaunch kobuki_node minimal.launch --screen
```

In order to pair the joystick:

Obviously it's necessary a bluetooth dongle, with the joystick connected with usb cable:

You will see something similar to:

```
Current Bluetooth master: 00:22:b0:d0:5a:09
Setting master bd_addr to 00:22:b0:d0:5a:09
```

Press Ctrl-D now so that you do not continue to run as root.
If you get something like the following:

```
Current Bluetooth master: 00:1b:dc:00:07:3c
Unable to retrieve local bd_addr from `hcitool dev`.
Please enable Bluetooth or specify an address manually.
```

Run the command:

```
$> hciconfig hci0 reset
```

and then retry:

```
$> rosrun ps3joy sixpair
```

Then execute joystick node:

```
$> rosrun ps3joy ps3joy_node.py
```

Unplug the joystick and press pairing button.

Execute turtlebot_teleop:

```
$> roslaunch turtlebot_teleop ps3_teleop.launch
```

Moving left analog joystick and L1 pressed kobuki should move now.

5.3.2 PS4 gamepad

It's necessary to make some changes in the launch file (this assumes that you are modifying the launch file from the original ps3_teleop.launch instead of the one used for PS3):

In /opt/ros/kinetic/share/turtlebot_teleop/launch/ps3_teleop.launch, change the following lines from:

```
<param name="axis_deadman" value="10"/>
<param name="axis_angular" value="0"/>
```

to:

```
<param name="axis_deadman" value="5"/>
<param name="axis_angular" value="2"/>
```

After conducting the above changes, save the launch file as ps4_teleop.launch.

When using the PS4 gamepad for the first time, you can pair your system with the gamepad wirelessly over Bluetooth. The **first** method is conducted via the terminal window. Follow the below instructions in order to install the required Linux driver (ds4drv) on your Linux system (stable version):

```
$> sudo apt-get install python-pip

$> sudo pip install ds4drv
```

Test the connection with:

```
$> sudo ds4drv
```

and then press and hold both the Share and PS buttons for a few seconds to enter the joystick into pairing mode (press the Share button first). The keyword ´Found device´ should appear once pairing is established.

Please note that if the joystick does not pair successfully, it might be caused by Bluetooth on the computer being disabled by default. Perform the following to enable it:

```
$> sudo chmod +x /usr/sbin/bluetoothd
$> sudo service bluetooth restart
```

You can also pair your system with the gamepad (also wirelessly via Bluetooth) directly on the Linux graphical user interface. First put the controller into pairing mode by pressing and holding the Share and PS buttons at the same time (press the Share button first) for a few seconds. Locate your Bluetooth menu on the desktop and click Bluetooth Settings. On the bottom-left corner, click the add icon and wait for the PS4 controller to appear on the found list.

To test controlling the Turtlebot with the PS4 joystick,

Start kobuki node:

```
$> roslaunch kobuki_node minimal.launch --screen
```

Execute turtlebot_teleop:

```
$> roslaunch turtlebot_teleop ps4_teleop.launch
```

First press and hold R1, and then move the Turtlebot by moving the left analog joystick vertically. To perform rotation, move the right analog joystick horizontally (while R1 is still held down).

6. Netbook cable modification

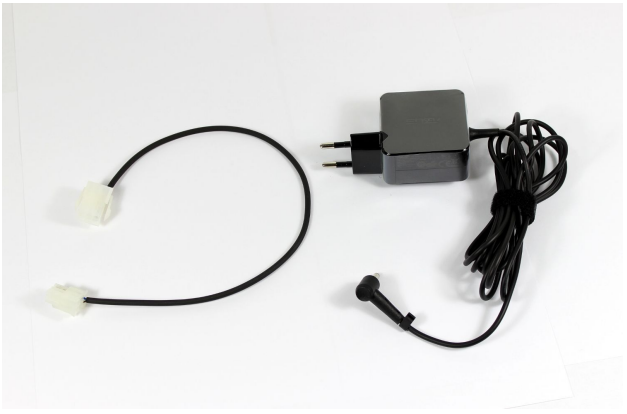
ATTENTION: This mod is OPTIONAL and may void the warranty of the netbook. This mod is made under responsibility of the client.

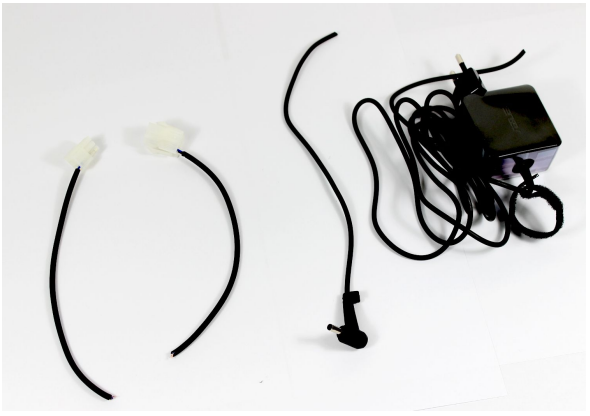
Here you find the instructions to make a custom cable to power your netbook through the Kobuki base.

6.1 Materials

- Heat shrink tube (2mm and 4mm)
- Scissors
- Soldering iron
- Solder
- Heat gun

6.2 Instructions

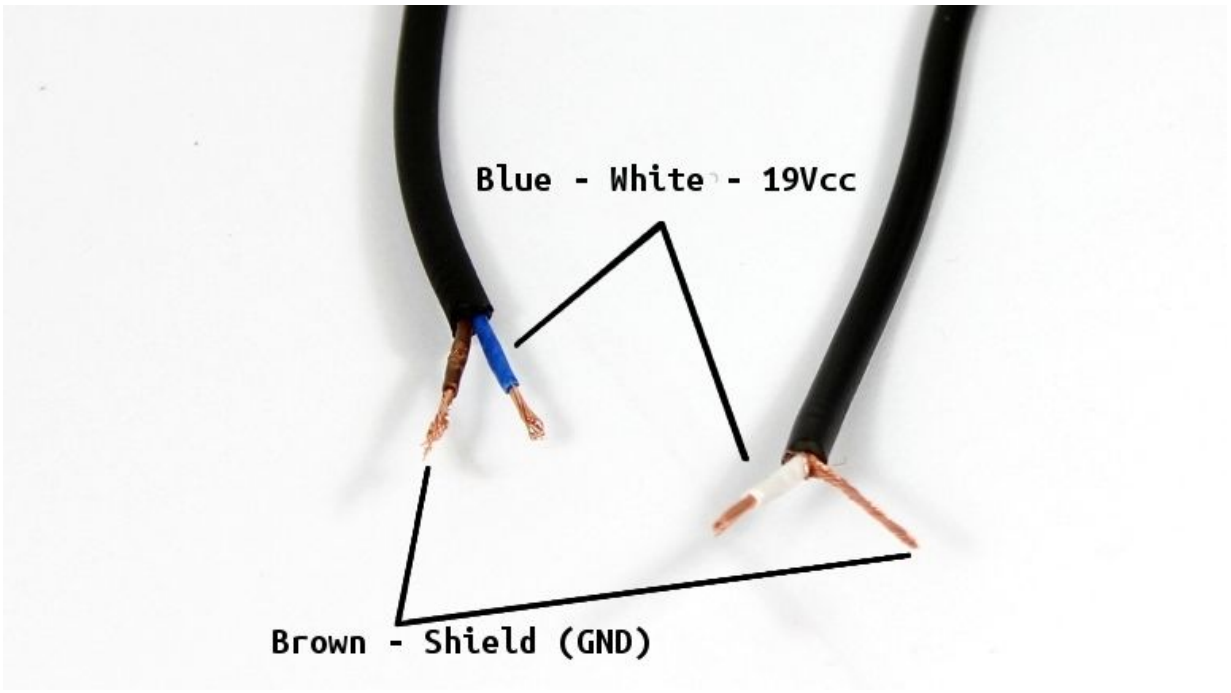




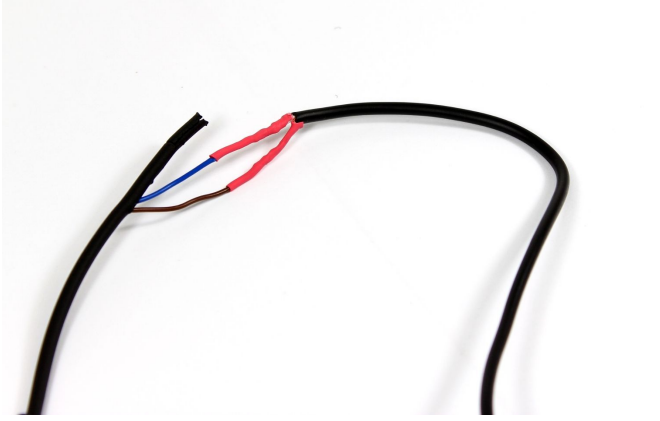
- Cut the cables as seen in the image.

- Male connector goes with Netbook plug

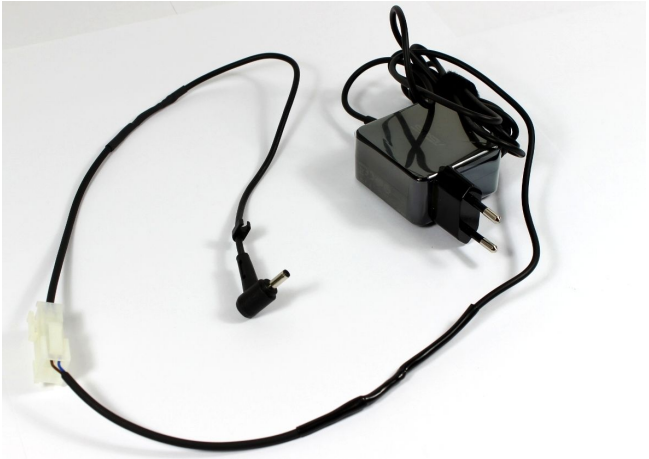
- Female connector goes with DC adapter.



- Strip the cables and the the cable sheath. Pass the heat shrink tubes (first the big one) and match cables as indicated.



- Then solder the cables and heat the shrink tubes. Repeat this with the other cable.



- Test both cables.

⚠NOTE: Cable colours could be different depending on the netbook model. Please check the polarity.

Appendixes

Appendix 1 - Setting up Arbotix board

1. Download Arduino ide from https://downloads.arduino.cc/arduino-1.0.6-linux64.tgz

   wget https://downloads.arduino.cc/arduino-1.0.6-linux64.tgz
1. Extract to a folder.
2. Download firmware archives from
   https://github.com/trossenrobotics/arbotix/archive/master.zip
   wget https://github.com/trossenrobotics/arbotix/archive/master.zip
3. Extract to ~/Documents/Arduino
4. Open arduino
   a. Cd ~/Downloads/arduino-1.0.6
   b. ./arduino
5. Open Arduino IDE and change folder to /Documents/Arduino/arbotixmaster
   a. File->Preferences->Sketchbook Location
   b. Tools->Board->Arbotix
   c. Tools->Serial Port->/dev/ttyUSBX
   d. File->Sketchbook->Arbotix Sketches ->ros
   e. Verify + Upload
6. The Arbotix is ready to work with ROS