**Hardware requirement**

* Baxter Robot
* Ubuntu system computer
* Exoskeleton

## Setup of a development workstation

## Ubuntu (tested on 16.04 <http://releases.ubuntu.com/16.04/>)

## ROS (tested on kinetic <http://wiki.ros.org/kinetic>)

## Baxter SDK (<http://sdk.rethinkrobotics.com/wiki/Workstation_Setup>)

**Required Python packages (tested on python 2.7)**

* Pyserial (<https://pypi.org/project/pyserial/>)
* Times (<https://pypi.org/project/times/>)
* Math
* Numpy (<https://scipy.org/install.html>)

**Files involved in this research**

|  |  |
| --- | --- |
| File name | Description |
| Exoskeleton\_reader.py | creates ROS node "Exoskeleton" that is streaming all the exoskeleton data in a form of exoskeleton.msg |
| Teleoperation\_baxter.py | creates ROS node "teleoperation" that subscribes to exo\_info and commands Baxter robot with the selected mode |
| exo\_info.msg | ROS message |
| Exo\_read.ino | Arduino code |

## Instruction to run the teleoperation

In order to run the system, you can either run on the Gazebo simulation or Real Baxter robot. First-connect the exoskeleton to the ubuntu system computer via USB cable.

The directory of the USB can be found by typing ls /dev on the terminal. (normally /dev/ttyACM0, depends on the order when you plug USB device to your computer) you have to do it every time you plug this exoskeleton to the computer

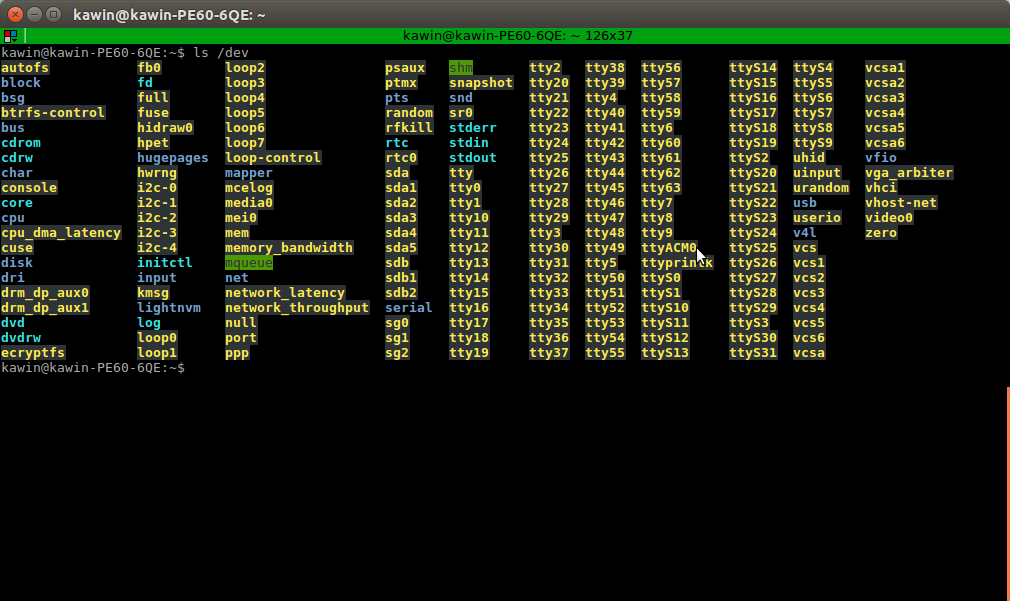


Figure 1: Screen shot of the list of USB device

Then, open terminal and allow permission on the exoskeleton device. By using $ sudo chmod a+rw /dev/ttyACM0

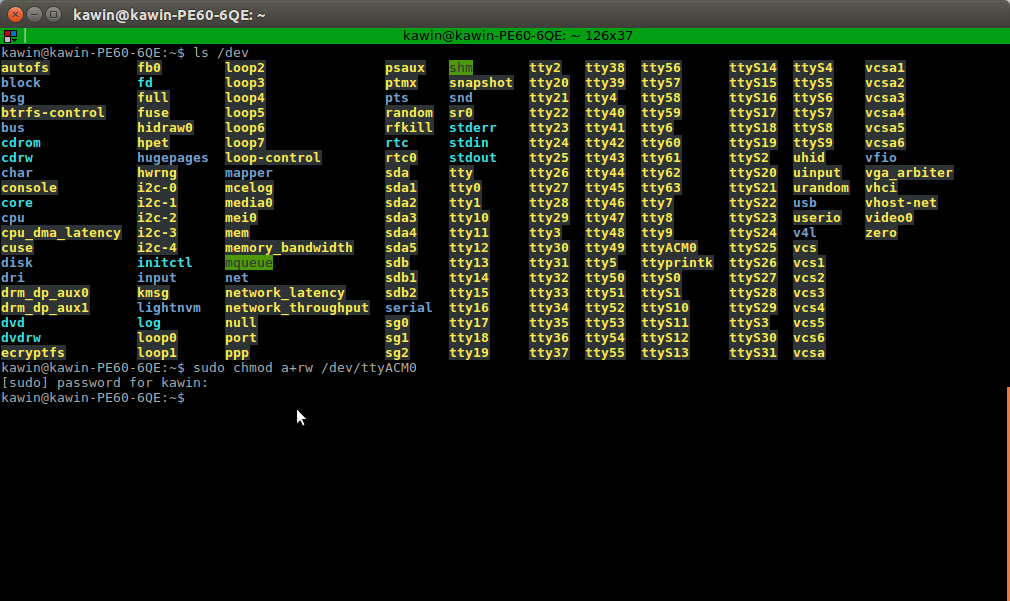


Figure 2: Screen shot of allowing a permission

Then, on the terminal, navigate the directory to the Catkin workspace where Baxter.sh is located. Type on the terminal $ ./baxter.sh sim (for simulation) $ ./baxter.sh (for teleoperate on real Baxter)

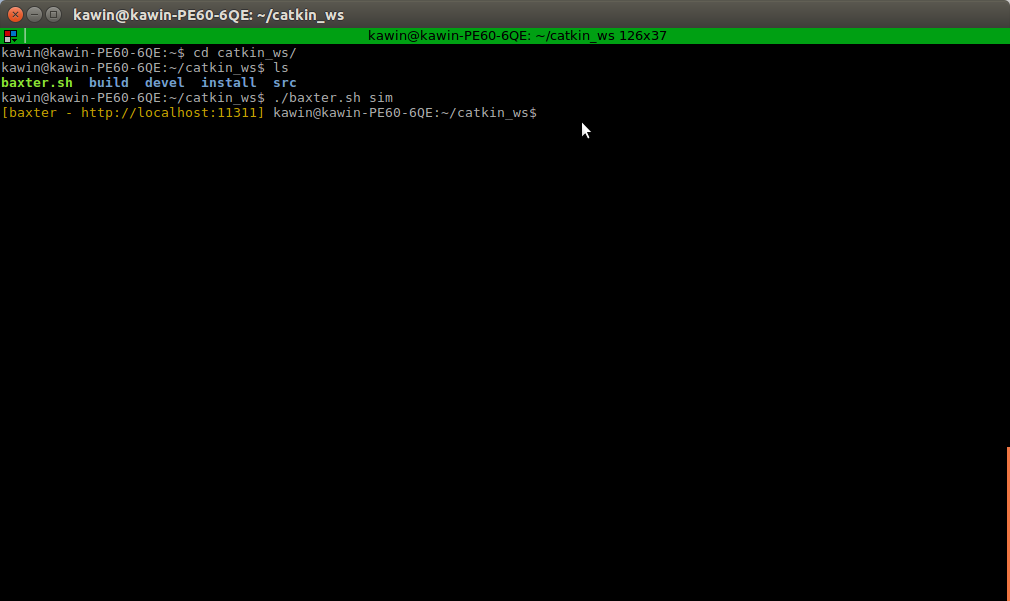
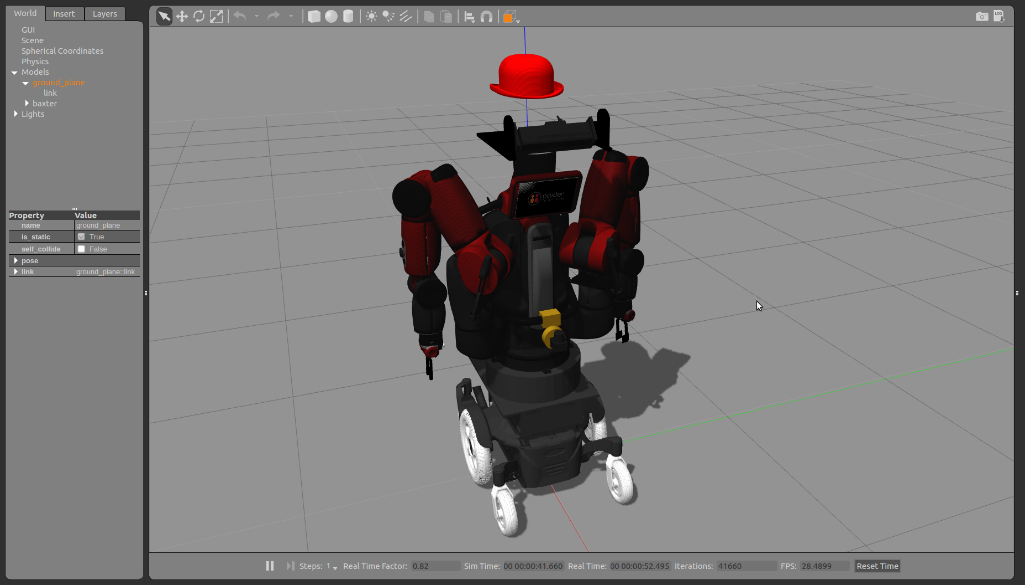
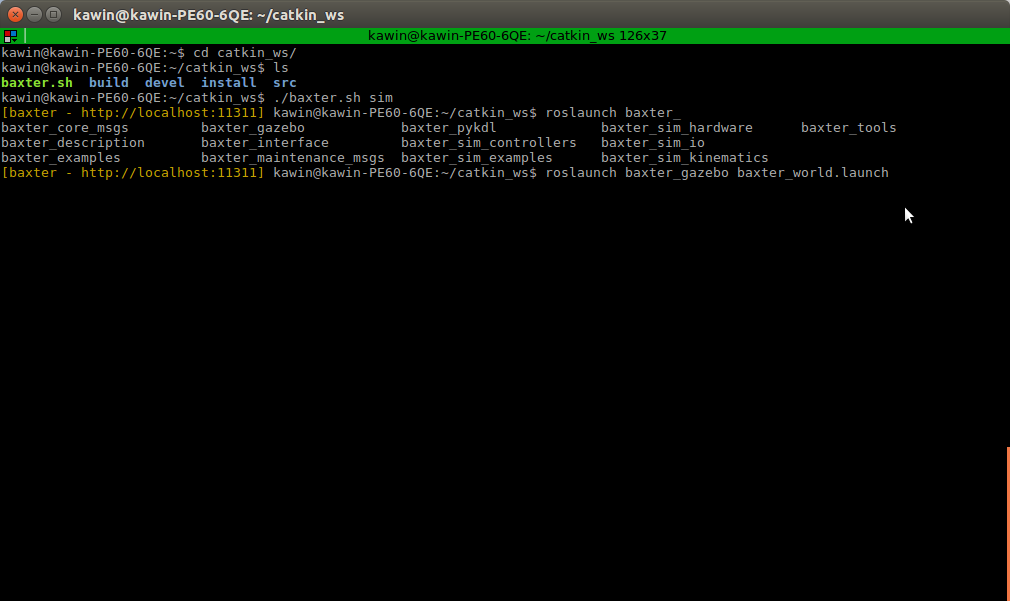


Figure 3: Screen shot of running baxter.sh file

If you choose to run it on the simulation type this on the terminal that already opened the Baxter.sh file $ roslaunch baxter\_gazebo baxter\_world.launch

Figure 4: Screen shot of launching the Gazebo simulation



Then, start reading the information from exoskeleton by running the node call exoskeleton reader

$ rosrun exoskeleton exoskeleton reader

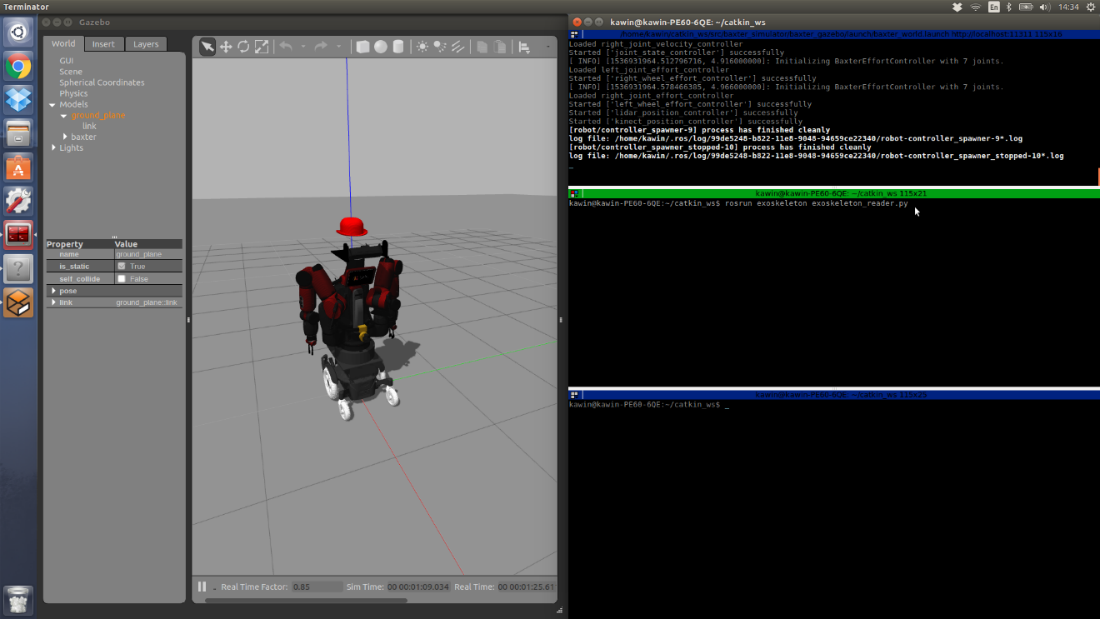


Figure 5: Screen shot of running an exoskeleton reader

Then, to start the teleoperation, run another node call exo-teleoperation.

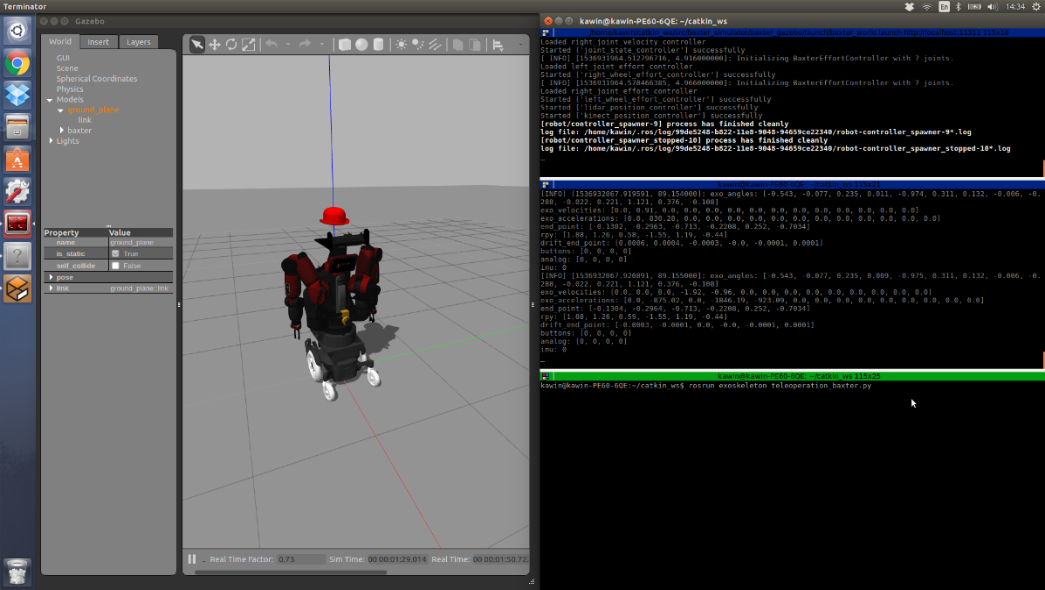


Figure 6 : Screen shot of running a teleoperation

Then, this is the screen when everything is success.

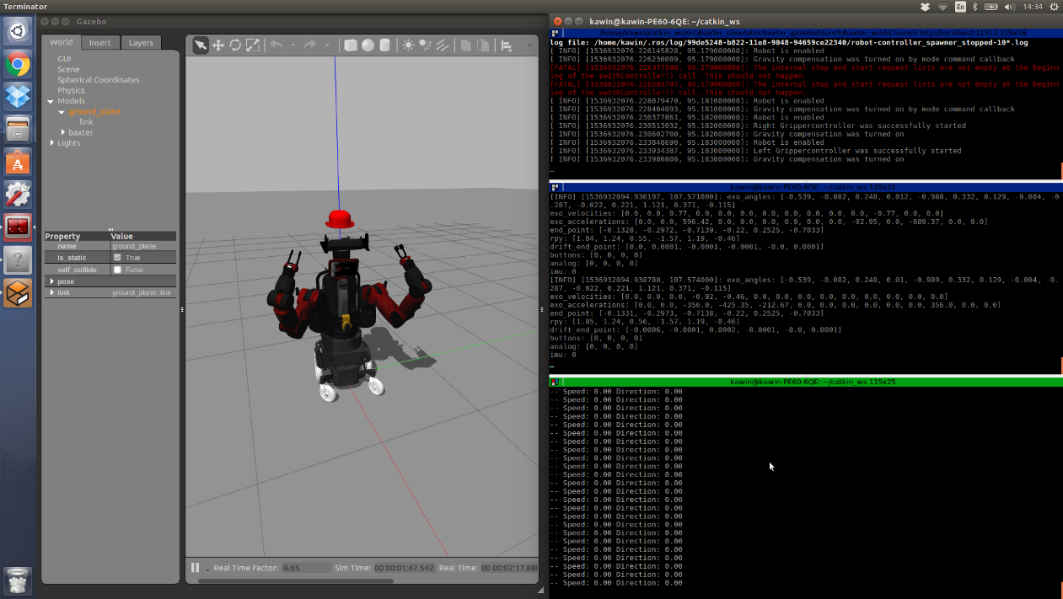


Figure 7: Screen shot of successfully teleoperate

## Instruction to change the mode of teleoperation

There are 2 main modes for the Baxter robot to be controlled by the exoskeleton.

* Controlled with the joint space mapping algorithm
* Controlled with the Cartesian mapping algorithm

All parameters that are adjustable for this teleoperation are only located in teleoperation\_baxter.py. Changing the parameter “mode” in to change the mode.

In joint space mapping algorithm, the scaling factor can be modified by changing the multiplier in the angle mapping function. Both arms can be differently tuned with various scaling factors. (parameters can be seen in line 273-279, 300-306 teleoperation\_baxter.py)

In Cartesian mapping algorithm, there are 2 predefined postures to be mapped. Changing the parameter mode to either 2 or 3 to select desired mode. The detail is commented in the python script. (parameters can be seen in line 39 teleoperation\_baxter.py)