Intel NUC Setup Instructions for the Beluga AUV

Installing Ubuntu

TODO

Enable Auto-Reboot

By default the Intel NUC will not power on after being connected to power, and will wait for the power button to be pressed. We can change this in the BIOS settings.

1. Hold F2 while the NUC is booting up to boot into the BIOS
2. From the drop-down menu select "Power"
3. Change the "After Power Failure" option to "Power On"
4. Hit F10 to save and exit
5. Test the changes by powering the machine off, unplugging the power cord, and plugging it back in. The NUC should reboot

Set Up Static-IP and SSH

We want the NUC to have the same IP address every time we connect to the router. To do this, use the instructions here: `https://help.ubuntu.com/stable/ubuntu-help/net-fixed-ip-address.html`. The IP address should be 192.168.1.[NUM], where NUM is outside the 100-149 range for the DHCP table (I used 200 for dolphin). Netmask should be 255.255.255.0, and the gateway should be 192.168.1.1. The DNS servers IP should also be 192.168.1.1. Reset the wifi connection for this to take effect.

To set up SSH, use the following two commands:

```
sudo apt-get install openssh-server
sudo restart ssh
```

Test that this worked by opening an ssh connection from another machine using the new static IP address:

```
ssh -Y [USERNAME]@192.168.1.[NUM]
```

Installing ROS

TODO

To allow the NUC to send ROS messages over wifi, add the following line to your .zshrc/.bashrc

```
export ROS_IP=[BELUGA_STATIC_IP]
(i.e. export ROS_IP=192.168.1.200)
```

Cloning the Beluga Repositories

There are two repositories to clone, one for the ROS workspace and one for the data and simulation. Clone them into the home directory:

```
cd
git clone https://github.com/TechmationClinic2017/beluga_ws.git
git clone https://github.com/TechmationClinic2017/BelugaSimulation.git
```

The ROS repository needs to be built, but has dependencies that don't come with the standard ROS install. Install those with the following commands:

```
sudo apt-get install ros-indigo-robot-localization
sudo apt-get install ros-indigo-rosserial
```

Some of the scripts also need the python package scipy. Use pip to install:

```
wget https://bootstrap.pypa.io/get-pip.py
sudo -H python get-pip.py
sudo -H pip install scipy
```

Now, build the Beluga ROS workspace:

```
cd ~/beluga_ws
catkin_make
```

Don't forget to source the correct setup file for your shell in `beluga_ws/devel` (from your .bashrc, .zshrc, or .cshrc, depending on your shell). Make sure that ROS knows about the new packages by trying `roscd beluga_main`, or by running any of the launch files.

Configuring USB Ports

USB Permissions
The VectorNav, GPS, and battery all connect over USB. To be able to read from USB ports in `/dev/`, we need to add the user to the dialout group. Do this with the following command:

```
sudo adduser [USERNAME] dialout
```

Log out and back in for this to go into effect. Check that this worked by making sure there is no "Permission Denied" error on the following command:

```
cat /dev/ttyUSB0
```

Static USB Ports
The physical USB ports are not guaranteed to map to a specific port in `/dev/`, and are instead mapped to `/dev/ttyUSB[PORT_NUM]` in the order that they are connected. To get around this, we can tell the system to create symlinks in `/dev/` whenever a specific device is connected. To do this, we need each USB device's idProduct number, idVendor number, and serial number. The process is as follows:

1. Determine which port in `/dev/` the device is connected to (easiest to just see which port gets added when the device is plugged in).
2. Get the idProduct, idVendor, and serial numbers which the following commands (substitute your port for /dev/ttyUSB0 if using a different port)
   - `udevadm info -a -n /dev/ttyUSB0 | grep "idProduct" | head -n1`
   - `udevadm info -a -n /dev/ttyUSB0 | grep "idVendor" | head -n1`
   - `udevadm info -a -n /dev/ttyUSB0 | grep "serial" | head -n1`
3. Create a new file in `/etc/udev/rules.d` named `99-serial-usb.rules`
4. Add a rule as a new line in the file as follows:
   - Assuming you're adding a device with idVendor number 0403, idProduct number 6001, and serial number A103W21E (this is one of our VectorNavs), with a desired symlink to `/dev/imu`, the command is:
   - `SUBSYSTEM=="tty", ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6001", ATTRS{serial}=="A103W21E", SYMLINK+="imu"`
   - Change the numbers and symlink around for different devices. Each device should get one line in `99-serial-usb.rules`

Set this up for the VectorNav and GPS that will be used with the NUC, and test it by reconnecting the devices and checking if the symlinks show up in `/dev/`. I put the VectorNav on `/dev/imu` and the GPS on `/dev/gps`.

Check that the USB ports work with the ROS repository
Once the USB ports are configured, ensure that we can publish data from those ports with the following two commands:

```
roslaunch beluga_nav vn100.launch port:=/dev/imu
roslaunch beluga_nav gps.launch port:=/dev/gps
```

There should be no errors and IMU/GPS data should begin to be published on their respective ROS topics.

Configuring Startup Script

The startup script is located in the beluga_ws repository at `beluga_ws/scripts/start_beluga`. It assumes the user is using zsh (change in necessary). To launch this file when the beluga boots, add a cronjob by typing `crontab -e` and adding the following line at the bottom of the crontab file:

```
@reboot sudo -u [USER] [PATH_TO_START_BELUGA]
```

I generally like to have the start_beluga file in `$HOME/bin`. On dolphin, the line is:

```
@reboot sudo -u dolphin /home/dolphin/bin/start_beluga
```