

# TESR\_iron-X Tutorial

## (Basic)

[Install Ubuntu using VMware player](#)

[Installation of ROS Noetic on Ubuntu](#)

[iron-X Setup](#)

[Basic File Transfer to iron-X](#)

[How to launch iron-X ROS package](#)



## Ubuntu and ROS Noetic Installation

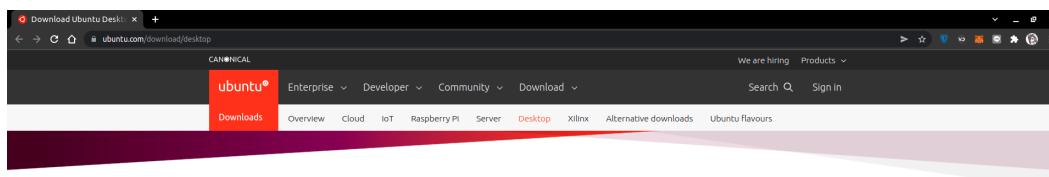
\*You can skip this Installation if you already have Ubuntu 20.04 LTS and ROS Noetic on your own PC/Laptop. Go to the Main Topic by [“Click Here”](#)

### Install Ubuntu using VMware player

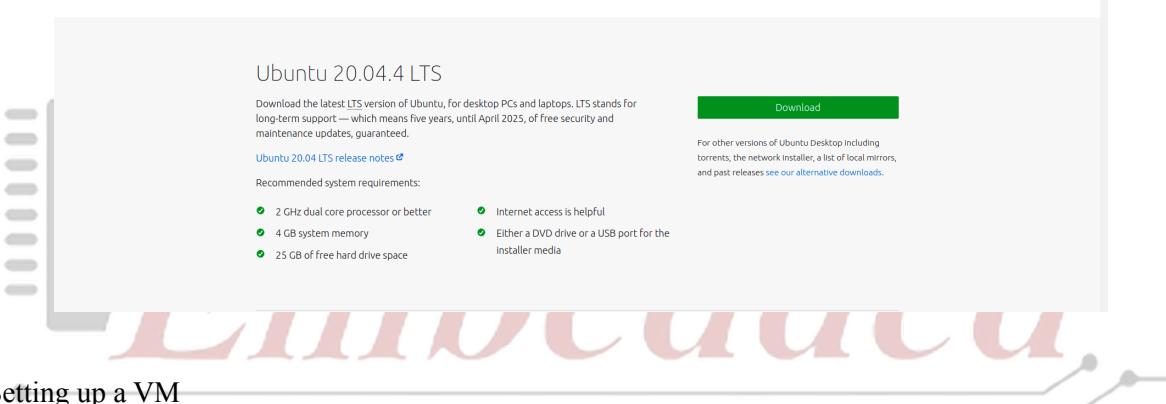
#### 1. Download Ubuntu Image

You can download an Ubuntu image at link: <https://ubuntu.com/download/desktop>

Make sure to save it to a memorable location on your PC! For this tutorial, we will use the Ubuntu 20.04 LTS release.

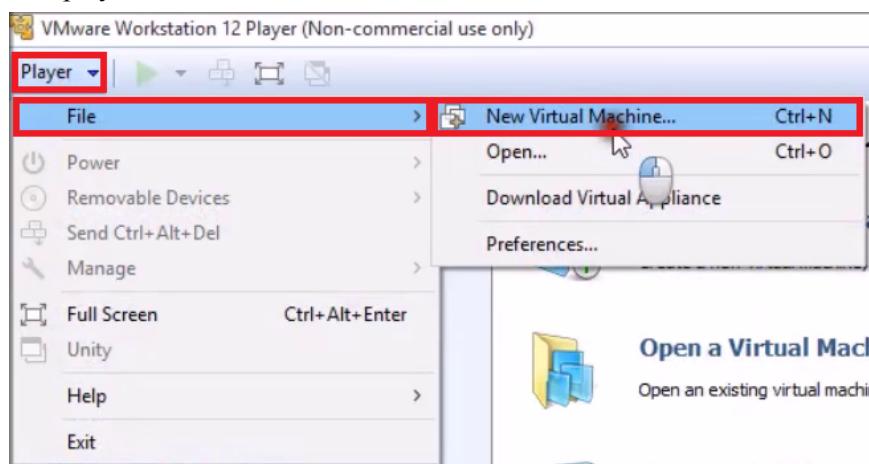


Download Ubuntu Desktop

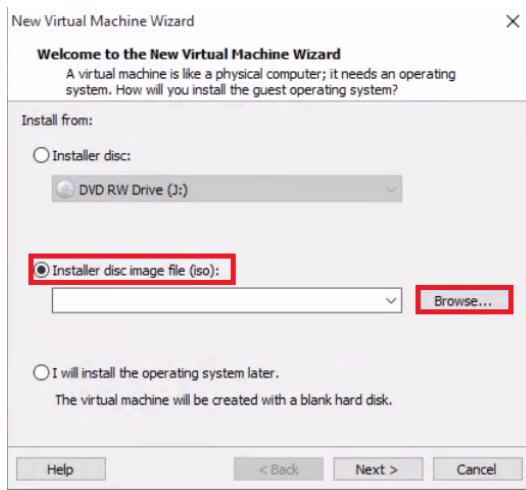


#### 2. Setting up a VM

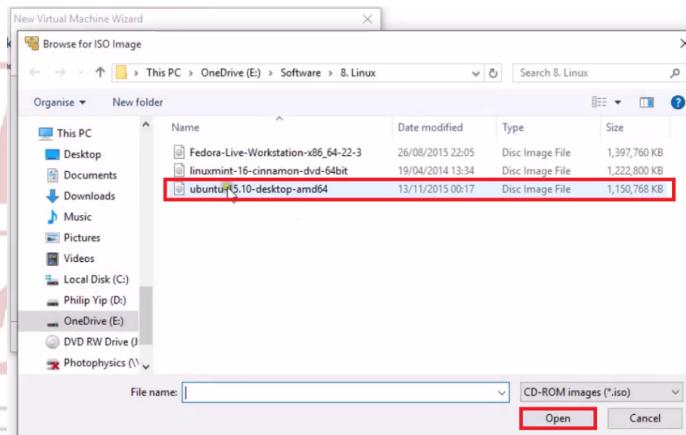
Left click player → Left Click File → Left Click New Virtual Machine:



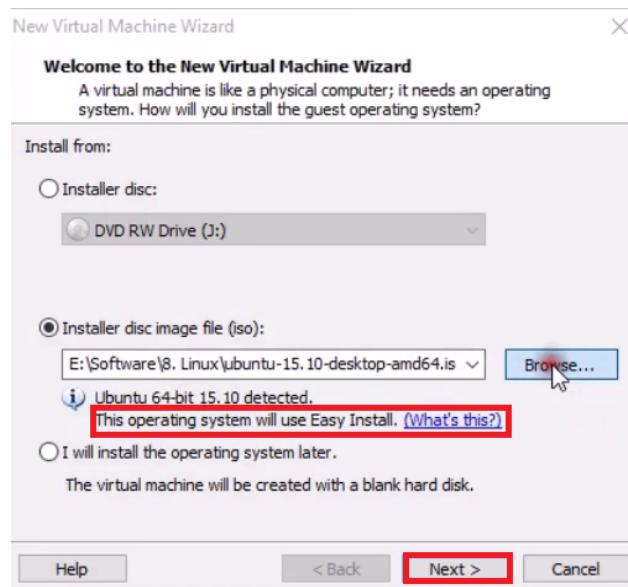
Select Installer Disk Image File (.iso):



Select browse.... and left click the Ubuntu .iso you downloaded:\



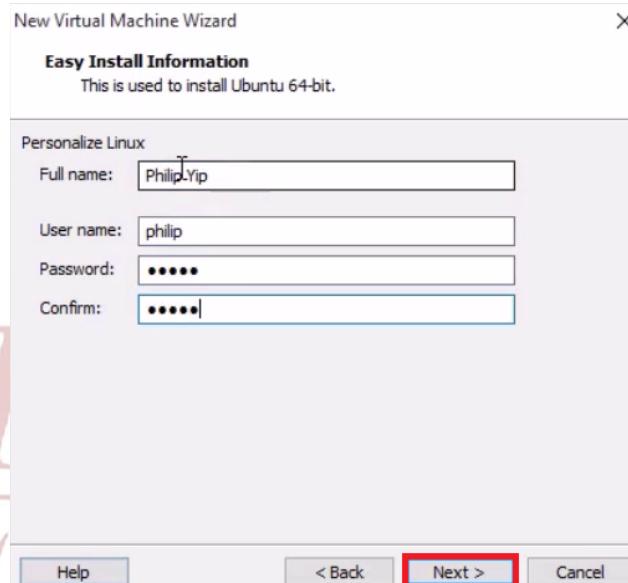
Select Open



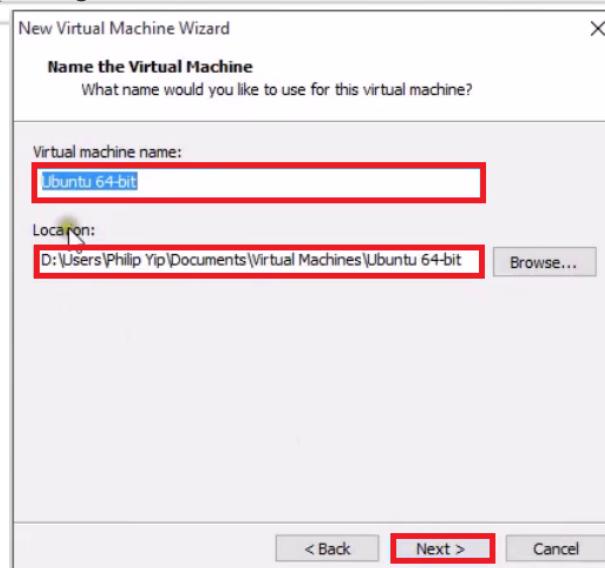
Ensure "This Operating System will use Easy Install" is shown.

With Easy Install VMWare Player will automatically install Ubuntu and VMWare tools without the need for user input. If easy\_install is not selected one may have difficulty installing VMWare tools manually. The only disadvantage of easy install is there is no option to select language options using VMWare player and it defaults to English US language and keyboard :( . The language settings for both can easily be changed after installation and I will demonstrate the change of English US to proper English UK.

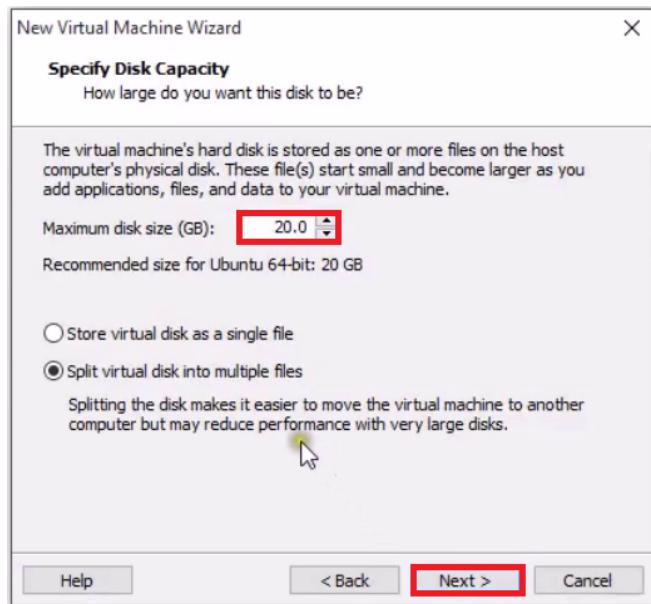
Type in your full name and username. The username has to all be lower case with no spaces. Then type in your password, Ubuntu won't let you install without a password:



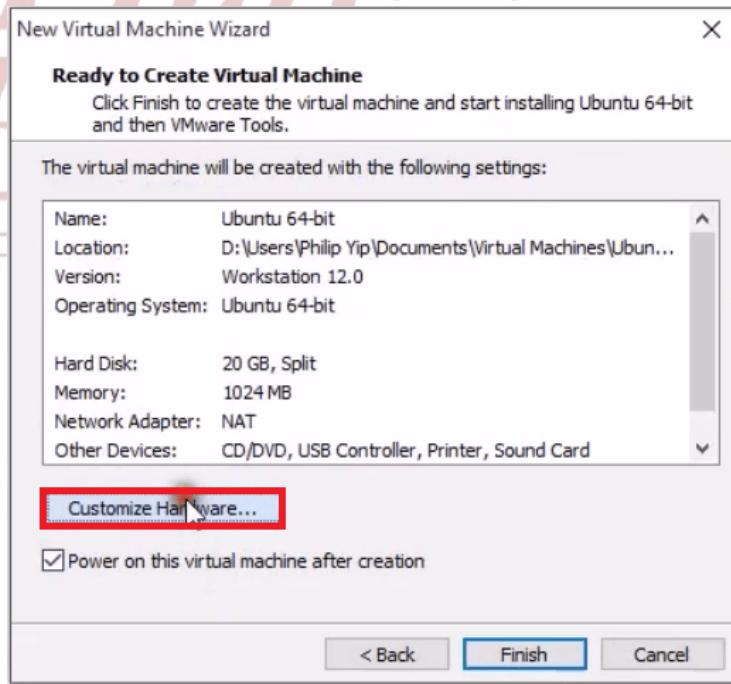
Here you can name your VM and you can also opt to change the location of the VM e.g. to a different HDD/SSD depending what suits.



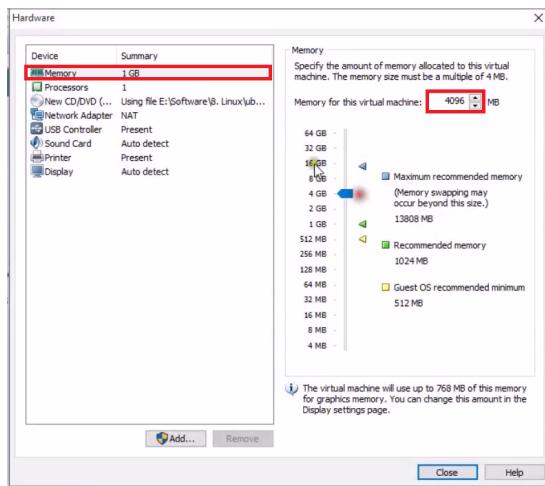
Select next. The default storage is 20 GB but you may increase it. I recommend leaving Split virtual disk into multiple files.



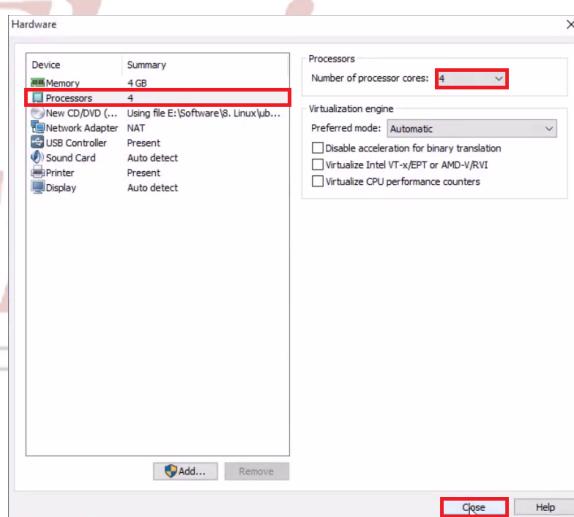
Before installing Ubuntu select Customize Hardware because the default hardware settings are pretty weak:



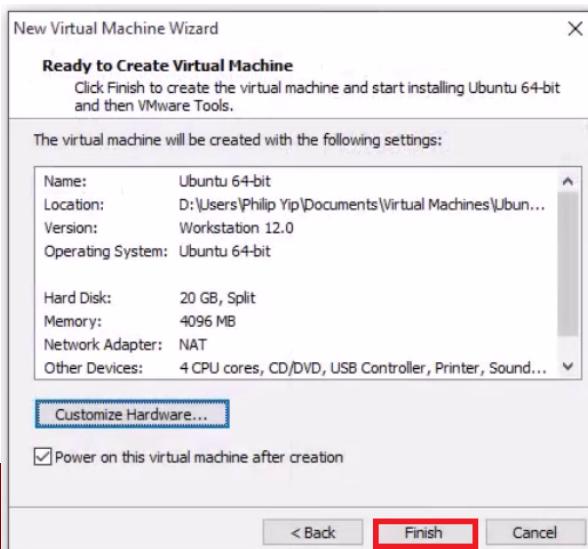
If your computer is capable (has at least 8 GB of RAM) up the RAM to 4 GB:



If your computer is capable of at least 8 threads, change the Number of processor cores to 4. If your processor has 4 threads change the Number of processor cores to 2. This will substantially increase the Ubuntu VMs system performance.

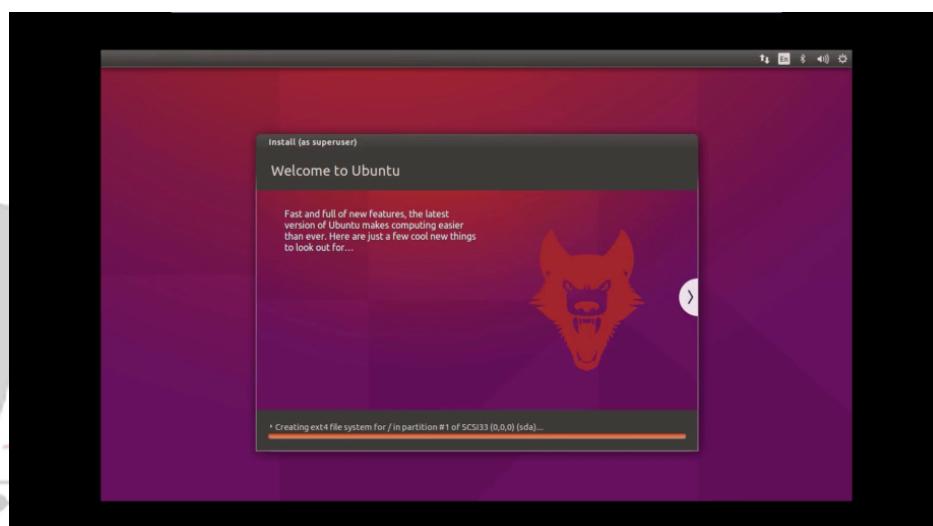
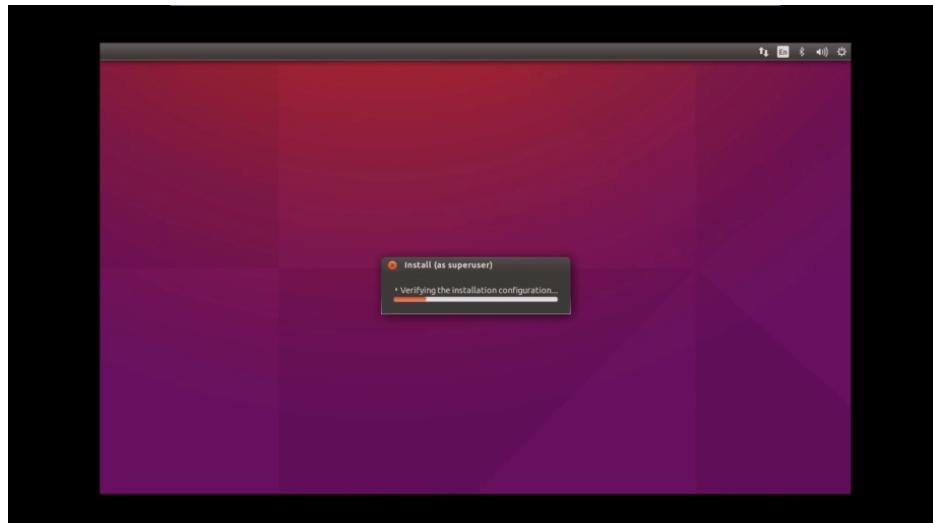


Select close and then Finish



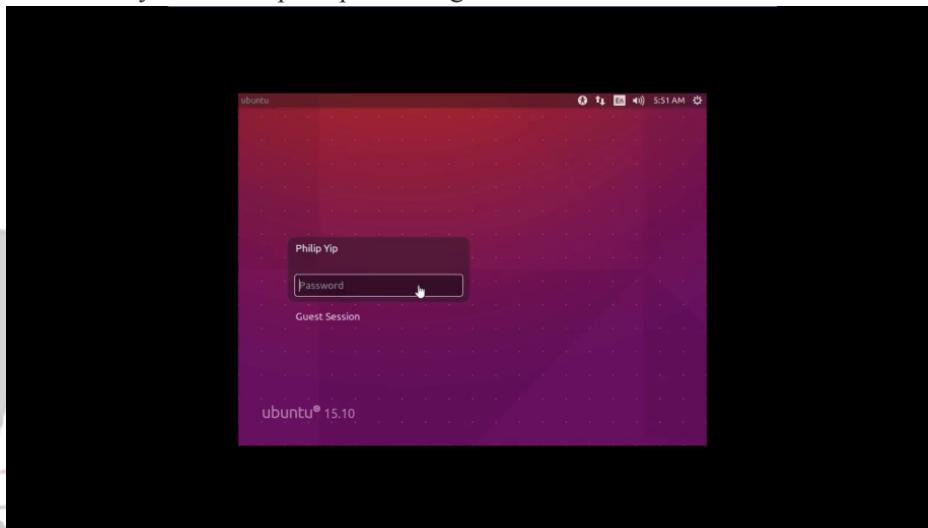
As mentioned the rest of the install will be automated. Make yourself a cup of tea and come back in 5-10 minutes:





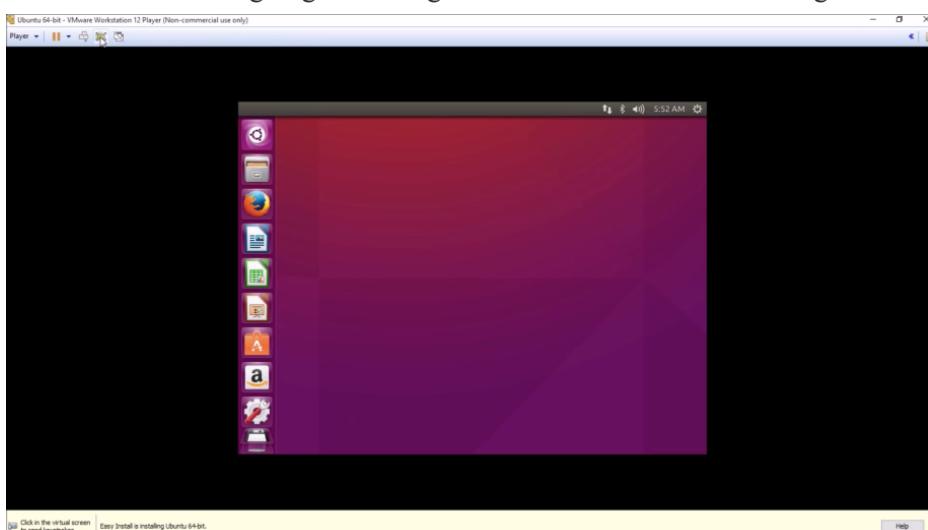


When the install is done you will be prompted to login:

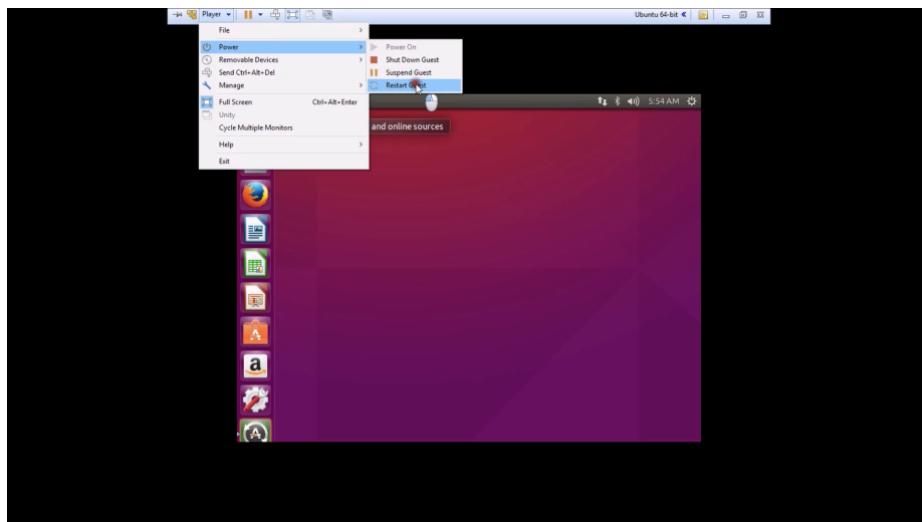


### Making Sure VMWare Tools are Installed

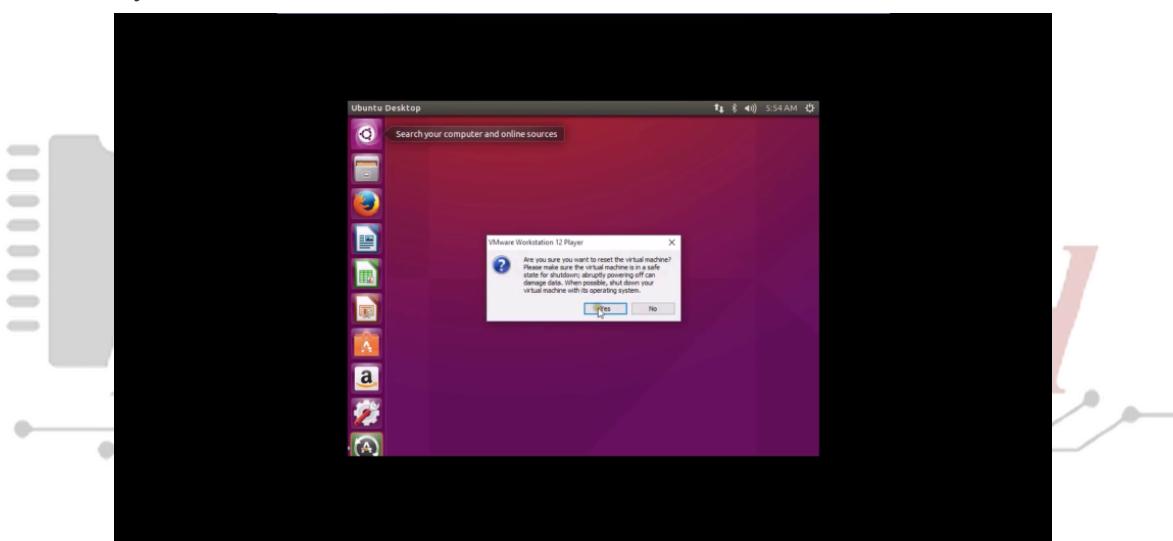
During the first launch the user experience may be terrible because VMWare tools are not installed therefore I recommend signing in waiting 5-10 minutes and then restarting:



Select the player button at the top left and then power and then restart guest:

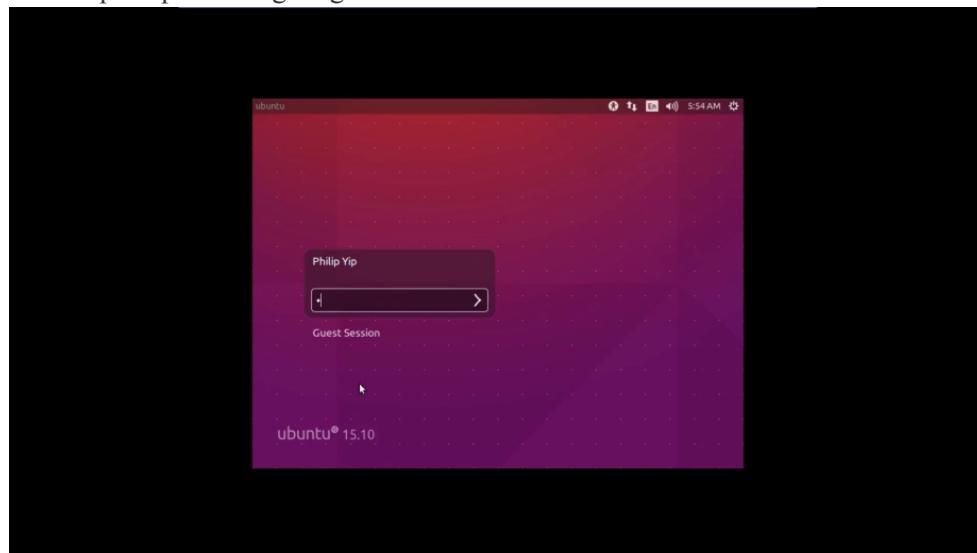


Select yes:

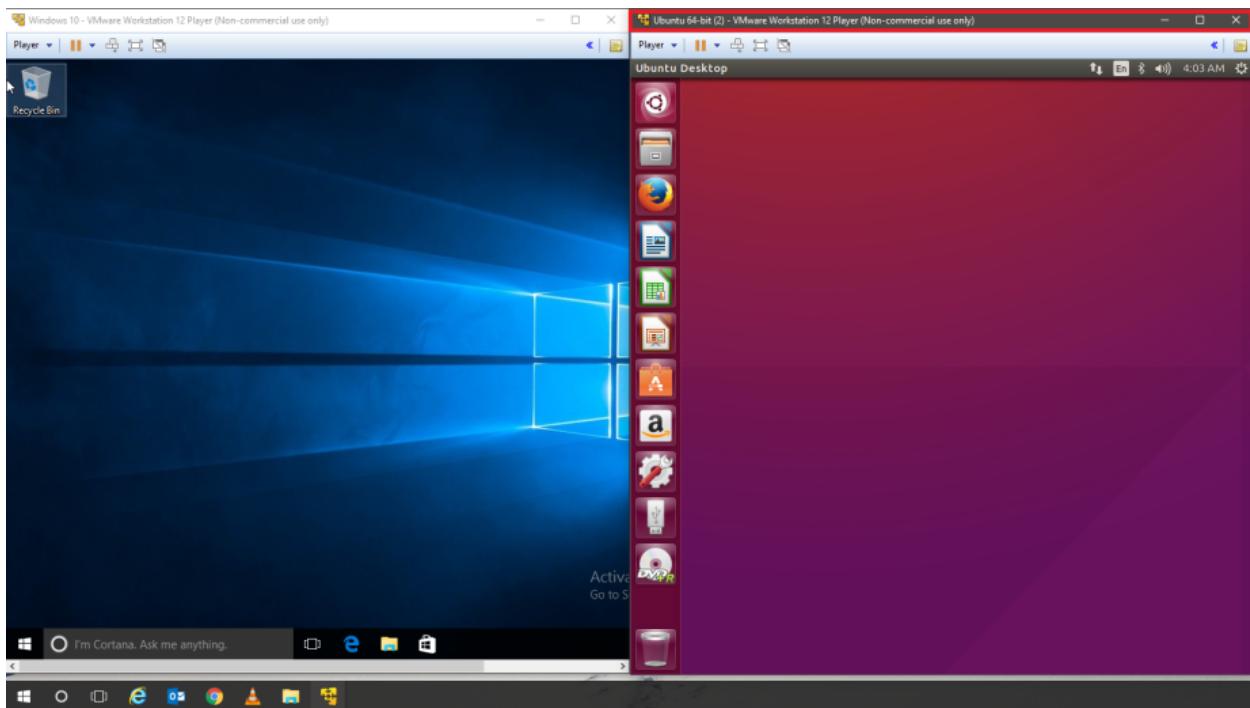




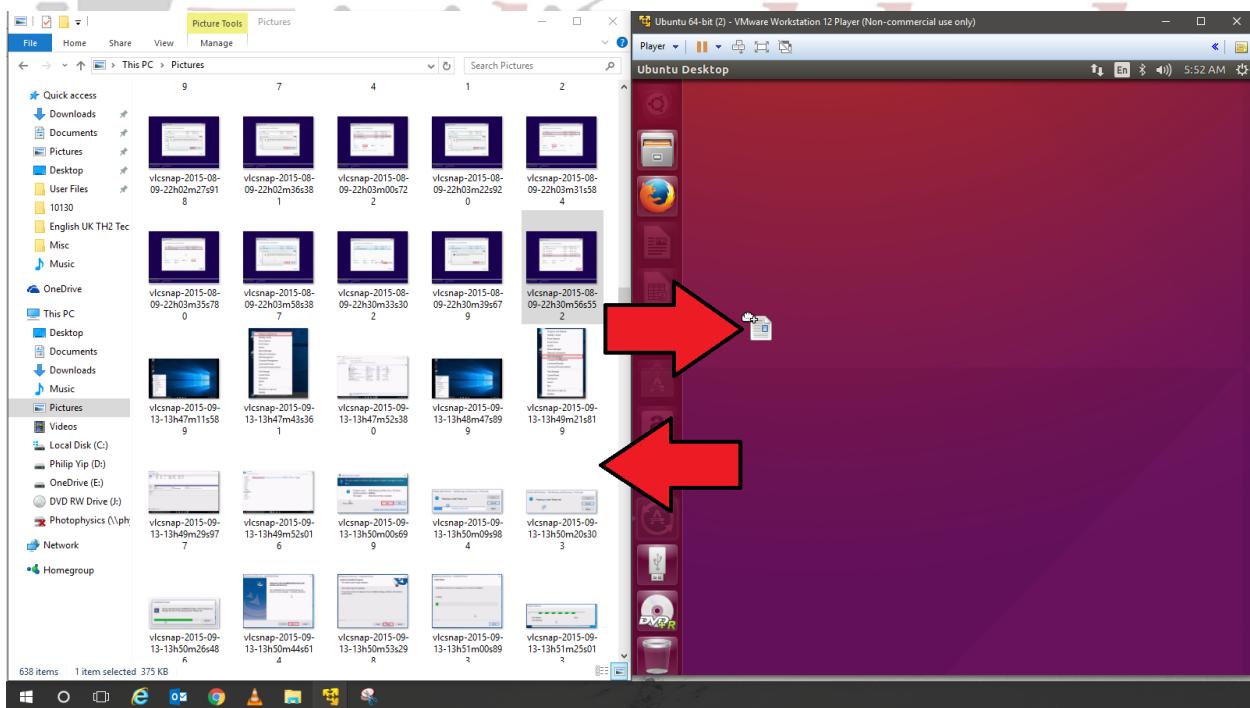
You will be prompted to login again and now it should be full screen:



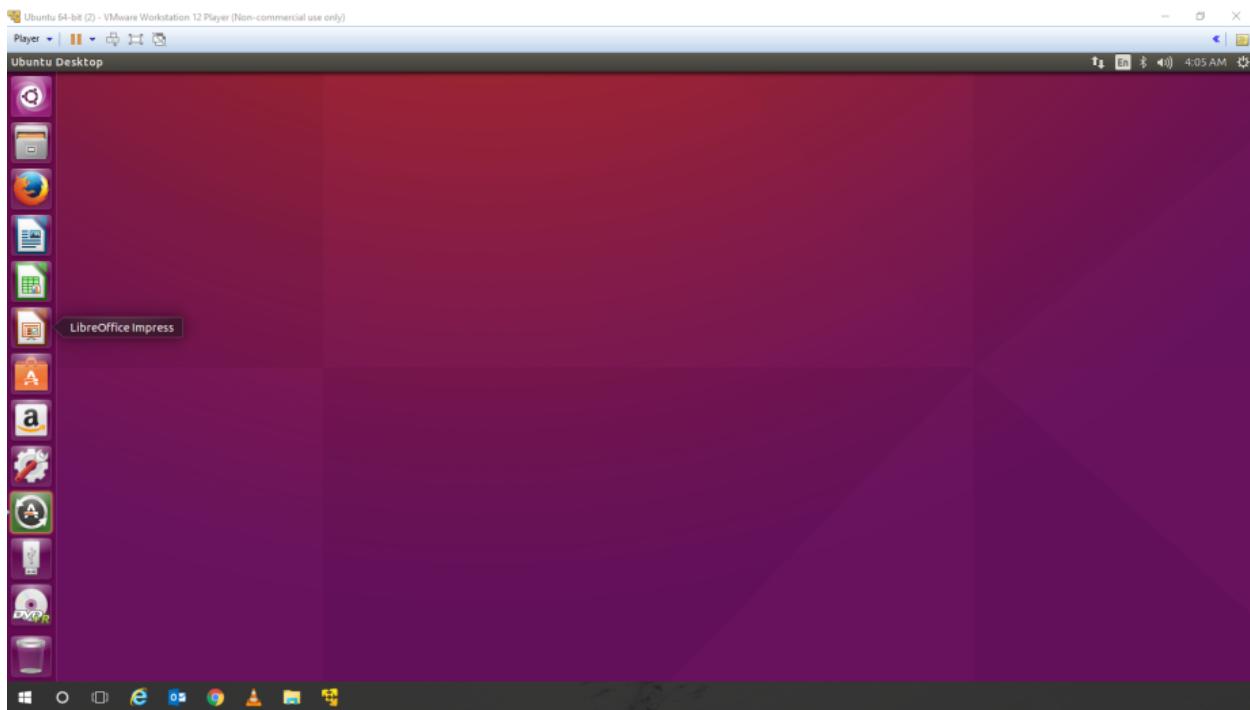
The VM is now full screen if it does not drag and drop the VMWare player Window to the left or right to aero snap half the screen. The VM should automatically resize to fill the Window:



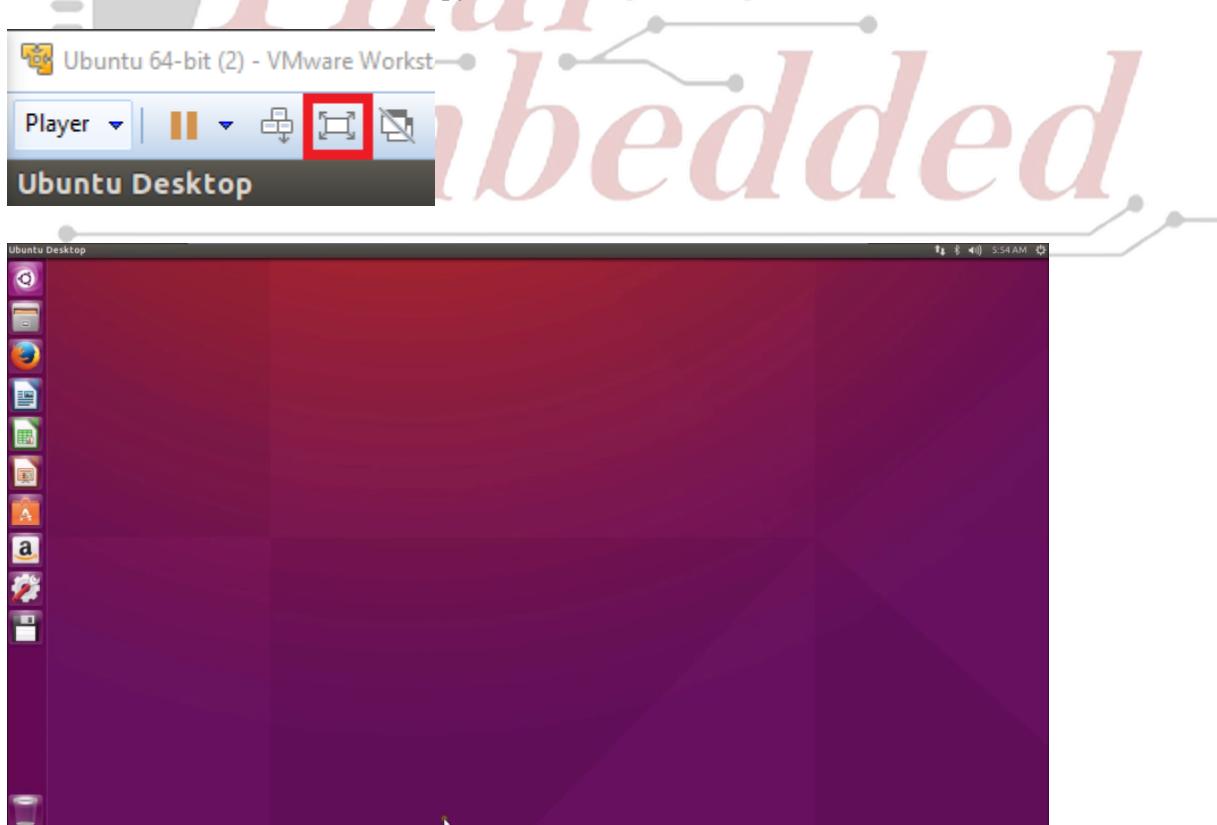
This also allows 2 way drag and drop from the host to the VM:



You can also aero snap it to the top. The VMWare player Window will then be maximized:

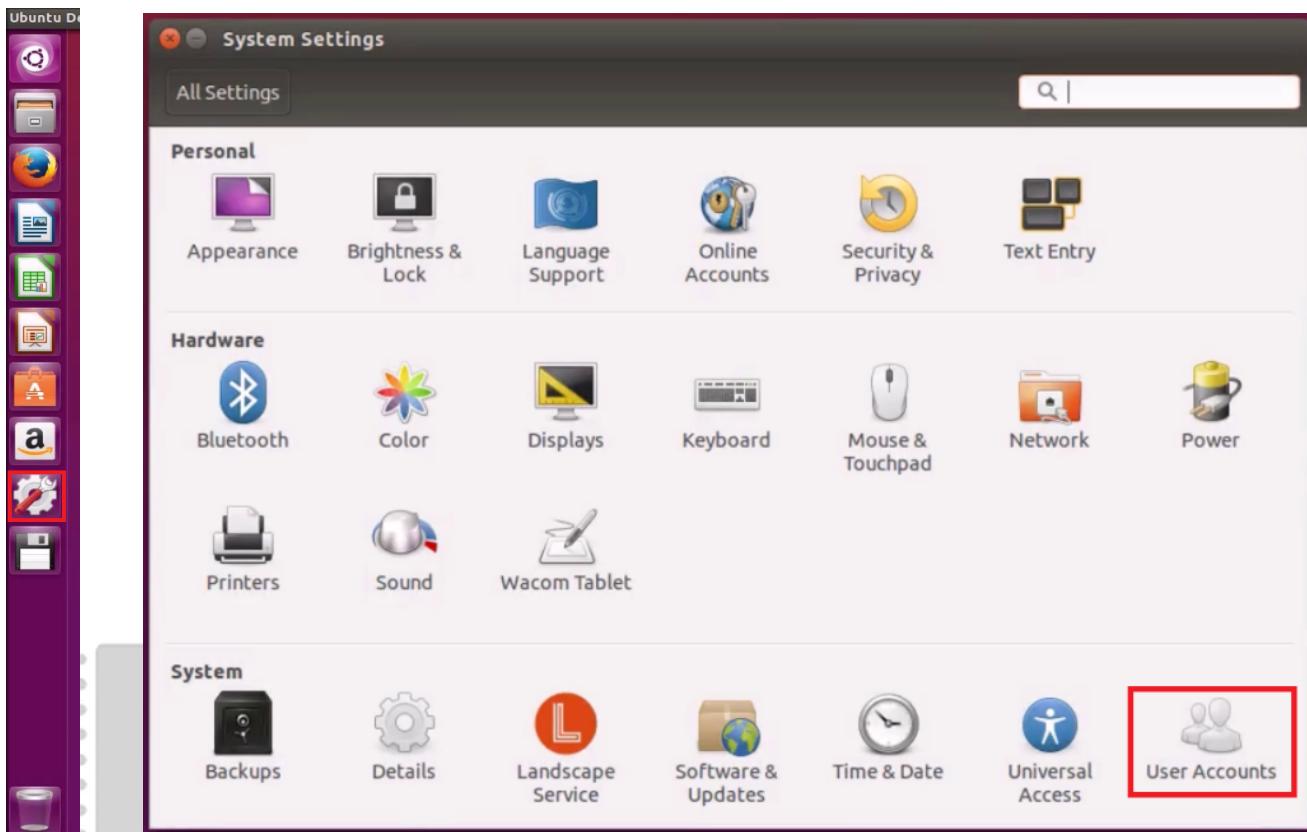


You can also select maximize to occupy the full monitor:

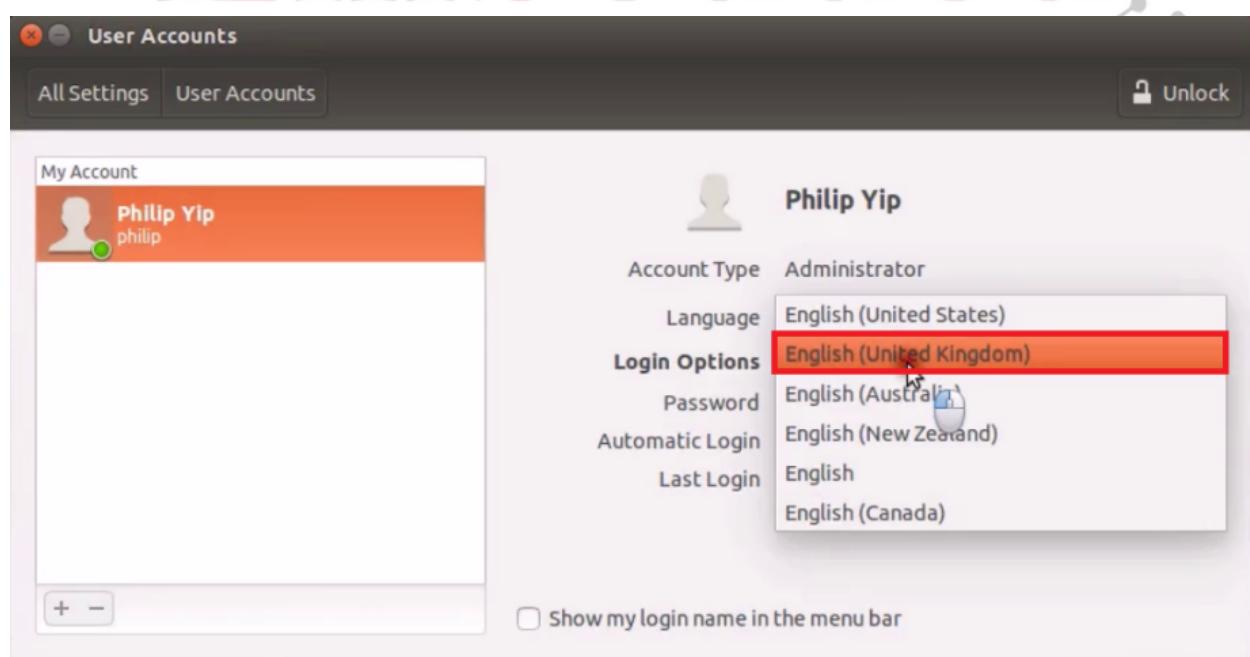


## Changing Language and Keyboard

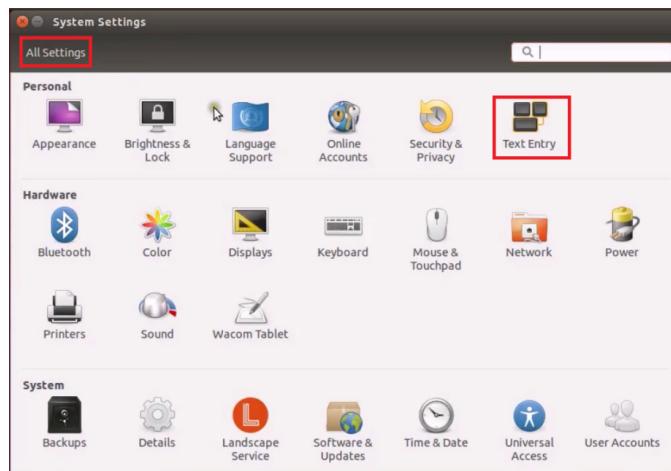
Select the settings crank to the left sidebar and Select User account:



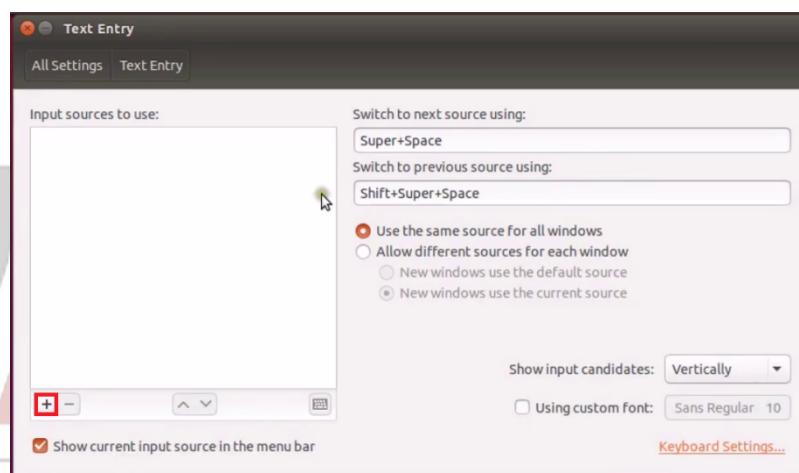
Change language from English (United States) to your desired Language:



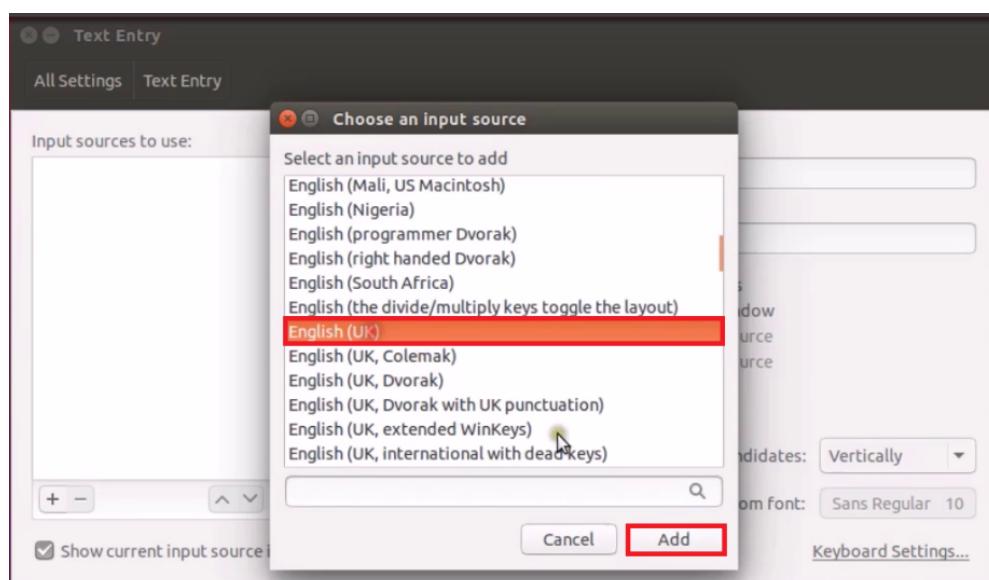
Select All settings and then go to Text Entry:



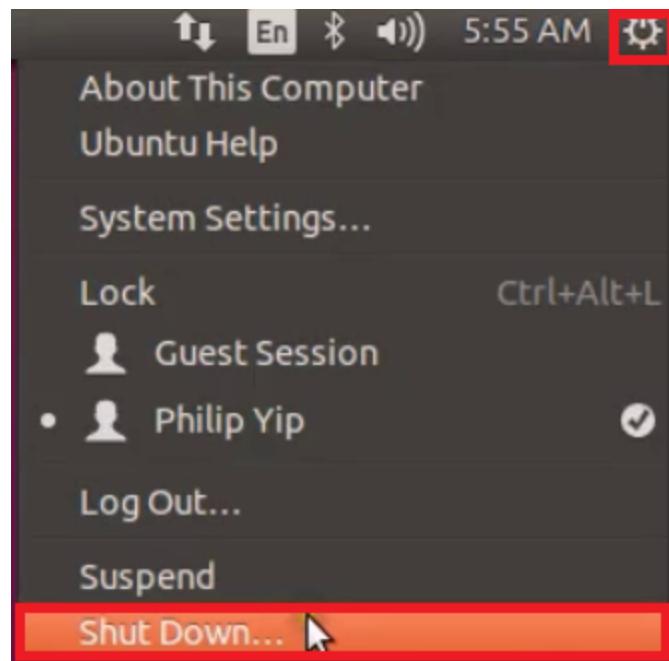
Select [+]



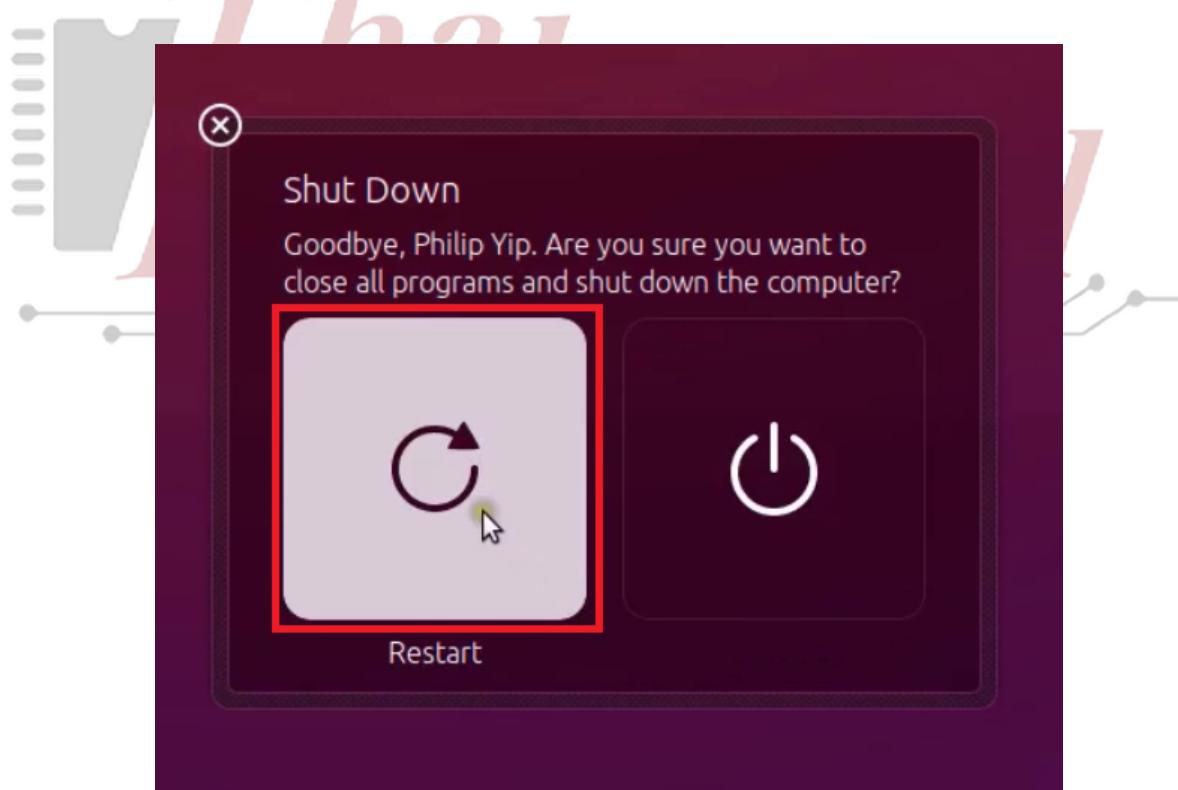
Select English (UK) and then select Add:



Select the power button and then shutdown:



Then restart:



You may now use your Ubuntu VM with your proper keyboard and language settings



## Installation of ROS Noetic on Ubuntu

We have 2 methods for install ROS-Noetic are by terminal and synaptic package manager

#Before Install, Let's install useful applications such as Visual Studio and File transfer. It will helpful for programming in Ubuntu

Install Visual Studio for amd64(Work Station)

```
sudo snap install --classic code
```

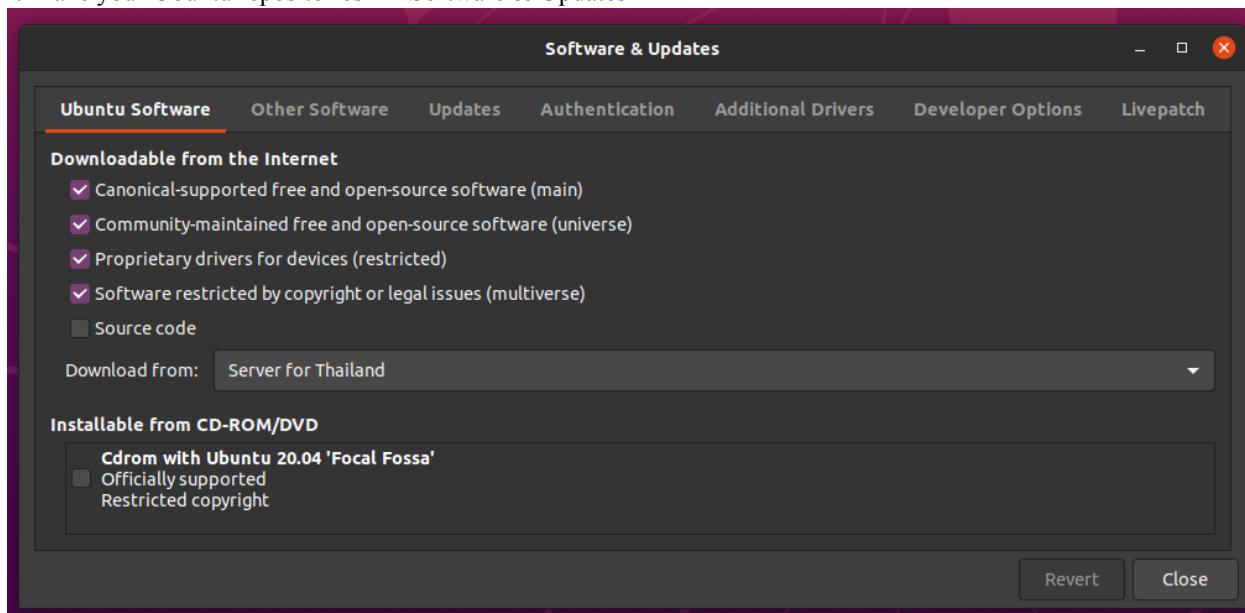
Install File transfers

```
sudo apt-get install openssh-server openssh-client
```

So, Let's get started!

### ROS Noetic Terminal method

1. Make your Ubuntu repositories in “Software & Updates”



2. Setup your sources.list

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

```
rengy@tesr-9939:~$ sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

3. Setup your keys

```
sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80'  
--recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```

```
rengy@tesr-9939:~$ sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80'  
--recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```



#### 4. Installation

Check software is up to date and upgrade  
 sudo apt update; sudo apt upgrade

```
rengy@tesr-9939:~$ sudo apt update; sudo apt upgrade
[sudo] password for rengy:
Hit:1 http://dl.google.com/linux/chrome/deb stable InRelease
Hit:2 http://th.archive.ubuntu.com/ubuntu focal InRelease
Hit:3 http://th.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:4 http://archive.canonical.com/ubuntu focal InRelease
Get:5 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Hit:6 https://librealsense.intel.com/Debian/apt-repo focal InRelease
Hit:7 http://packages.ros.org/ros-testing/ubuntu focal InRelease
Fetched 114 kB in 2s (74.4 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
All packages are up to date.
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.

Install ROS desktop-full
sudo apt install ros-noetic-desktop-full
```

```
rengy@tesr-9939:~$ sudo apt install ros-noetic-desktop-full
```

#### 5. Environment setup

Open .bashrc to setup environment by gedit or nano  
 gedit ~/.bashrc

And then, fill this data to last line of .bashrc  
 source /opt/ros/noetic/setup.bash

```
111
112
113
114
115
116
117
118
119
120
121
122
```

Source .bashrc  
 source ~/.bashrc

#### 6. Check ROS version

rosversion -d

```
rengy@tesr-9939:~$ rosversion -d
noetic
```

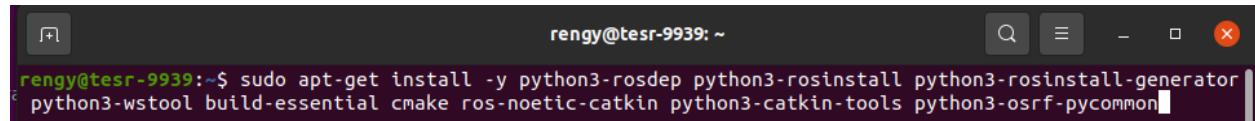
If your installation is complete and by right version “noetic” will show up



**[Optional]**

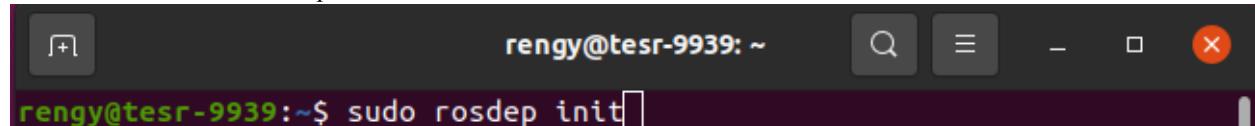
Install dependencies packages for building packages

```
sudo apt install python3-rosdep python3-rosinstall python3-rosinstall-generator python3-wstool  
build-essential cmake python3-catkin-tools python3-osrf-pycommon
```



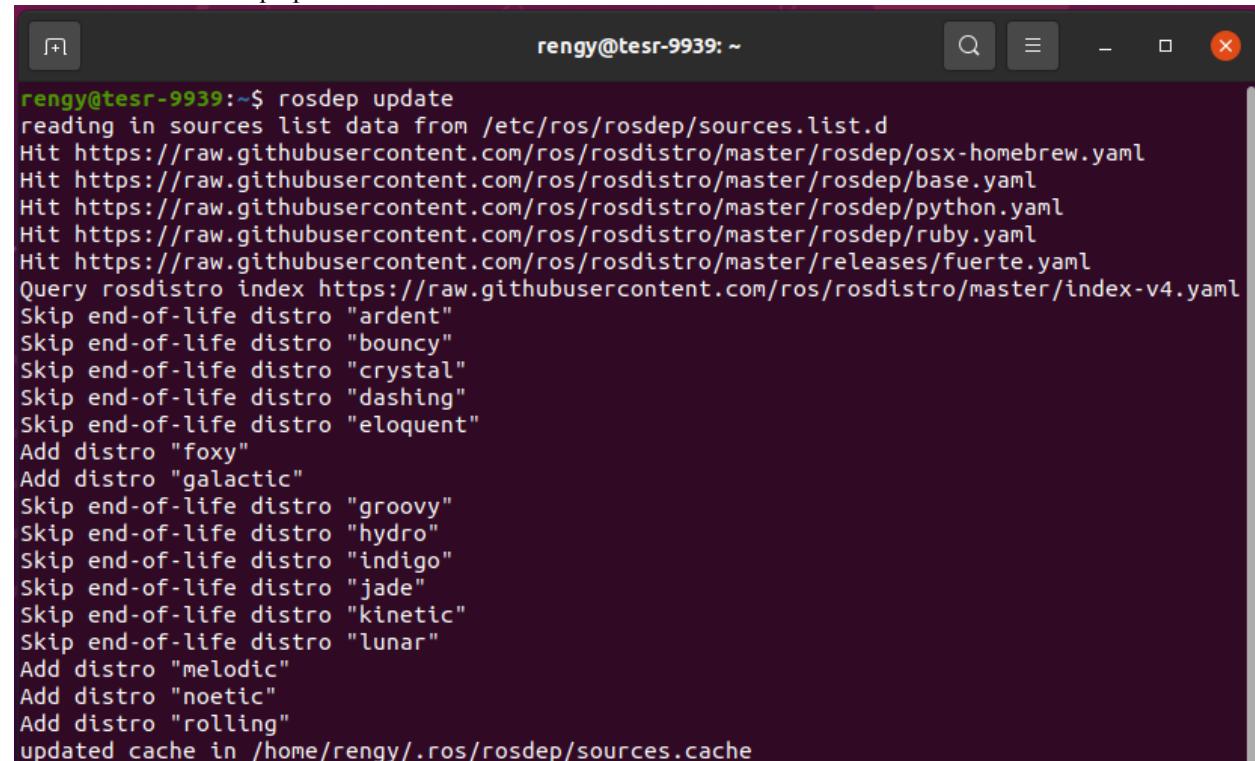
```
rengy@tesr-9939:~$ sudo apt-get install -y python3-rosdep python3-rosinstall python3-rosinstall-generator  
python3-wstool build-essential cmake ros-noetic-catkin python3-catkin-tools python3-osrf-pycommon
```

Initialize and update rosdep  
sudo rosdep init



```
rengy@tesr-9939:~$ sudo rosdep init
```

rosdep update

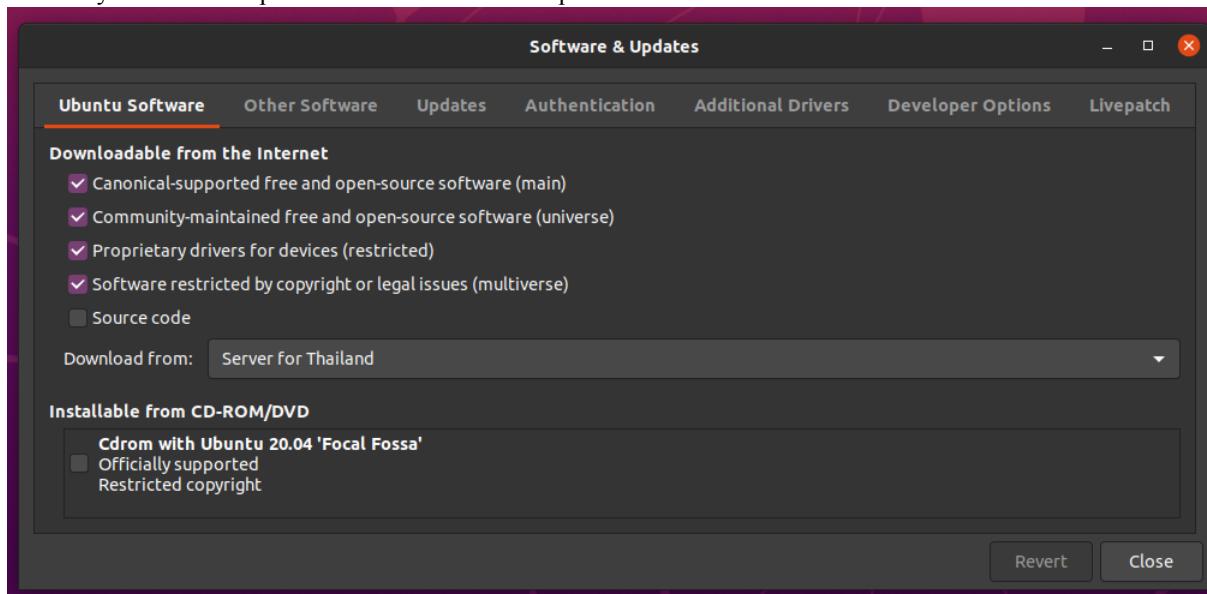


```
rengy@tesr-9939:~$ rosdep update  
reading in sources list data from /etc/ros/rosdep/sources.list.d  
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/osx-homebrew.yaml  
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/base.yaml  
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/python.yaml  
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/ruby.yaml  
Hit https://raw.githubusercontent.com/ros/rosdistro/master/releases/fuerte.yaml  
Query rosdistro index https://raw.githubusercontent.com/ros/rosdistro/master/index-v4.yaml  
Skip end-of-life distro "ardent"  
Skip end-of-life distro "bouncy"  
Skip end-of-life distro "crystal"  
Skip end-of-life distro "dashing"  
Skip end-of-life distro "eloquent"  
Add distro "foxy"  
Add distro "galactic"  
Skip end-of-life distro "groovy"  
Skip end-of-life distro "hydro"  
Skip end-of-life distro "indigo"  
Skip end-of-life distro "jade"  
Skip end-of-life distro "kinetic"  
Skip end-of-life distro "lunar"  
Add distro "melodic"  
Add distro "noetic"  
Add distro "rolling"  
updated cache in /home/rengy/.ros/rosdep/sources.cache
```



## ROS Noetic Synaptic package manager method

1. Make your Ubuntu repositories in “Software & Updates”



2. Setup Synaptic package manager

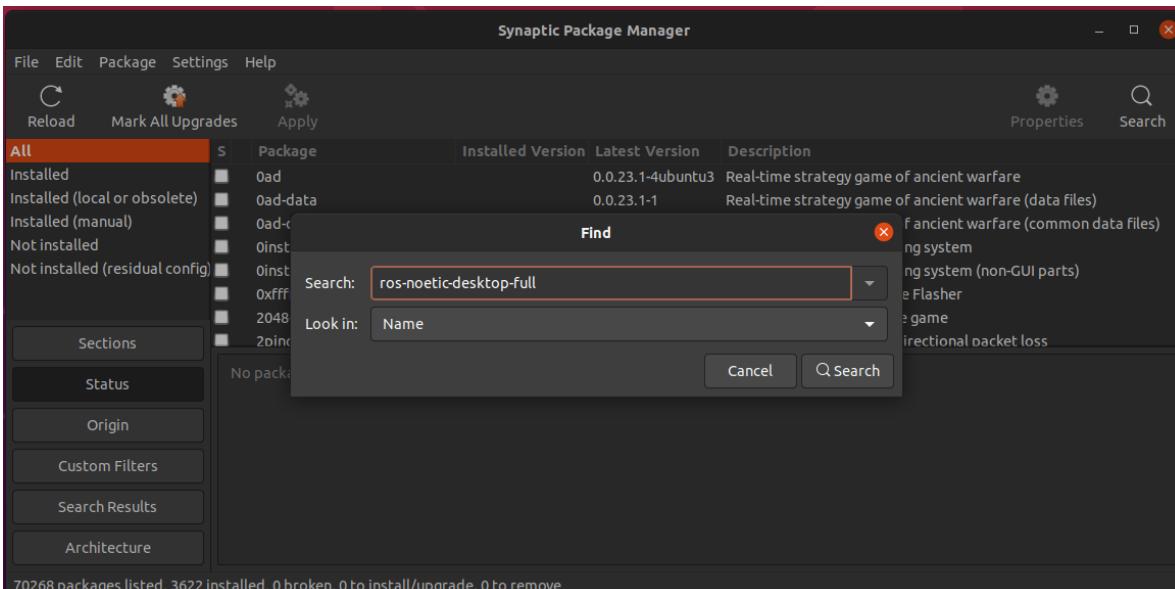
Install “Synaptic package manager”

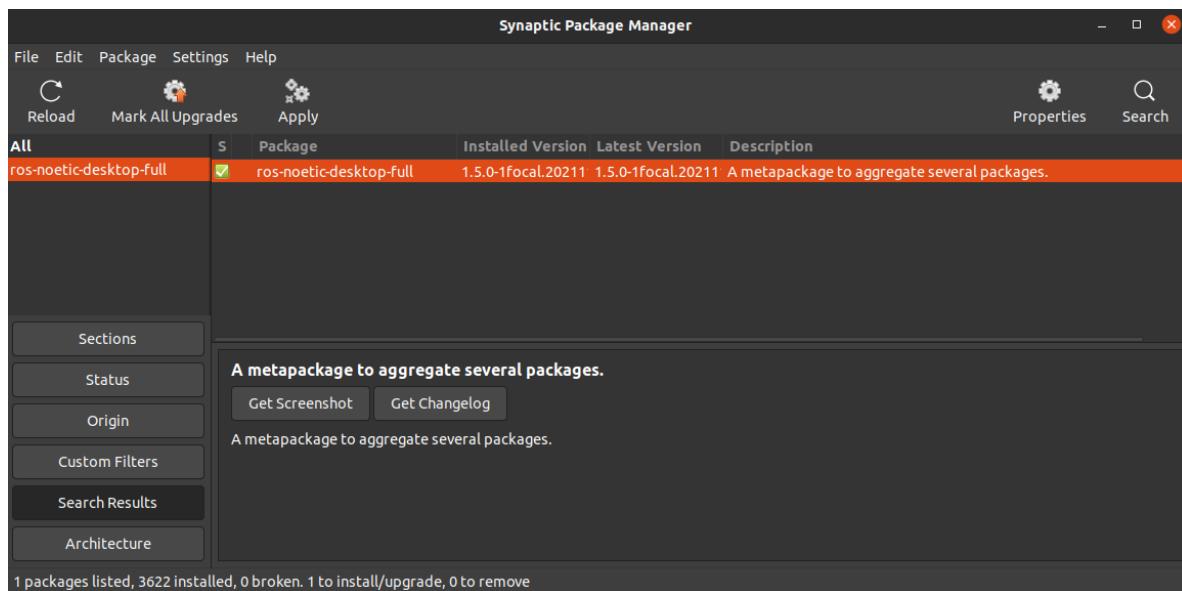
sudo apt synaptic

```
rengy@tesr-9939:~$ sudo apt install synaptic
Reading package lists... Done
Building dependency tree
Reading state information... Done
synaptic is already the newest version (0.84.6ubuntu5).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

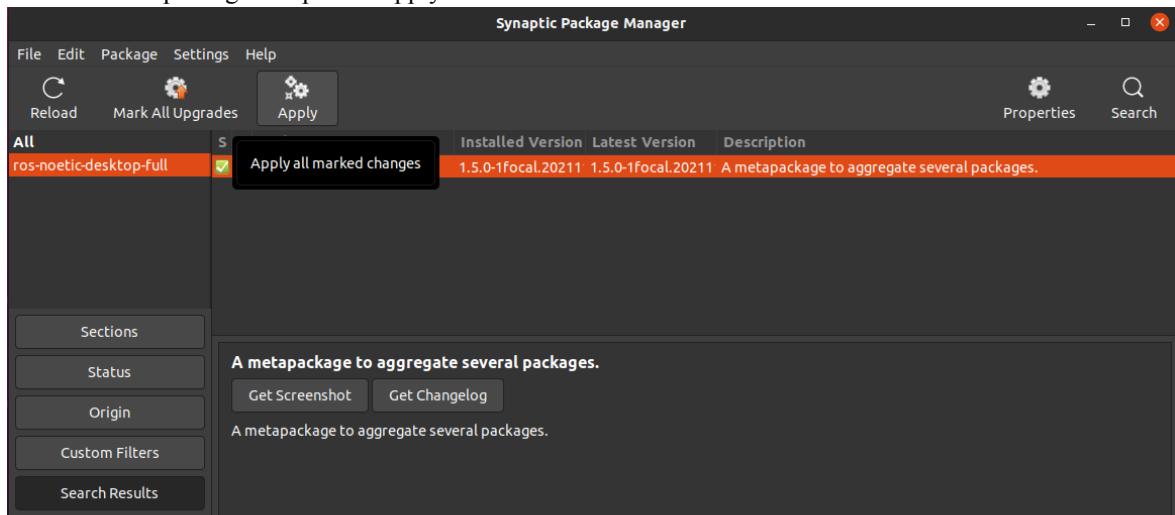
3. Installation

Open Synaptic packages manager and type “ros-noetic-desktop-full” to search box and mark for installation.





Mark package and press “Apply”



#### 4. Environment setup

Open .bashrc to setup environment by gedit or nano  
gedit ~/.bashrc

And then, fill this data to last line of .bashrc  
source /opt/ros/noetic/setup.bash

A terminal window titled "rengy@tesr-9939: ~" is shown. The left side shows the content of the .bashrc file, which includes various aliases and bash completion definitions. The right side shows the terminal prompt and the command "rengy@tesr-9939: ~\$ gedit ~/.bashrc" being entered. Below the terminal, a file browser window titled ".bashrc" is open, showing the same file content.

```

90 # some more ls aliases
91 alias ll='ls -alF'
92 alias la='ls -A'
93 alias l='ls -CF'
94
95 # Add an "alert" alias for long running commands. Use like so:
96 # sleep 10; alert
97 alias alert='notify-send --urgency=low -l "[$(echo $PPID)] $(echo $PNAME) | echo error"' "$@"
98
99 # Alias definitions.
100 # You may want to put all your additions into a separate file like
101 # ~/.bash_aliases, instead of adding them here directly.
102 # See /usr/share/doc/bash-doc/examples in the bash-doc package.
103
104 if [ -f ./_.bash_aliases ]; then
105 . ./_.bash_aliases
106 fi
107
108 # enable programmable completion features (you don't need to enable
109 # this, if it's already enabled in /etc/bash.bashrc and /etc/profile
110 # sources /etc/bash.bashrc)
111 if ! shopt -oq posix; then
112   if ! shopt -oq posix; then
113     . /usr/share/bash-completion/bash_completion
114   else
115     eval $( /etc/bash_completion )
116   fi
117 fi
118
119 source /opt/ros/noetic/setup.bash
120

```



Source .bashrc

source ~/.bashrc

```
rengy@tesr-9939:~$ source ~/._.bashrc
rengy@tesr-9939:~$
```

## 5. Check ROS version

If your installation is complete and by right version “noetic” will show up  
rosversion -d

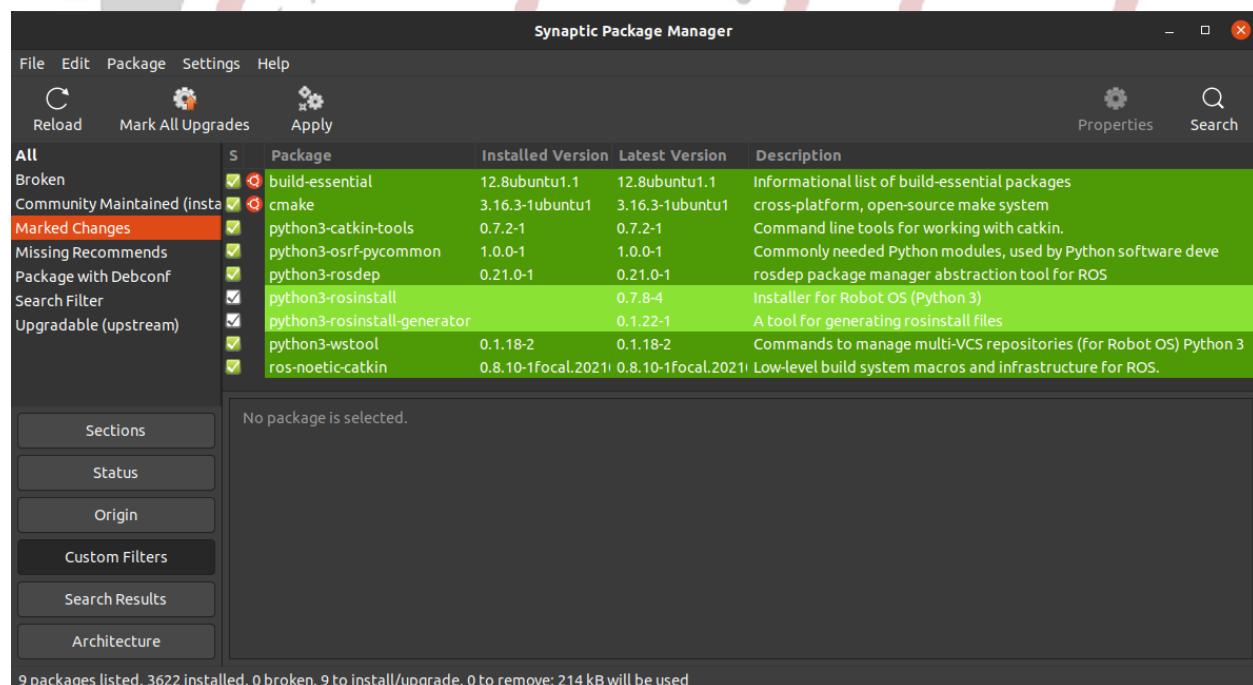
```
rengy@tesr-9939:~$ rosversion -d
noetic
```

[Optional]

Install dependencies packages for building packages. Type this list in search box and mark it to installation

- python3-rosdep
- python3-rosinstall
- python3-rosinstall-generator
- Python3-wstool
- Build-essential
- Cmake
- ros-noetic-catkin
- python3-catkin-tools
- python3-osrf-pycommon

\* You can see marked list at “Custom Filter > Marked Changes”



Initialize and update rosdep

Open terminal and initialize rosdep  
sudo rosdep init

```
rengy@tesr-9939:~$ sudo rosdep init
```

rosdep update

```
rengy@tesr-9939:~$ rosdep update
reading in sources list data from /etc/ros/rosdep/sources.list.d
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/osx-homebrew.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/base.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/python.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/ruby.yaml
Hit https://raw.githubusercontent.com/ros/rosdistro/master/releases/fuerte.yaml
Query rosdistro index https://raw.githubusercontent.com/ros/rosdistro/master/index-v4.yaml
Skip end-of-life distro "ardent"
Skip end-of-life distro "bouncy"
Skip end-of-life distro "crystal"
Skip end-of-life distro "dashing"
Skip end-of-life distro "eloquent"
Add distro "foxy"
Add distro "galactic"
Skip end-of-life distro "groovy"
Skip end-of-life distro "hydro"
Skip end-of-life distro "indigo"
Skip end-of-life distro "jade"
Skip end-of-life distro "kinetic"
Skip end-of-life distro "lunar"
Add distro "melodic"
Add distro "noetic"
Add distro "rolling"
updated cache in /home/rengy/.ros/rosdep/sources.cache
```

## Install Dependent ROS Packages

Run command line following below:

```
sudo apt-get install ros-noetic-joy ros-noetic-teleop-twist-joy \
ros-noetic-teleop-twist-keyboard ros-noetic-laser-proc \
ros-noetic-rgbd-launch ros-noetic-rosserial-arduino \
ros-noetic-rosserial-python ros-noetic-rosserial-client \
ros-noetic-rosserial-msgs ros-noetic-amcl ros-noetic-map-server \
ros-noetic-move-base ros-noetic-urdf ros-noetic-xacro \
ros-noetic-compressed-image-transport ros-noetic-rqt* ros-noetic-rviz \
ros-noetic-gmapping ros-noetic-navigation ros-noetic-interactive-markers
```

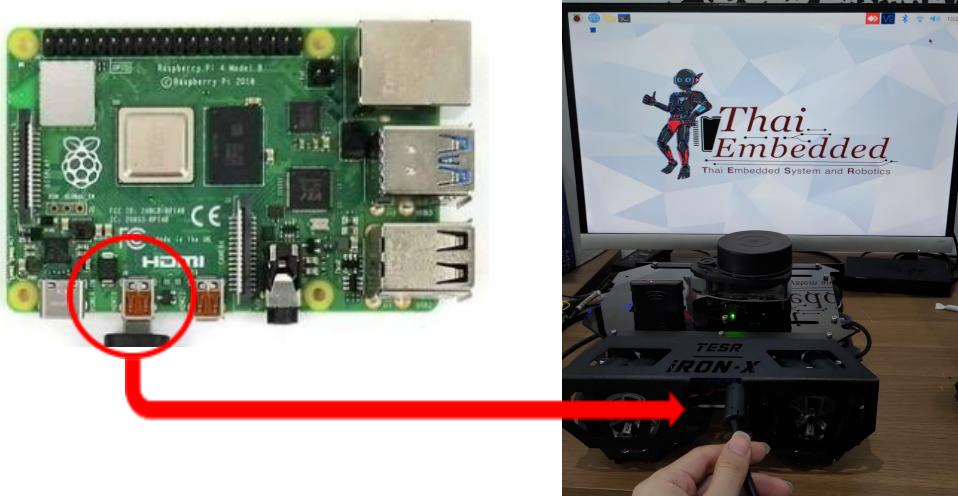
```
rengy@tesr-9939:~$ sudo apt-get install ros-noetic-joy ros-noetic-teleop-twist-joy \
> ros-noetic-teleop-twist-keyboard ros-noetic-laser-proc \
> ros-noetic-rgbd-launch ros-noetic-rosserial-arduino \
> ros-noetic-rosserial-python ros-noetic-rosserial-client \
> ros-noetic-rosserial-msgs ros-noetic-amcl ros-noetic-map-server \
> ros-noetic-move-base ros-noetic-urdf ros-noetic-xacro \
> ros-noetic-compressed-image-transport ros-noetic-rqt* ros-noetic-rviz \
> ros-noetic-gmapping ros-noetic-navigation ros-noetic-interactive-markers
[sudo] password for rengy: [REDACTED]
```



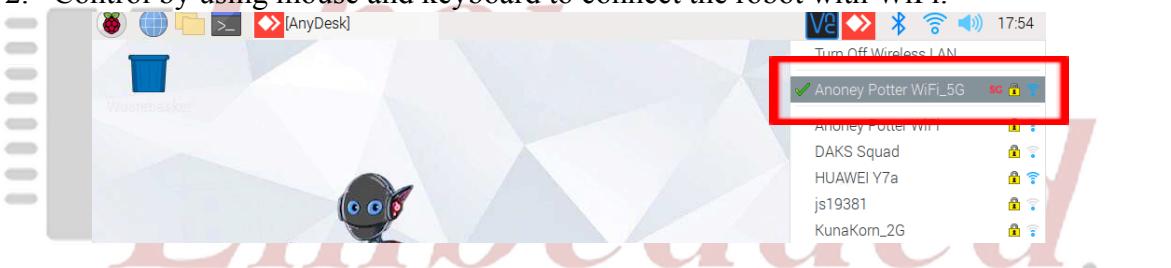
## iron-X Setup

### Step 1 Check the iron-X's IP(Desktop mode) for the first time setup.

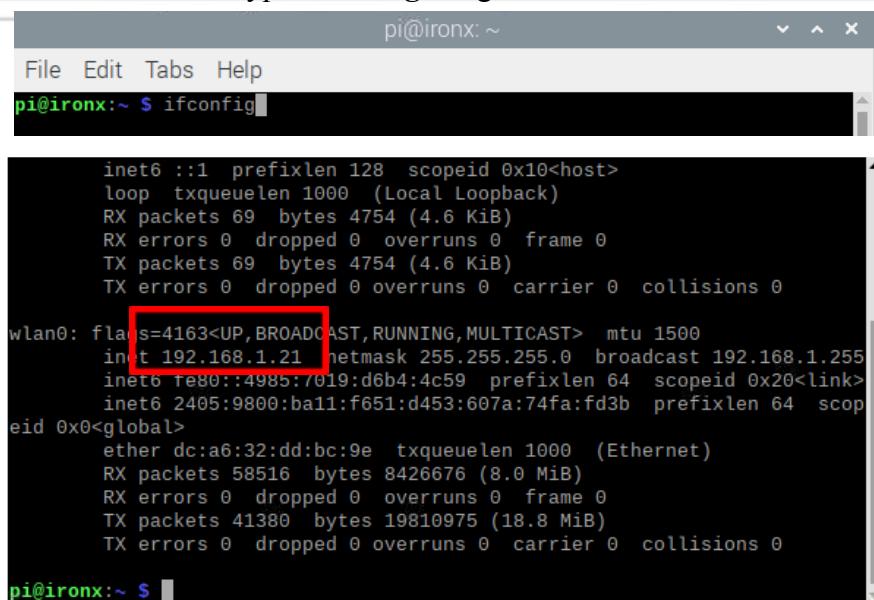
1. Connect your robot to the monitoring screen by Micro HDMI wire.



2. Control by using mouse and keyboard to connect the robot with WiFi.



3. Open terminal and then type “ifconfig” to get the iron-X's IP.



```
pi@ironx:~ $ ifconfig

    inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 69  bytes 4754 (4.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 69  bytes 4754 (4.6 KiB)
        TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
      inet 192.168.1.21  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::4985:7019:d6b4:4c59  prefixlen 64  scopeid 0x20<link>
        inet6 2405:9800:ba11:f651:d453:607a:74fa:fd3b  prefixlen 64  scopeid 0x0<global>
          ether dc:a6:32:dd:bc:9e  txqueuelen 1000  (Ethernet)
          RX packets 58516  bytes 8426676 (8.0 MiB)
          RX errors 0  dropped 0  overruns 0  frame 0
          TX packets 41380  bytes 19810975 (18.8 MiB)
          TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

pi@ironx:~ $
```

(In this case, iron-X's IP is 192.168.1.21)



## Step 2 Access iron-X's terminal

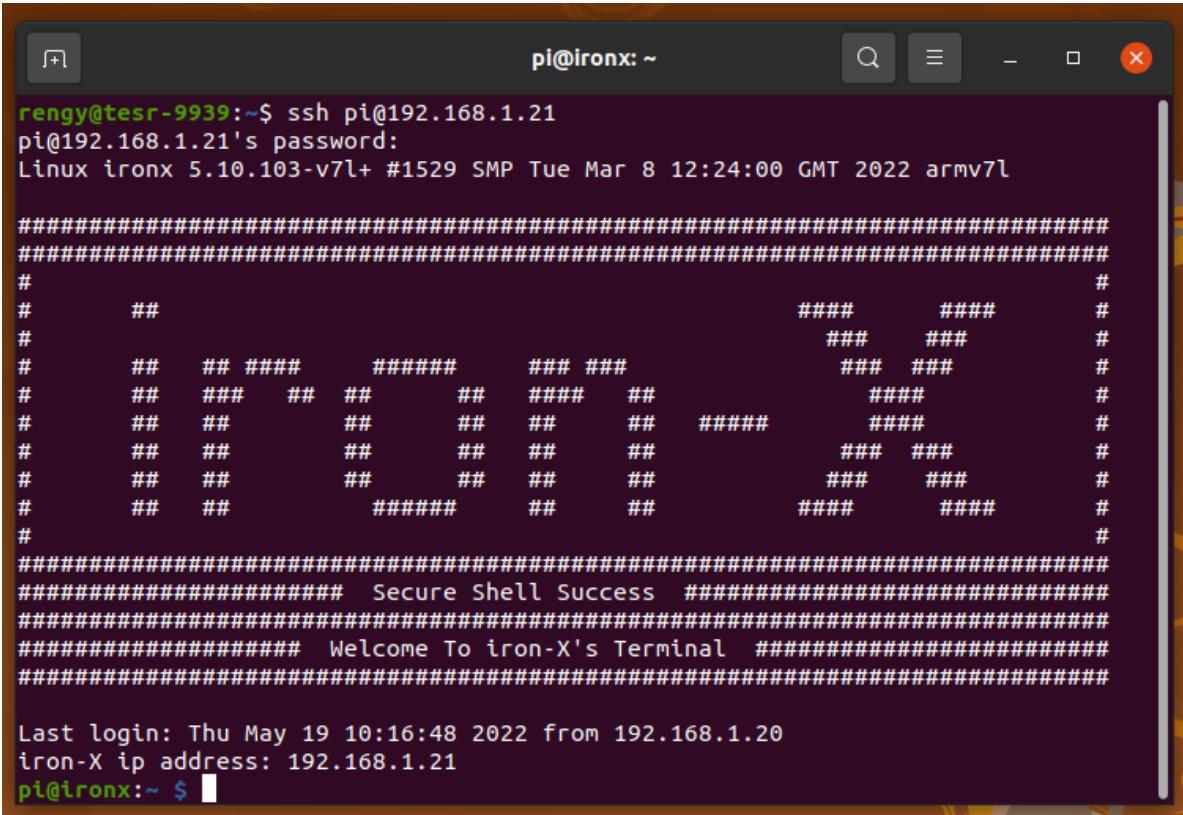
To Access iron-X's terminal we can do 2 methods following as:

- Terminal Mode
- Remote Desktop Mode

### (Terminal Mode)

**Secure shell to iron-X. \*Connect you PC and Robot to the same local network.**

1. Open Command Prompt or Terminal on your PC. Type "ssh pi@192.168.1.21".
2. After that, It will ask for password. So, type "ironxtesr".
3. Now, you will access to remote iron-X on your PC/Laptop.



The screenshot shows a terminal window titled "pi@ironx: ~". The session starts with the command "ssh pi@192.168.1.21" followed by the password "pi@192.168.1.21's password:". The system then displays its details: "Linux ironx 5.10.103-v7l+ #1529 SMP Tue Mar 8 12:24:00 GMT 2022 armv7l". A decorative banner follows, consisting of a grid of '#' characters. The banner includes the text "Secure Shell Success" and "Welcome To iron-X's Terminal". At the bottom, it shows the last login information: "Last login: Thu May 19 10:16:48 2022 from 192.168.1.20" and "iron-X ip address: 192.168.1.21", ending with the prompt "pi@ironx:~ \$".

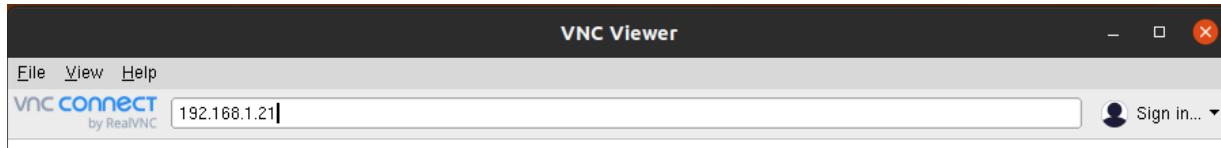


## (Remote Desktop Mode)

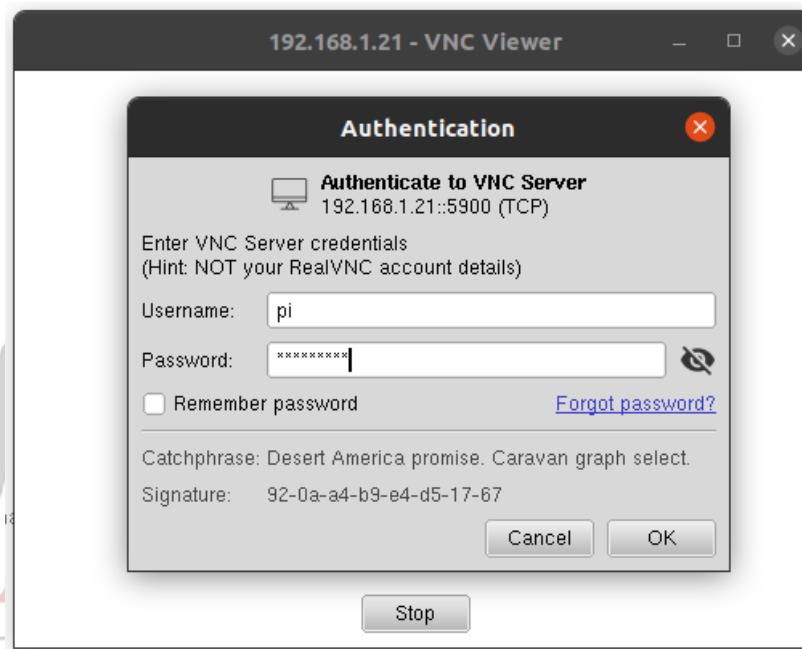
Access to iron-X's desktop by using VNC Viewer. (Local network)

\*Your PC/Laptop and iron-X must to connect on the same local network.

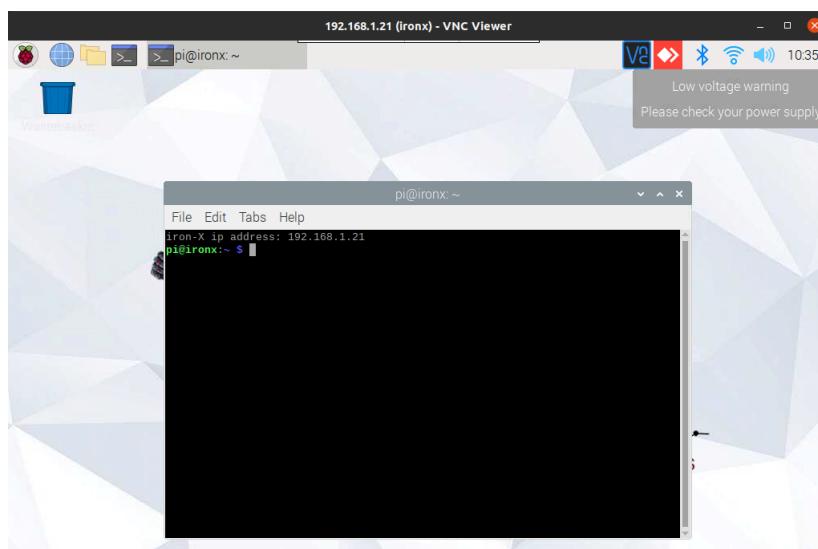
1. Open VNC Viewer and type "192.168.1.21"



2. Press Enter and fill Username "pi" and Password "ironxtesr"



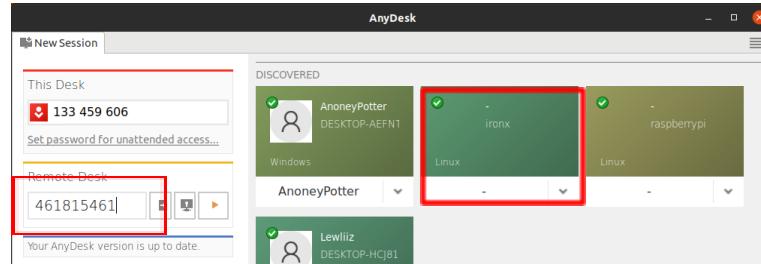
3. Open terminal by click at the Terminal's icon. So, we're finishes.



## (Remote Desktop Mode)

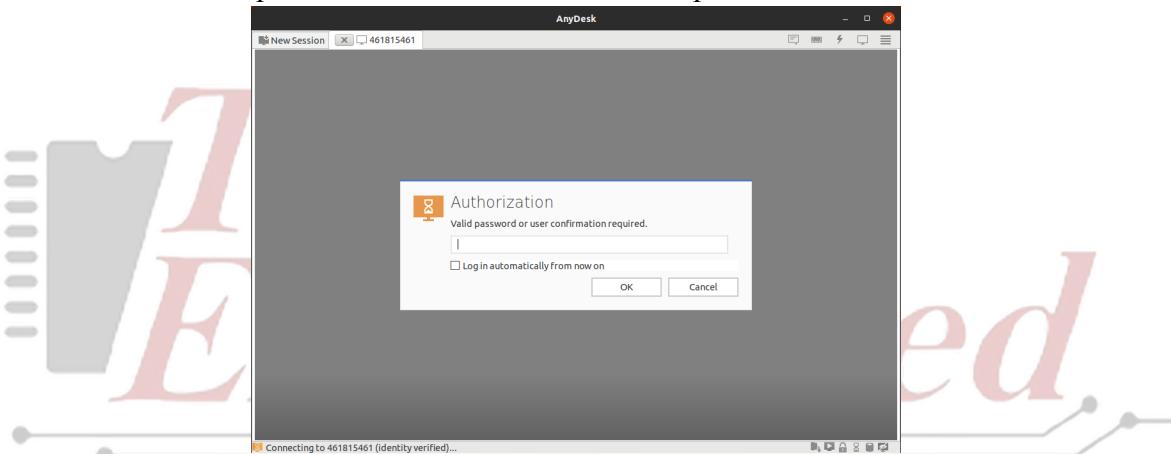
### Access to iron-X's desktop by using AnyDesk. (Internet)

- To use AnyDesk you must to know the Desk code or you can access from DISCOVERED Desk following as picture:

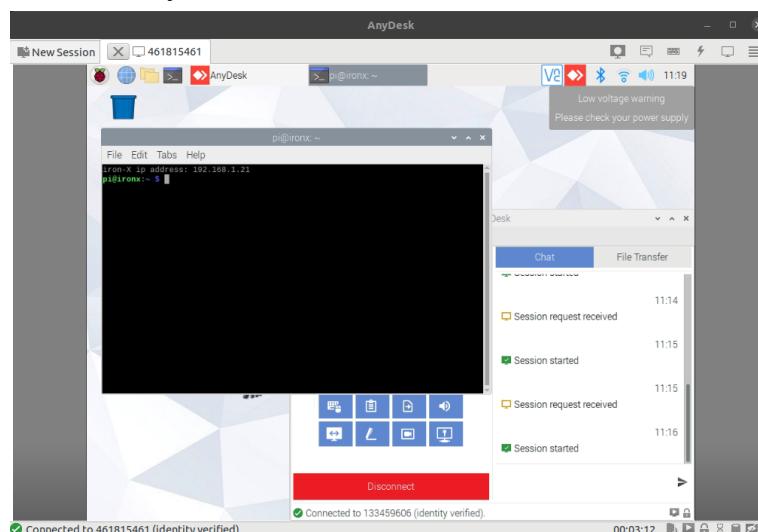


\*The desk from DISCOVERED may not pop-up. In that case, you must connect micro HDMI with iron-X's pi to get the Desk code from "This Desk" of iron-X's AnyDesk Screen.

- Fill the password "ironxtesr112296" And press OK..



- Open terminal by click at the Terminal's icon. So, we're finishes.



### Step 3 Setup ROS-IP's configuration in file .bashrc on PC/Laptop and iron-X.

On PC/Laptop we will check for IP and use it as Master's IP.

**(In this case, PC's IP is 192.168.1.20)**

```
rengy@tesr-9939:~$ ifconfig
enp2s0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
      ether 04:42:1a:88:d7:a4 txqueuelen 1000 (Ethernet)
      RX packets 0 bytes 0 (0.0 B)
      RX errors 0 dropped 0 overruns 0 frame 0
      TX packets 0 bytes 0 (0.0 B)
      TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
      inet 127.0.0.1 netmask 255.0.0.0
      inet6 ::1 prefixlen 128 scopeid 0x10<host>
        loop txqueuelen 1000 (Local Loopback)
        RX packets 5546 bytes 516037 (516.0 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 5546 bytes 516037 (516.0 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
      inet 192.168.1.20 netmask 255.255.255.0 broadcast 192.168.1.255
        inet6 fe80::2ad2:10ee:34d7:e693 prefixlen 64 scopeid 0x20<link>
          inet6 2405:9800:ba11:f651:b531:6539:c9db:39a0 prefixlen 64 scopeid 0x0<global>
          inet6 2405:9800:ba11:f651:7b29:a6a2:4215:8a55 prefixlen 64 scopeid 0x0<global>
            RX packets 16716 bytes 145616 (145.6 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 16716 bytes 145616 (145.6 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Setup Master's IP config to use PC/Laptop as Server. **(PC/Laptop's Terminal)**

Type "sudo nano ~/.bashrc"

```
rengy@tesr-9939:~$ sudo nano ~/.bashrc
```

Write in last line of the .bashrc following as:

```
"export ROS_MASTER_URI=http://192.168.1.20:11311"
"export ROS_HOSTNAME=192.168.1.20"
"export ROS_IP=192.168.1.20"
```

```
GNU nano 4.8           /home/rengy/.bashrc           Modified

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

source /opt/ros/noetic/setup.bash
source ~/catkin_ws/devel/setup.bash
source ~/ws_moveit/devel/setup.bash
source ~/cartographer_ws/devel_isolated/setup.bash
source ~/kobuki_ws/devel/setup.bash

#ROS-IP's configuration:
export ROS_MASTER_URI=http://192.168.1.20:11311
export ROS_HOSTNAME=192.168.1.20
export ROS_IP=192.168.1.20
```

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
 ^X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell ^ Go To Line

Save by Press "Ctrl+o" and then Enter to confirm and then "Ctrl+x" to exit from nano  
 After that, type "source .bashrc".

```
rengy@tesr-9939:~$ sudo nano ~/.bashrc
[sudo] password for rengy:
rengy@tesr-9939:~$ source .bashrc
rengy@tesr-9939:~$
```



**On iron-X's setup ROS-IP's configuration to Subscriber to Master.**

**\*Keep that in mind, using robot's IP refer with the check the iron-X's IP topic part.**

## Terminal mode

**(In this case, iron-X's IP is 192.168.1.21)**

Type “ssh pi@192.168.1.21”

Type “`sudo nano ~/.bashrc`”

Write in last line of the .bashrc following as:

```
"export ROS_MASTER_URI=http://192.168.1.20:11311"  
"export ROS_HOSTNAME=192.168.1.21"
```

```
pi@ironx: ~
GNU nano 3.2          /home/pi/.bashrc      Modified

source /opt/ros/noetic/setup.bash
source ~/cartographer_ws/devel_isolated/setup.bash
source ~/cartographer_ws/devel_isolated/cartographer_ros/setup.bash
source ~/cartographer_ws/devel_isolated/cartographer_ros_msgs/setup.bash
source ~/cartographer_ws/devel_isolated/cartographer_rviz/setup.bash

export ROS_PACKAGE_PATH=/usr/share:/home/pi/ros_catkin_ws/src:/opt/ros/noetic/share

echo "iron-X ip address: $(/bin/hostname -I | awk '{print $1}')"
echo "iron-X ip address: $(/bin/ip -o -4 addr list wlan0 | awk '{print $2}')"

export ROS_MASTER_URI=http://192.168.1.20:11311
export ROS_HOSTNAME=192.168.1.21
```

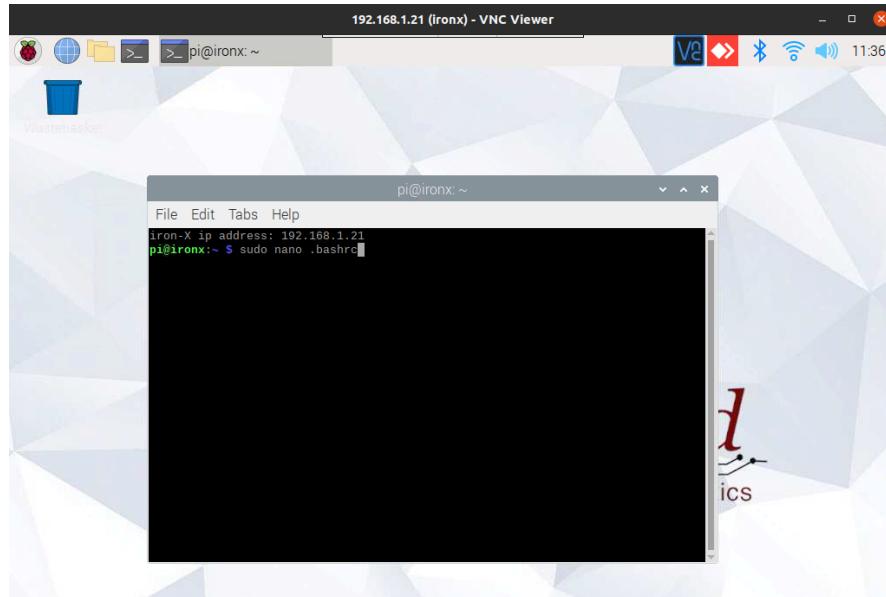
Save by Press "Ctrl+o" and then Enter to confirm and then "Ctrl+x" to exit from nano  
After that, type "source .bashrc".

```
pi@ironx:~ $ sudo nano ~/.bashrc  
pi@ironx:~ $ source .bashrc  
iron-X ip address: 192.168.1.21  
pi@ironx:~ $
```



## Remote Desktop mode

Type “sudo nano ~/.bashrc”



Write in last line of the .bashrc following as:

```
"export ROS_MASTER_URI=http://192.168.1.20:11311"
```

```
"export ROS_HOSTNAME=192.168.1.21"
```

**(In this case, iron-X's IP is 192.168.1.21)**

```
source /opt/ros/noetic/setup.bash
source ~/cartographer_ws/devel_isolated/setup.bash
source ~/cartographer_ws/devel_isolated/cartographer_ros/setup.bash
source ~/cartographer_ws/devel_isolated/cartographer_ros_msgs/setup.bash
source ~/cartographer_ws/devel_isolated/cartographer_rviz/setup.bash

export ROS_PACKAGE_PATH=/usr/share:/home/pi/ros_catkin_ws/src:/opt/ros/noetic/share

echo "iron-X ip address: `ifconfig -a | awk '/inet addr/{print $2}'`" > /etc/hosts

export ROS_MASTER_URI=http://192.168.1.20:11311
export ROS_HOSTNAME=192.168.1.21
```

Save by Press "Ctrl+o" and then Enter to confirm and then "Ctrl+x" to exit from nano  
After that, type "source .bashrc".

```
pi@ironx:~$ sudo nano ~/.bashrc
pi@ironx:~$ source .bashrc
iron-X ip address: 192.168.1.21
pi@ironx:~$
```



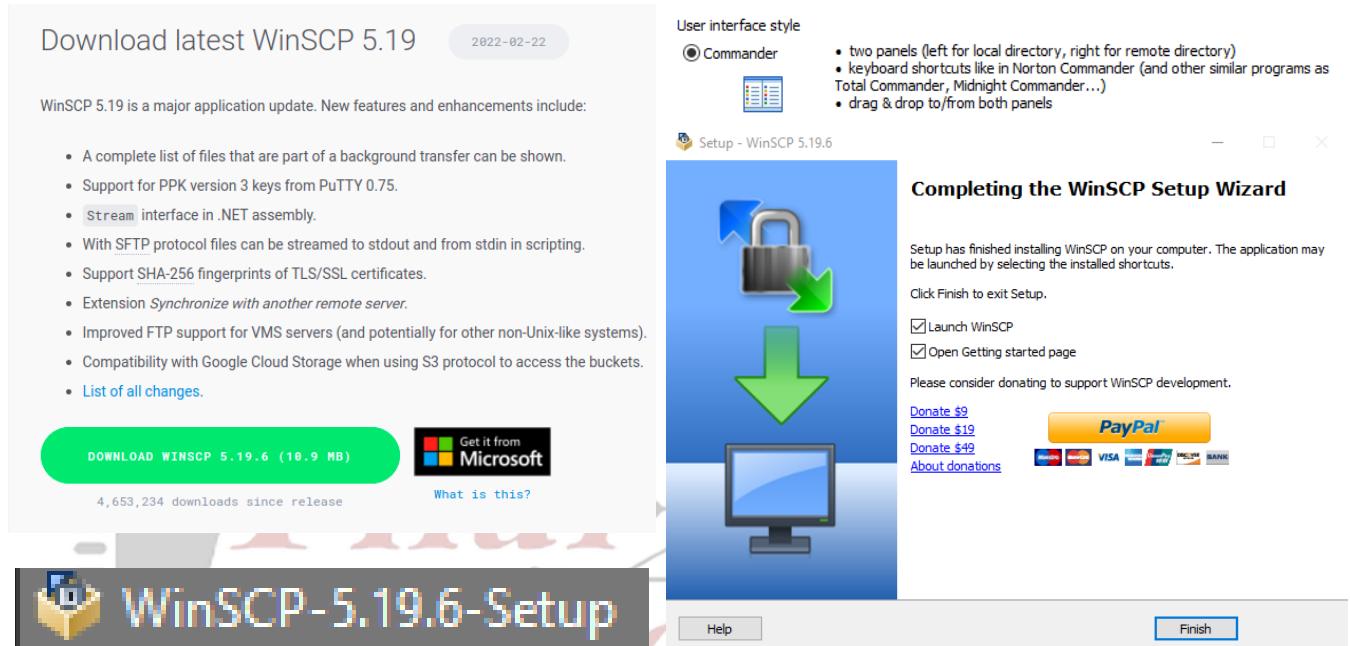
## Basic Files Transfer to iron-X.

### WinSCP [Recommended]

\*WinSCP is available for Window OS only if you using Linux OS you must use Anydesk or VNC Viewer method instead.

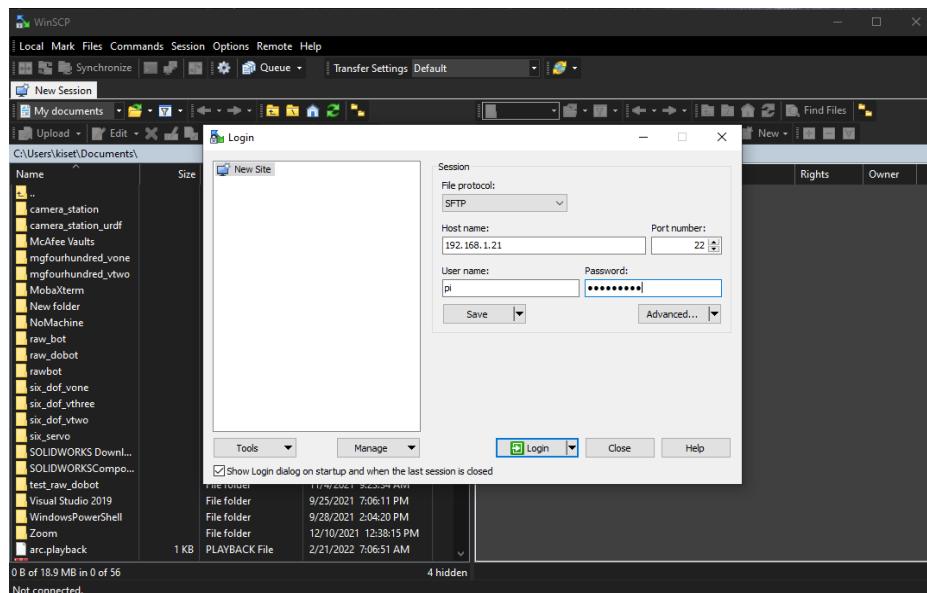
#### Download and Install WinSCP

<https://winscp.net/eng/downloads.php>

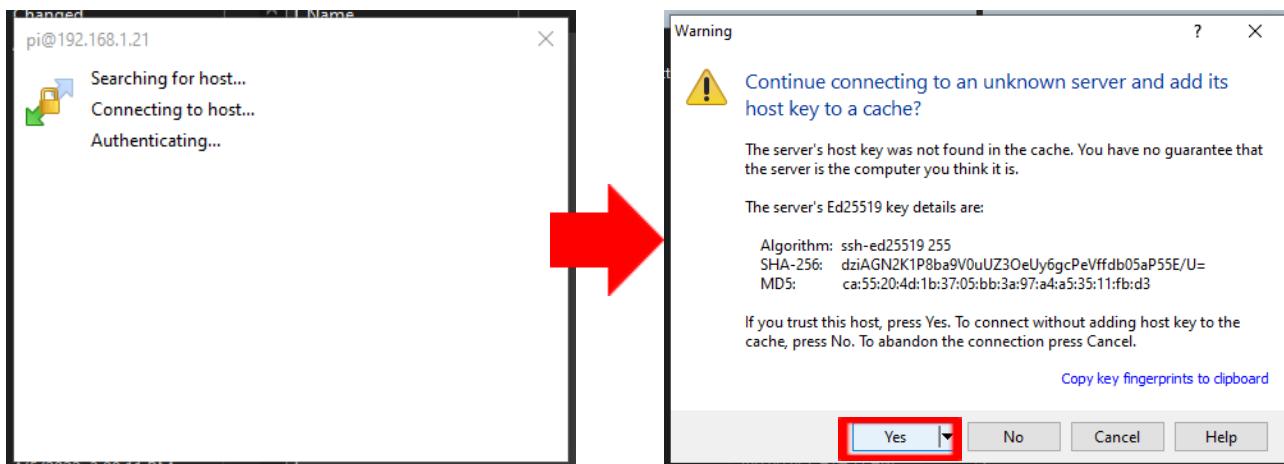


Open WinSCP and fill IP, Username and Password

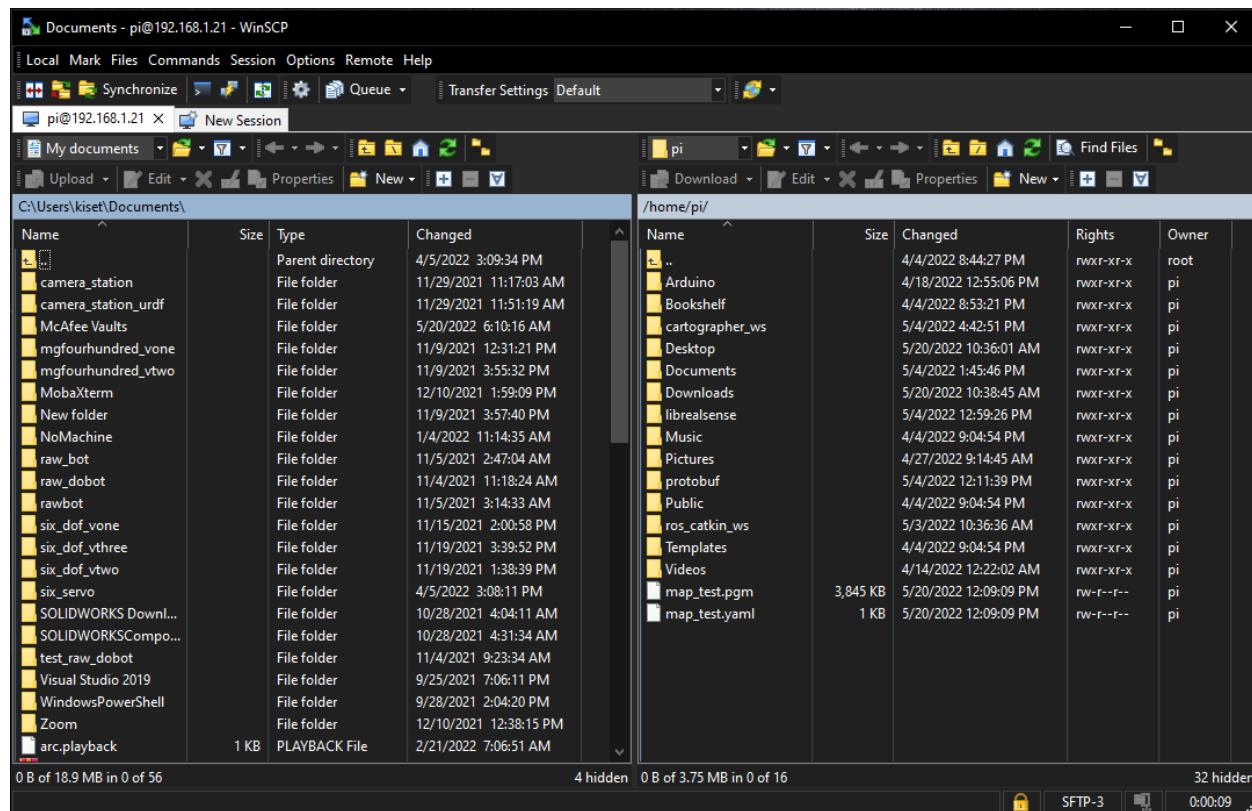
(In this case, IP = 192.168.1.21, Username: pi , Password: ironxtesr)



Wait for Authentication and Press “Yes” to confirm to add host key to a cache.



Now, you can transfer file from PC/Laptop to iron-X by Drag and Drop  
 (Left panel's PC/Laptop and Right panel's iron-X)

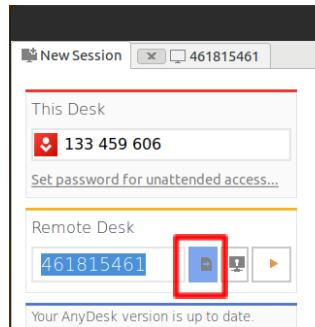


## (Desktop mode)

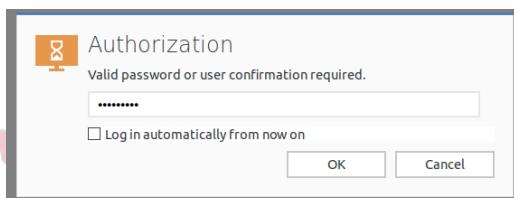
Remote iron-X through Desktop mode by Anydesk or VNC Viewer

### Transfer file using Anydesk

Open Anydesk and Select iron-X's desk and click at file transfer icon.

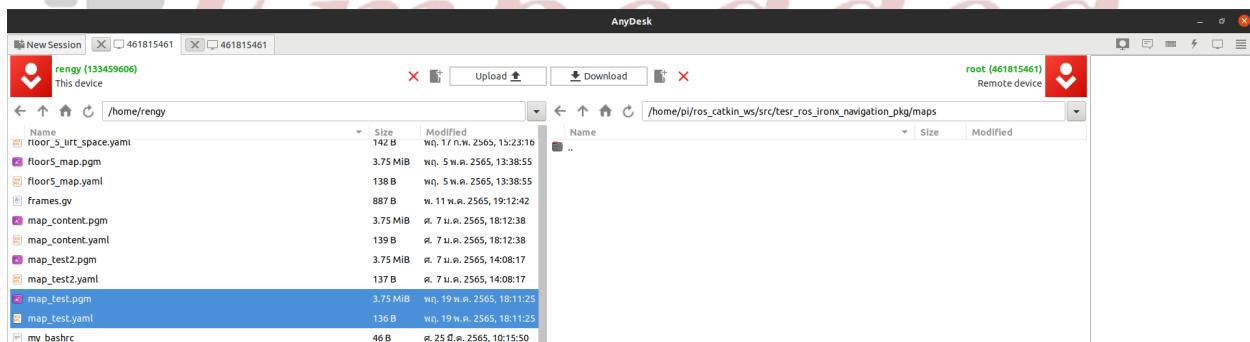


Fill the valid Password “ironxtesr112296” and Click “OK”

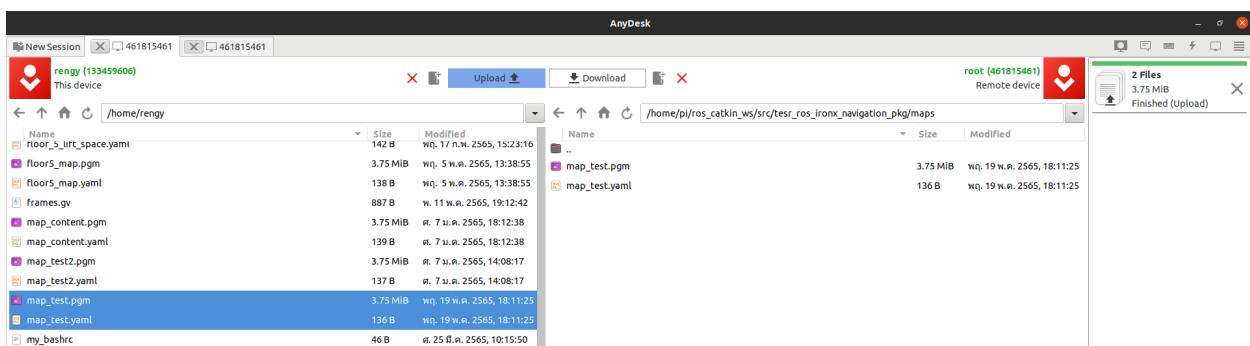


AnyDesk's File Transfer panel will show up. So, you can select file (left panel) and select File path (right panel) Then, click “Upload”.

(In this case, select map\_test.pgm and map\_test.yaml)

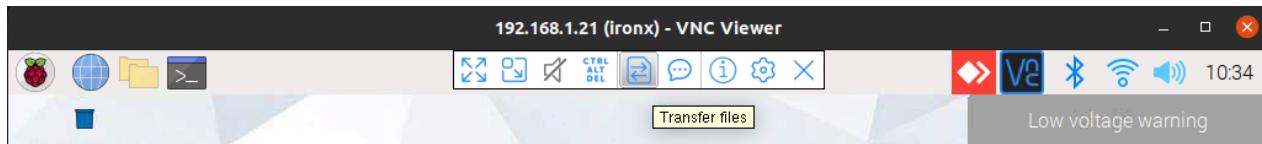


After “Finished(Upload)” pop-up that mean file transfer is completed.

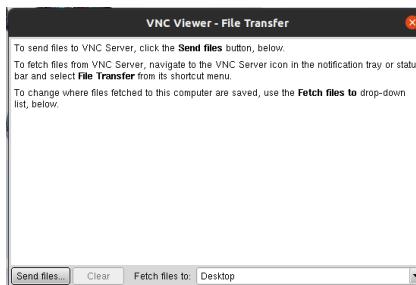


## Transfer file using VNC Viewer

Open and Connect VNC Viewer to iron-X. And then, move cursor around the top-middle of VNC Viewer. The Tools bar is show up and then click at “Transfer Files” icon.

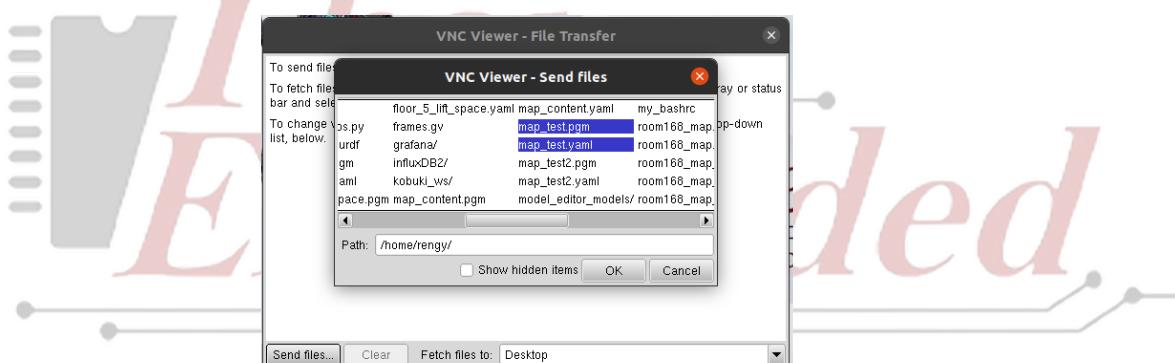


VNC Viewer - File Transfer will show up and then click “Send files...”

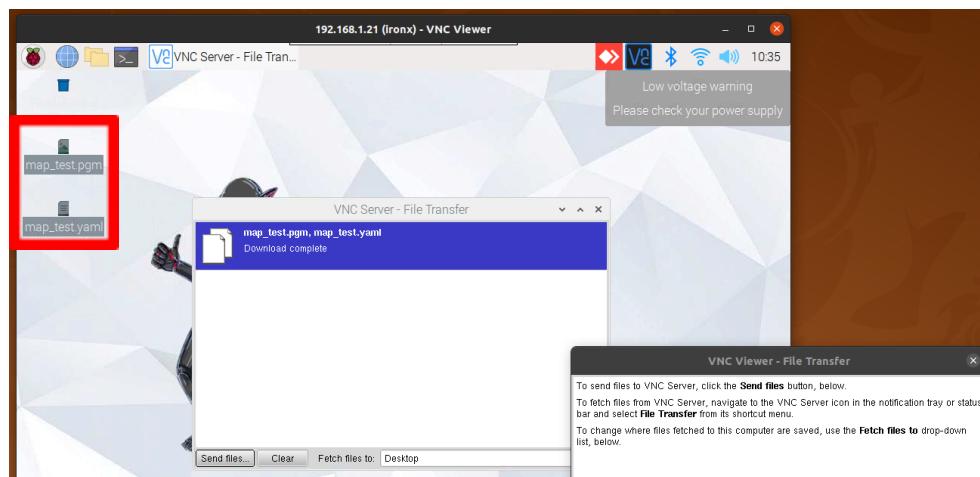


Select file you're prefer to transfer and click “OK”

(In this case, select map\_test.pgm and map\_test.yaml)



After that, VNC Server will pop-up and Show “Download Complete”  
So, both of the map\_file will show on iron-X’s Desktop.



# How to launch iron-X ROS package

## Package lists (In iron-X's workspace “`ros_catkin_ws`”)

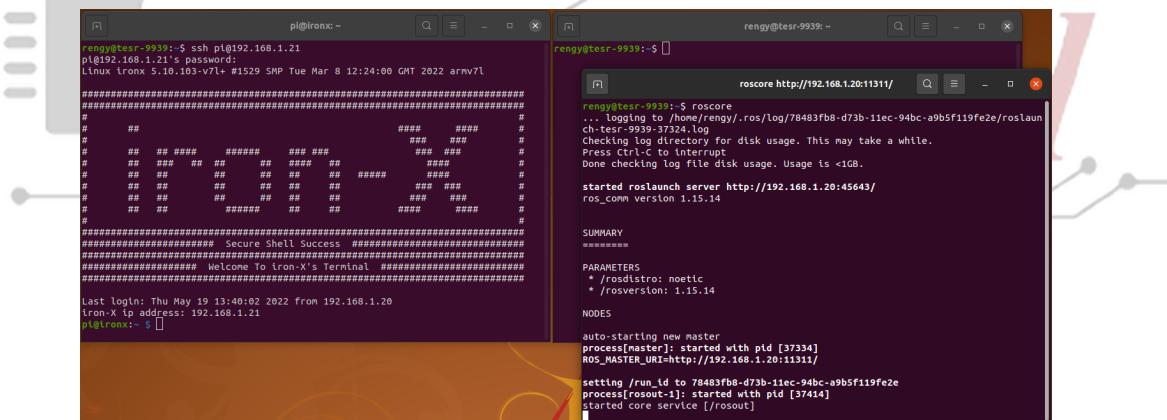
- tesr\_ros\_ironx\_driver\_pkg
  - tesr\_ros\_ironx\_navigation\_pkg

## Launch file lists

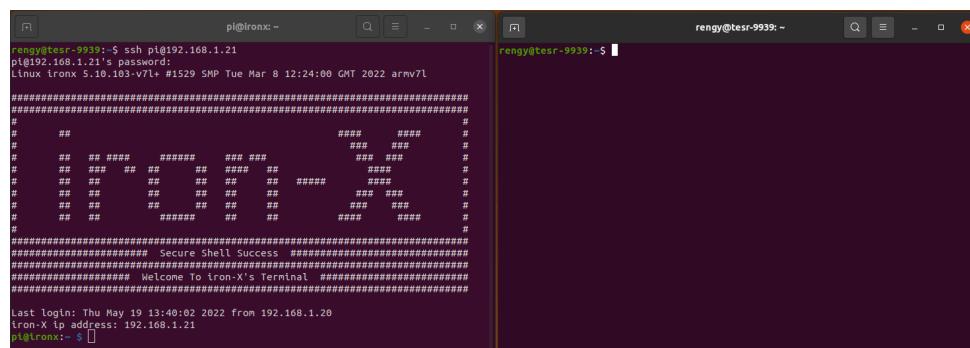
- **tesr\_ros\_ironx\_driver\_pkg**
    - ironx\_driver.launch
    - ironx\_bringup.launch
  - **tesr\_ros\_ironx\_navigation\_pkg (exclude the non-application launch file)**
    - ironx\_slam.launch
    - ironx\_mapping\_gmapping.launch
    - ironx\_mapping\_hector.launch
    - ironx\_mapping\_cartographer.launch
    - ironx\_nav.launch

## Start to Run iron-X applications (Terminal Mode)

First thing first, type “roscore” to open the ROS based system on PC/Laptop terminal. After that, you may minimize the terminal that run “roscore” and open the new one by press “Ctrl+Alt+t”.



We will operate on both of Remote Terminal (**Left panel**) and PC Terminal(**Right panel**)



## Remote Terminal

## PC/Laptop Terminal



## iron-X's driver

Launch ironx\_driver.launch (On Remote terminal)

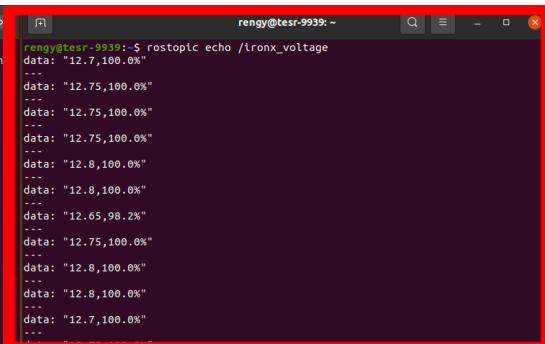
Type “roslaunch tesr\_ros\_ironx\_driver\_pkg ironx\_driver.launch”.

```
pi@ironx:~ $ rosrun tesr_ros_ironx_driver_pkg ironx_driver.launch
```

Check iron-X is working properly by check Voltage, odom\_raw and imu.

### Battery Data

(On PC/Laptop terminal) Check Voltage data by type “rostopic echo ironx\_voltage”.



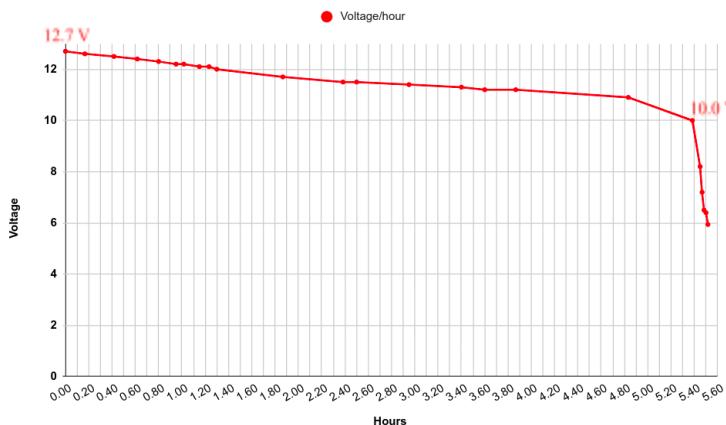
```
/home/pi/ros_catkin_ws/src/tesr_ros_ironx_driver_pkg/launc... pi@ironx: ~ $ rosrun tesr_ros_ironx_driver_pkg ironx_driver.launch ... logging to /home/pi/.ros/log/aa80b6fe-d7f6-11ec-b5f7-bd0d7991e7fb/roslaunch_ironx-6374.log Checking log directory for disk usage. This may take a while. Press Ctrl-C to interrupt Done checking log file disk usage. Usage is <1GB. started rosrun server http://192.168.1.21:37491/ SUMMARY ===== PARAMETERS * /rosdistro: noetic * /rosversion: 1.15.14 NODES / ironx_driver (tesr_ros_ironx_driver_pkg/ironx_driver.py) ROS_MASTER_URI=http://192.168.1.20:11311 process[ironx_driver-1]: started with pid [6421]
```

Battery's Data: "Voltage, Percentage"



- Battery is maxed at 12.7~13.0 V [100%] and should recharge when less than 10.5V.
- Battery can operate around 3 - 5 hours and need to recharge for 2 hours

The battery consumption is following as graph below:



\*Battery consumption rate is up to iron-X usage.

\*\*Please be aware to not let the Battery consumed to be less than 10.0V for good of battery's life time. And recharge while open the iron-X is not recommend too.



## Odometry\_raw Data

**(On PC/Laptop terminal)** Check Imu data by type “`rostopic echo /odom_raw`”.

## Remote Terminal

## PC/Laptop Terminal

## Imu Data

**(On PC/Laptop terminal)** Check Imu data by type “`rostopic echo /imu`”.

```
Last login: Thu May 19 13:40:02 2022 from 192.168.1.20
iron-X ip address: 192.168.1.21
pi@tronix: ~$ roslaunch tesr_ros_irnx_driver_pkg irnx_driver.launch
... logging to /home/pi/.ros/log/78483fb8-d73b-11ec-94bc-a9b5f119feze/roslaunch-
irnx-19263.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started rosrun server http://192.168.1.21:33795/
SUMMARY
=====
PARAMETERS
  * /rostdistro: noetic
  * /rosversion: 1.15.14
NODES
  /
    irnx_driver (tesr_ros_irnx_driver_pkg/irnx_driver.py)

ROS_MASTER_URI=http://192.168.1.20:11311

process[irnx_driver-1]: started with pid [19317]
[  ]
```

```
reny@tronix-9939: ~
[  ]***header:
  seq: 238
  stamp:
    secs: 1652943917
    nsecs: 120107412
    frame_id: "base_imu_link"
orientation:
  x: 0.0
  y: 0.0
  z: 0.0
  w: 1.0
orientation_covariance: [1000000.0, 0.0, 0.0, 0.0, 1000000.0, 0.0, 0.0, 0.0, 0.0,
5]angular_velocity:
  x: 0.0
  y: 0.0
  z: 0.0
angular_velocity_covariance: [1000000.0, 0.0, 0.0, 0.0, 1000000.0, 0.0, 0.0, 0.0,
, 1000000.0]
linear_acceleration:
  x: 0.5886
  y: -0.1962
  z: 10.300500000000001
linear_acceleration_covariance: [0.01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
```

## Remote Terminal

PC/Laptop Terminal



## Keyboard & Joy

### Keyboard's control (On PC/Laptop terminal)

After launch ironx\_driver, type to PC/Laptop terminal following as:

“rosrun teleop\_twist\_keyboard teleop\_twist\_keyboard.py”

```
rengy@tesr-9939: ~
rengy@tesr-9939: ~
rengy@tesr-9939: ~$ rosrun teleop_twist_keyboard teleop_twist_keyboard.py
```

\*ros-noetic-teleop-twist-keyboard is included in **dependent ROS packages**. If you can't use it properly. You may re-install ros-noetic-teleop-twist-keyboard and try again.

```
/home/pi/ros_catkin_ws/src/tesr_ros_ironx_driver_pkg/laun...
Last login: Thu May 19 11:45:57 2022 from 192.168.1.20
iron-X ip address: 192.168.1.21
pi@ironx: ~ rosrun tesr_ros_ironx_driver_pkg ironx_driver.launch
... Logging to /home/pi/.ros/log/78483fb8-d73b-11ec-94bc-a9b5f119fe2e/roslaunch-ironx-17251.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

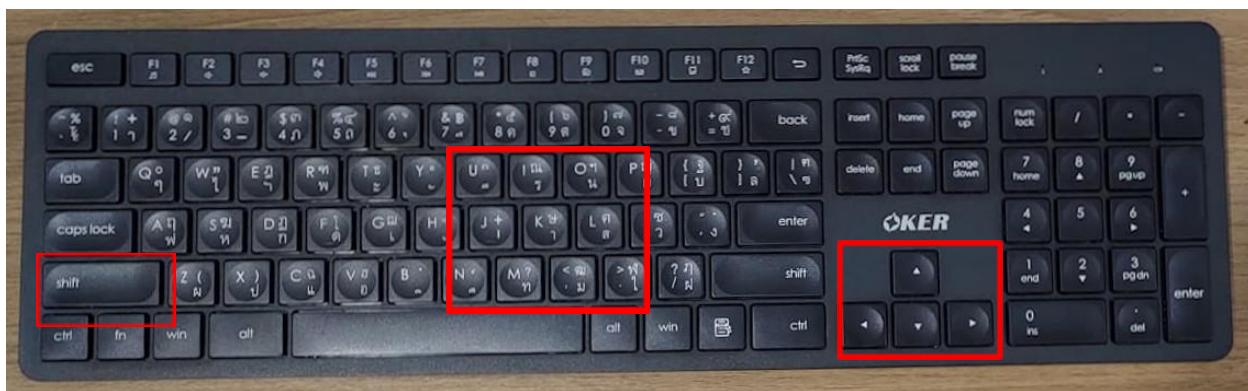
started roslaunch server http://192.168.1.21:42805/
SUMMARY
=====
PARAMETERS
  * /rosdistro: noetic
  * /rosversion: 1.15.14
NODES
/
  ironx_driver (tesr_ros_ironx_driver_pkg/ironx_driver.py)
ROS_MASTER_URI=http://192.168.1.20:11311
process[ironx_driver-1]: started with pid [17332]
```

```
rengy@tesr-9939: ~
rengy@tesr-9939: ~
rengy@tesr-9939: ~$ rosrun teleop_twist_keyboard teleop_twist_keyboard.py
Reading From the Keyboard and Publishing to Twist!
-----
Moving around:
  u   i   o
  j   k   l
  m   ,   .
For Holonomic mode (strafing), hold down the shift key:
-----
  U   I   O
  J   K   L
  M   <   >
anything else : stop
q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
CTRL-C to quit
```

Remote Terminal

PC/Laptop Terminal

The Keyboard Control button will show the list as the right terminal following the explanation in the right panel.



## Joy's control

The Joystick module is connected directly to the STM32 microcontroller board. So, after launch the “ironx\_driver.launch” you can turn on the Joystick switch and wait for it pair to the module. If joystick is paired LED on module will not blinking. So, you’re ready to use the joystick.



Joystick module



Joystick's status isn't blinking

Joystick controlling by 2 Analog stick and L1 button following as:

1. Right Analog stick use to Rotation of iron-X itself as CW and CCW.  
*(CW: Clockwise and CCW: Counter Clockwise)*
2. Left Analog stick use to Move in 8-direction.



## iron-X's bringup

\*You **should** stop the ironx\_driver before run ironx Bringup. Because ironx Bringup is already set to run ironx\_driver.

ironx Bringup.launch is launch file that runs to bringup the other components required to use in iron-X system. The component that ironx Bringup.launch brings up is as follows:

- iron-X's driver
- iron-X's model
- RPLIDAR A1

### (On Remote Terminal)

Type “rosrun tesr\_ros\_ironx\_driver\_pkg ironx Bringup.launch”

```
pi@ironx:~ $ rosrun tesr_ros_ironx_driver_pkg ironx Bringup.launch
```

### (On PC/Laptop Terminal)

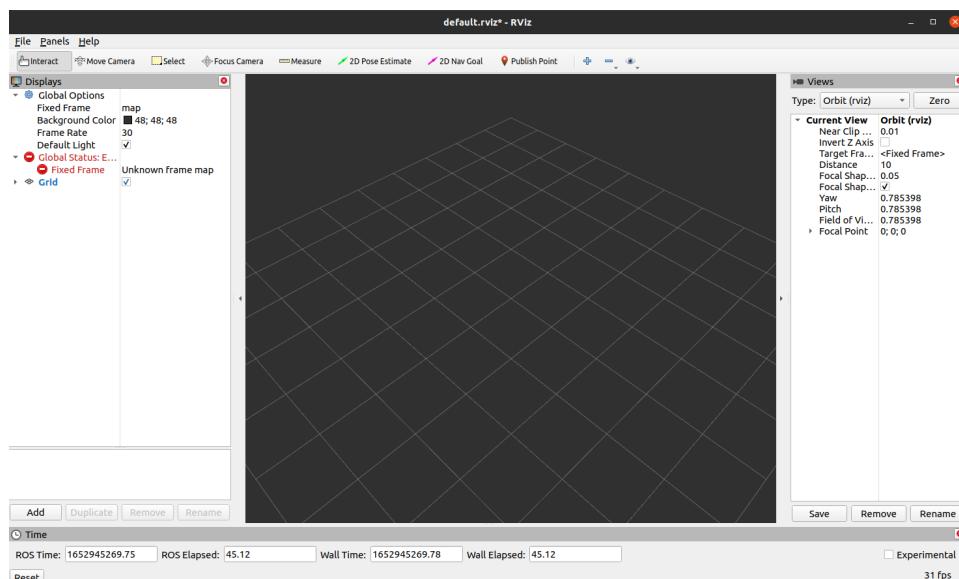
Type “rviz” or “rosrun rviz rviz” is also work the same to open rviz.

```
pi@ironx:~ $ /home/pi/roscatkin_ws/src/tesr_ros_ironx_driver_pkg/laun... /home/pi/roscatkin_ws/src/tesr_ros_ironx_driver_pkg/laun...
ROS_MASTER_URI=http://192.168.1.20:11311
[INFO] [1652944867.018674380]: rvviz version 1.14.14
[INFO] [1652944867.018717672]: compiled against Qt version 5.12.8
[INFO] [1652944867.018724896]: compiled against OGRE version 1.9.0 (Ghadamon)
[INFO] [1652944867.024291623]: Forcing OpenGL version 0.
[INFO] [1652944867.243962496]: Stereo is NOT SUPPORTED
[INFO] [1652944867.244018292]: OpenGL device: AMD RENOIR (DRM 3.42.0, 5.14.0-1008-oem, LLVM 12.0.0)
[INFO] [1652944867.244033220]: OpenGL version: 4.6 (GLSL 4.6) limited to GLSL 1.4 on Mesa system.

reneny@tesr-9939:~ $ rosrun rviz rviz
[INFO] [1652944867.018674380]: rvviz version 1.14.14
[INFO] [1652944867.018717672]: compiled against Qt version 5.12.8
[INFO] [1652944867.018724896]: compiled against OGRE version 1.9.0 (Ghadamon)
[INFO] [1652944867.024291623]: Forcing OpenGL version 0.
[INFO] [1652944867.243962496]: Stereo is NOT SUPPORTED
[INFO] [1652944867.244018292]: OpenGL device: AMD RENOIR (DRM 3.42.0, 5.14.0-1008-oem, LLVM 12.0.0)
[INFO] [1652944867.244033220]: OpenGL version: 4.6 (GLSL 4.6) limited to GLSL 1.4 on Mesa system.
```

Remote Terminal

PC/Laptop Terminal

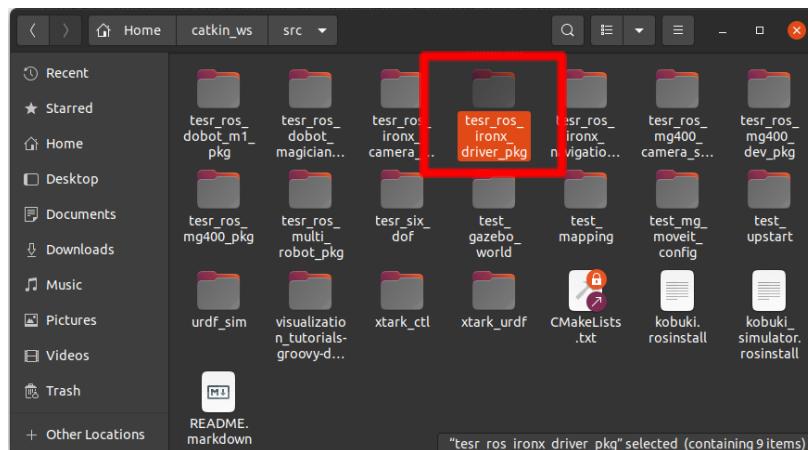


## Download iron-X's model for show on Rviz (On PC/Laptop Terminal)

To show iron-X's model. Please download tesr\_ros\_ironx\_driver\_pkg to src of your ros\_workspace (**In this case, ros\_workspace is catkin\_ws**)

from link below:

[https://drive.google.com/drive/folders/1DeE5T1W0gDIersN1\\_Ynw\\_VF\\_31eEHKRC?usp=sharing](https://drive.google.com/drive/folders/1DeE5T1W0gDIersN1_Ynw_VF_31eEHKRC?usp=sharing)



\*If you're not download package to workspace. It will cause an error from unable to find the URDF file of iron-X's model.

After that, type "cd catkin\_ws/" and type "catkin\_make" and wait until [100%]

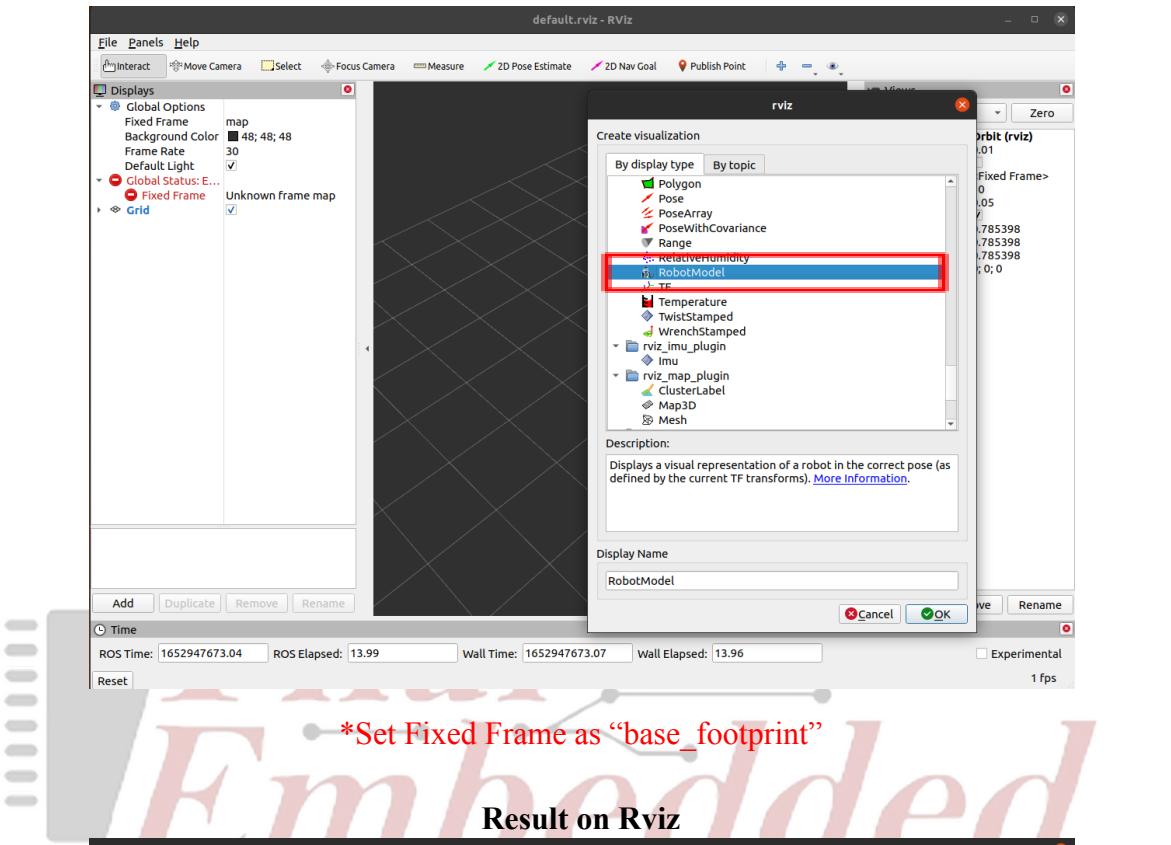
```
rengy@tesr-9939:~$ cd catkin_ws/
rengy@tesr-9939:~/catkin_ws$ catkin_make
```

[ 50%] Built target std\_msgs\_generate\_messages\_py
[ 50%] Built target std\_msgs\_generate\_messages\_nodejs
[ 53%] Built target std\_msgs\_generate\_messages\_lisp
[ 53%] Built target std\_msgs\_generate\_messages\_eus
[ 63%] Built target std\_msgs\_generate\_messages\_cpp
[ 63%] Built target \_roscpp\_generate\_messages\_check\_deps\_GetLoggers
[ 69%] Built target rospack\_msgs\_generate\_messages\_cpp
[ 69%] Built target rospack\_msgs\_generate\_messages\_eus
[ 69%] Built target rospack\_msgs\_generate\_messages\_lisp
[ 69%] Built target rospack\_msgs\_generate\_messages\_py
[ 69%] Built target std\_msgs\_generate\_messages
[ 73%] Built target roscpp\_generate\_messages\_nodejs
[ 73%] Built target rospack\_msgs\_generate\_messages\_nodejs
[ 73%] Built target roscpp\_generate\_messages\_py
[ 74%] Built target roscpp\_generate\_messages\_lisp
[ 77%] Built target roscpp\_generate\_messages\_eus
[ 78%] Built target roscpp\_generate\_messages\_cpp
[ 78%] Built target rospack\_msgs\_generate\_messages
[ 78%] Built target roscpp\_generate\_messages
[ 93%] Built target roscpp
[ 97%] Built target jpeg\_streamer
[ 97%] Built target uvc\_camera\_node
[ 97%] Built target uvc\_stereo\_node
[100%] Built target nodelet\_uvc\_camera

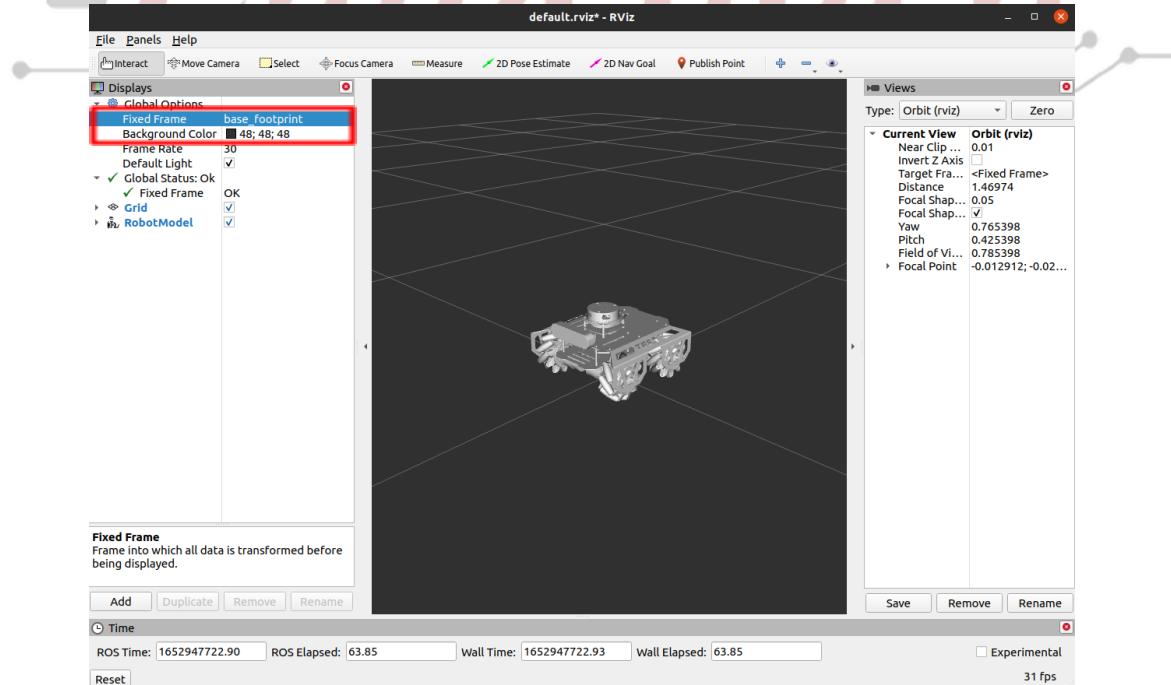


## Show iron-X's model on Rviz

To show RobotModel. Select "Add" at left below of Display panel go to "By display type" and Select “RobotModel” and then “OK”.

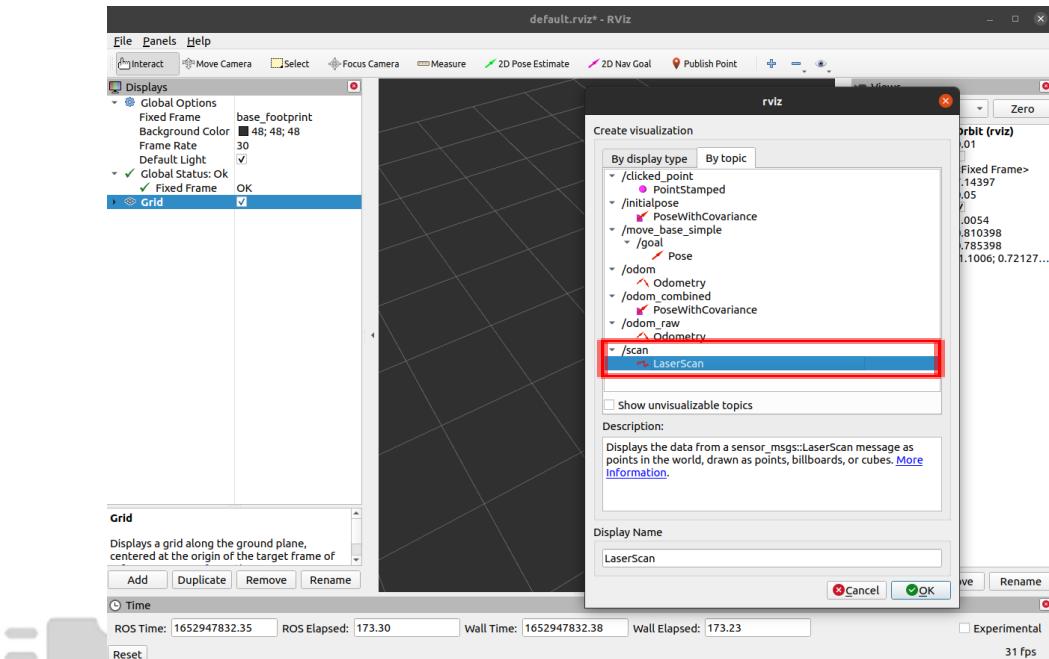


## Result on Rviz



## LaserScan from RPLIDAR A1 on Rviz

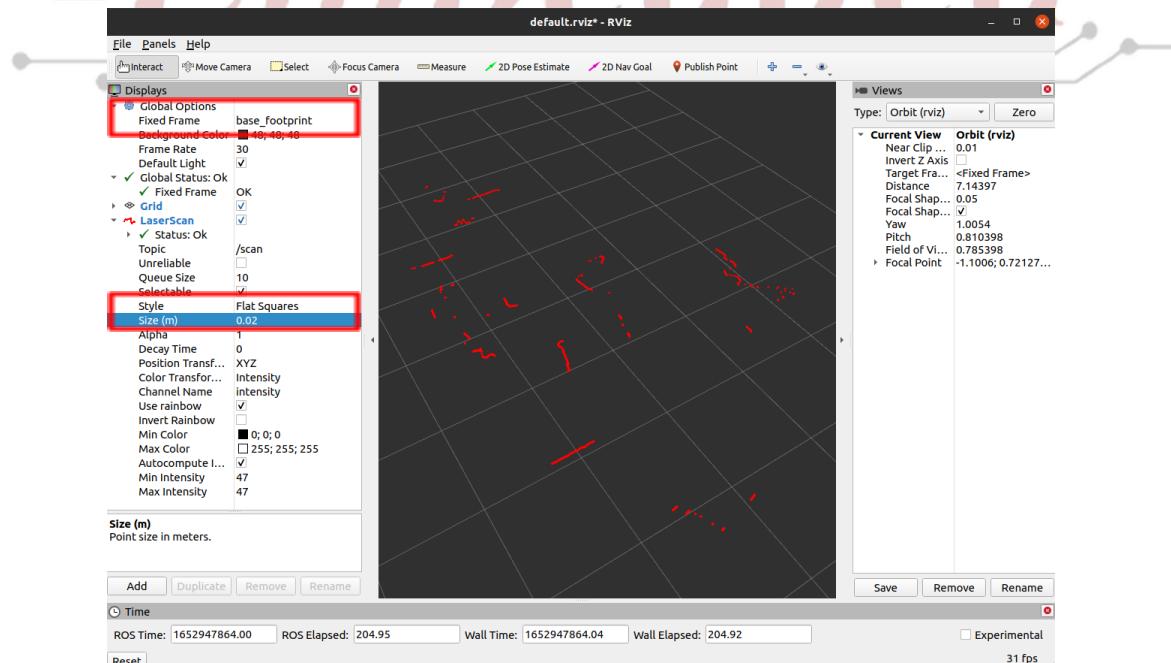
To show lidar\_scan. Select "Add" at left below of Display panel go to By topic and select topic "/scan" > "LaserScan"



\*Make sure Fixed Frame is set as “base\_footprint” or “laser”

\*\*You can change Size of LaserScan by increase values of Size(m)

## Result on Rviz



## SLAM

(Simultaneous Localization and Mapping)

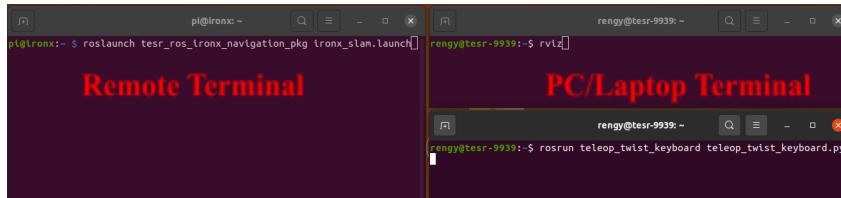
\*Keep that in mind, you **should** close the ironx Bringup before run the SLAM because within SLAM's launch file is already set to launch the ironx Bringup.

### SLAM methods

iron-X have 3 SLAM methods available for now following as:

- Gmapping SLAM method
- Hector SLAM method
- Cartographer SLAM method

### Gmapping SLAM(default)



#### (On Remote Terminal)

Type “roslaunch tesr\_ros\_ironx\_navigation\_pkg ironx\_slam.launch”  
(wait until “Publishing combined odometry on /odom\_ekf” is show)

```
process[slam_gmapping-9]: started with pid [2451]
[ INFO] [1652955686.934018534]: RPLIDAR running on ROS package rplidar_ros, SDK Version:2.0.0
[ INFO] [1652955687.012529391]: RPLIDAR S/N: 42829A87C5E392D2A5E49EF0D2573D64
[ INFO] [1652955687.012777338]: Firmware Ver: 1.25
[ INFO] [1652955687.012909284]: Hardware Rev: 5
[ INFO] [1652955687.064715869]: RPlidar health status : OK.
[ INFO] [1652955687.349116684]: current scan mode: Boost, sample rate: 8 KHz, max_distance: 12.0
m, scan frequency:10.0 Hz,
[ INFO] [1652955694.646204124]: Initializing Odom sensor
[ INFO] [1652955694.648483515]: Initializing Imu sensor
[ INFO] [1652955694.665853334]: Odom sensor activated
[ INFO] [1652955694.668369079]: Imu sensor activated
[ INFO] [1652955694.682455702]: Kalman filter initialized with odom measurement
[INFO] [1652955694.759984]: Publishing combined odometry on /odom_ekf
```

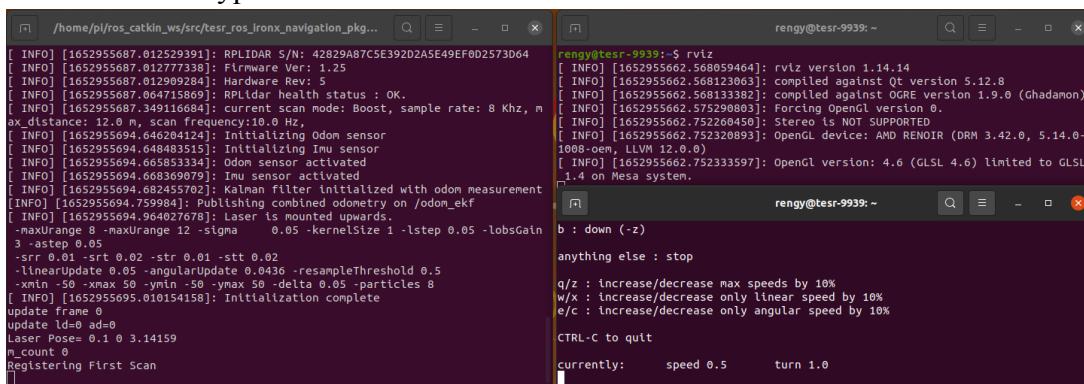
#### (On PC/Laptop Terminal)

And then, run Keyboard and Rviz

Type “rosrun teleop\_twist\_keyboard teleop\_twist\_keyboard.py”

Or you may use joystick instead of a keyboard.

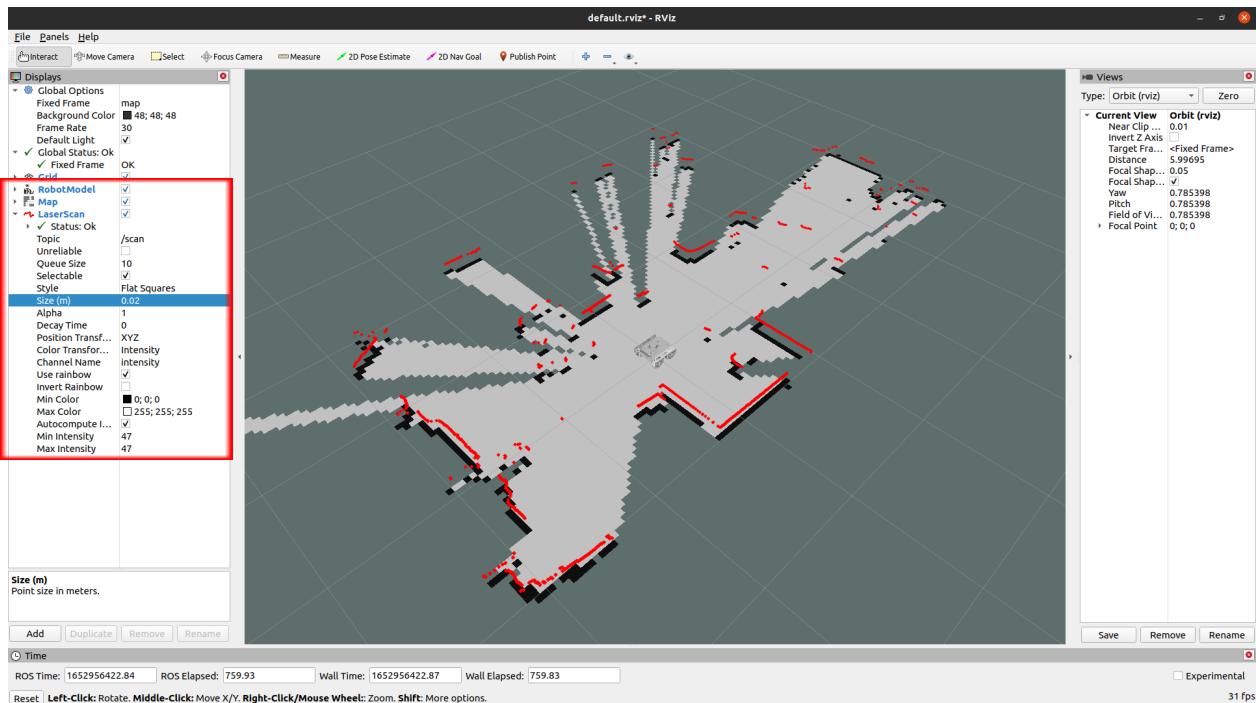
Type “rviz”



## Setup configuration to show RobotModel, Map and LaserScan in Rviz

1. Add “RobotModel” from select topic in list of “By display type”
2. Add “Map” by select from “By topic” > “/map” > Map
3. Add “LaserScan” by select from “By topic” > “/scan” > LaserScan

## After Setup, This is Result on Rviz



And this is the example of exploring around the room.



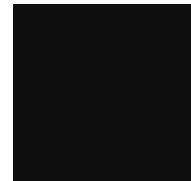
Map's display is the occupancy grid is treated the way the navigation stack treats it.



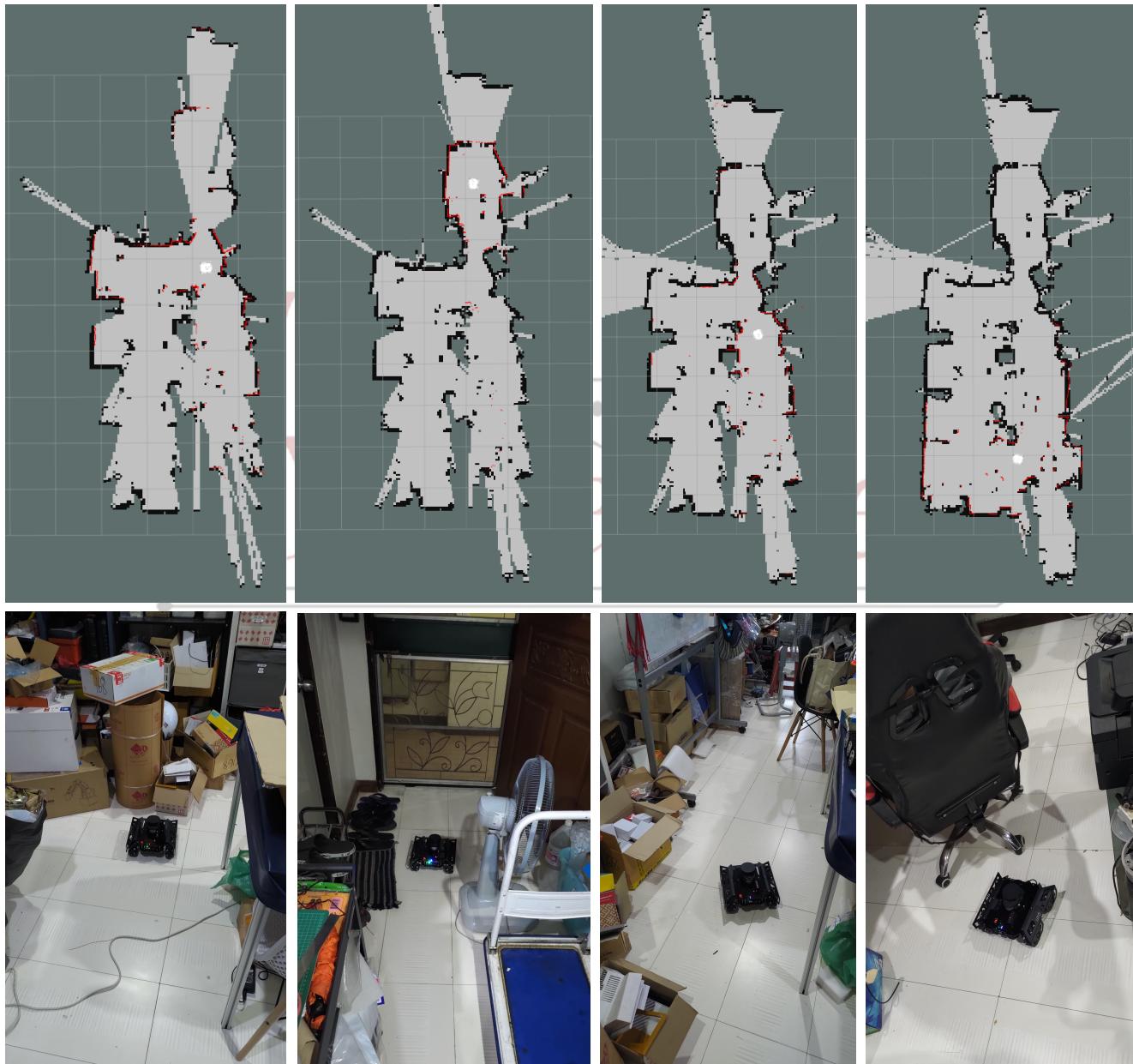
Grey is unknown space



White is unoccupied space



Black is occupied space.



## Save a map from gmapping SLAM (On Remote Terminal)

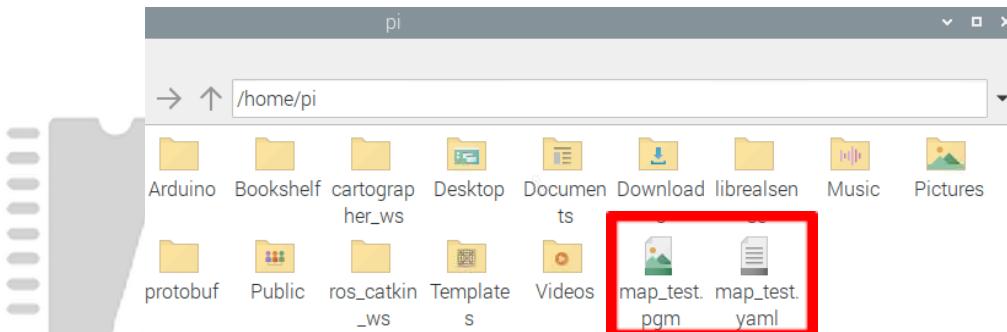
1. Open the new Remote terminal and type “rosrun map\_server map\_saver -f map\_test”

```
Last login: Fri May 20 11:37:37 2022 from 192.168.1.20
iron-X ip address: 192.168.1.21
pi@ironx:~ $ rosrun map_server map_saver -f map_test
[ INFO] [1653023348.566520081]: Waiting for the map
[ INFO] [1653023348.855629494]: Received a 1984 X 1984 map @ 0.050 m/pix
[ INFO] [1653023348.855872531]: Writing map occupancy data to map_test.pgm
[ INFO] [1653023349.625065297]: Writing map occupancy data to map_test.yaml
[ INFO] [1653023349.625847411]: Done
```

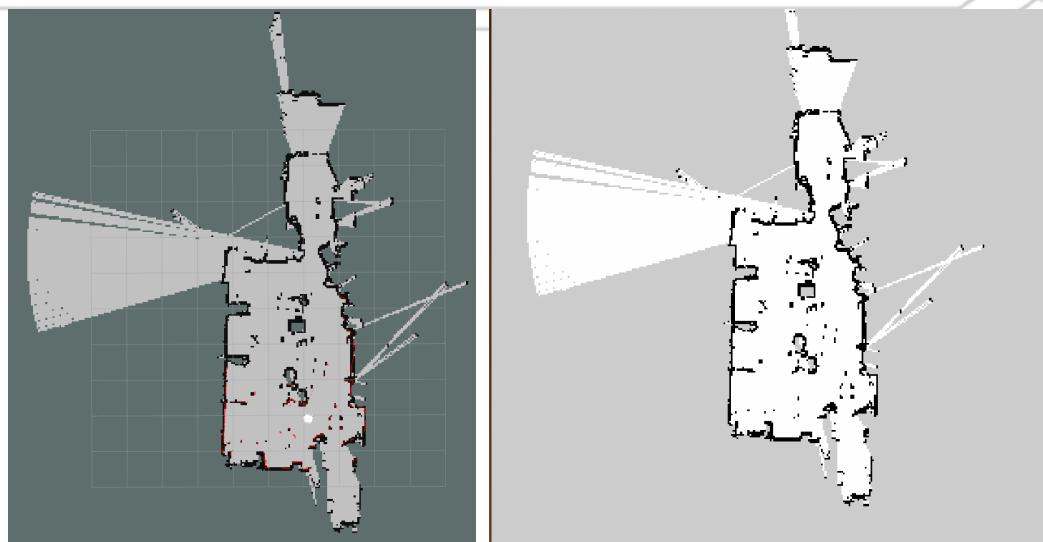
\* rosrun [rosnode\_pkg] [rosnode\_service] [-f map's name]

\*\* map\_saver retrieves map data and writes it out to map.pgm and map.yaml. Use the -f option to provide a different base name for the output files.

2. You will get the map file as "map\_test.pgm" and "map\_test.yaml".



You can see the map\_test.pgm is as same as map from Rviz.



\*Save a map by map\_saver can work on PC/Laptop and Remote Terminal



## Hector SLAM

### (On Remote Terminal)

Type “`roslaunch tesr_ros_ironx_navigation_pkg ironx_slam.launch slam_methods:=hector`”  
 (wait until “Publishing combined odometry on /odom\_ekf” is show)

```
[ INFO] [1653031791.835406440]: HectorSM p_base_frame_: base_footprint
[ INFO] [1653031791.852477589]: HectorSM p_map_frame_: map
[ INFO] [1653031791.853413304]: HectorSM p_odom_frame_: base_footprint
[ INFO] [1653031791.855281049]: HectorSM p_scan_topic_: scan_raw
[ INFO] [1653031791.856794853]: HectorSM p_use_tf_scan_transformation_: true
[ INFO] [1653031791.857652661]: HectorSM p_pub_map_odom_transform_: true
[ INFO] [1653031791.858809448]: HectorSM p_scan_subscriber_queue_size_: 5
[ INFO] [1653031791.859641905]: HectorSM p_map_pub_period_: 2.000000
[ INFO] [1653031791.862669234]: HectorSM p_update_factor_free_: 0.400000
[ INFO] [1653031791.863746800]: HectorSM p_update_factor_occupied_: 0.900000
[ INFO] [1653031791.866423687]: HectorSM p_map_update_distance_threshold_: 0.400000
[ INFO] [1653031791.867371754]: HectorSM p_map_update_angle_threshold_: 0.060000
[ INFO] [1653031791.868316932]: HectorSM p_laser_z_min_value_: -1.000000
[ INFO] [1653031791.870449749]: HectorSM p_laser_z_max_value_: 1.000000
[ INFO] [1653031798.656906204]: Initializing Imu sensor
[ INFO] [1653031798.670250510]: Imu sensor activated
[ INFO] [1653031798.760383548]: Initializing Odom sensor
[ INFO] [1653031798.797465063]: Odom sensor activated
[ INFO] [1653031798.827142394]: Kalman filter initialized with odom measurement
[INFO] [1653031798.901645]: Publishing combined odometry on /odom_ekf
```

### (On PC/Laptop Terminal)

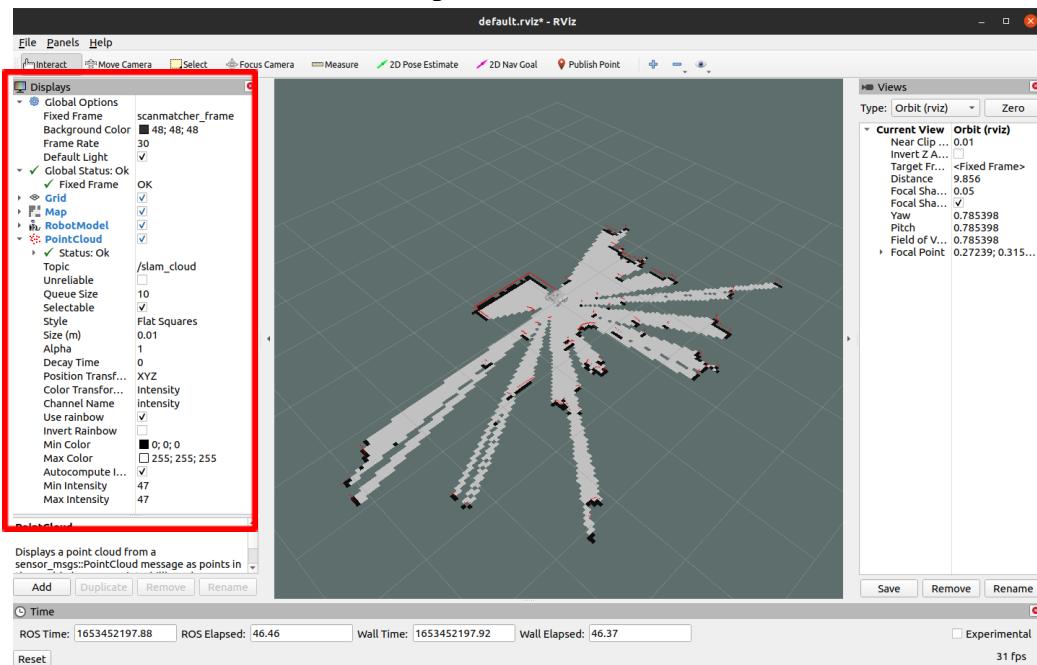
Type “`rviz`”

### Setup configuration to show RobotModel, Map and LaserScan in Rviz

1. Select Fixed Frame to “`scanmatcher_frame`”
2. Add “`RobotModel`” from select topic in list of “By display type”
3. Add “`Map`” by select from “`By topic`” > “`/map`” > `Map`
4. Add “`PointCloud`” by select from “`By topic`” > “`/slam_cloud`” > `PointCloud`

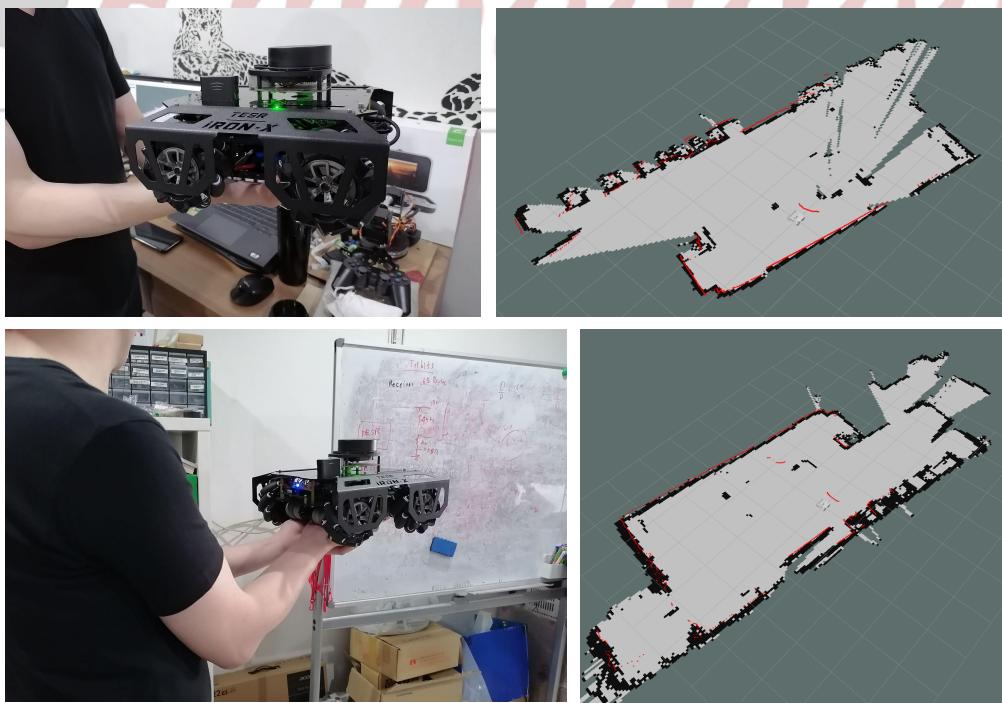


## After Setup, This is Result on Rviz



And this is an example of exploring the room. Since Hector slam using only LaserScan -Data to draw the map. So, you can carry a robot around your place.

\*Moving or rotating too fast will cause noise and error on Map drawing. So, while using hector slam you must move slow and carefully to draw the map perfectly.



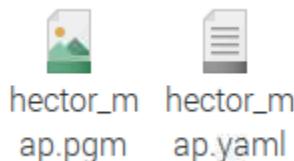
## Save map from hector slam

### (On Remote Terminal)

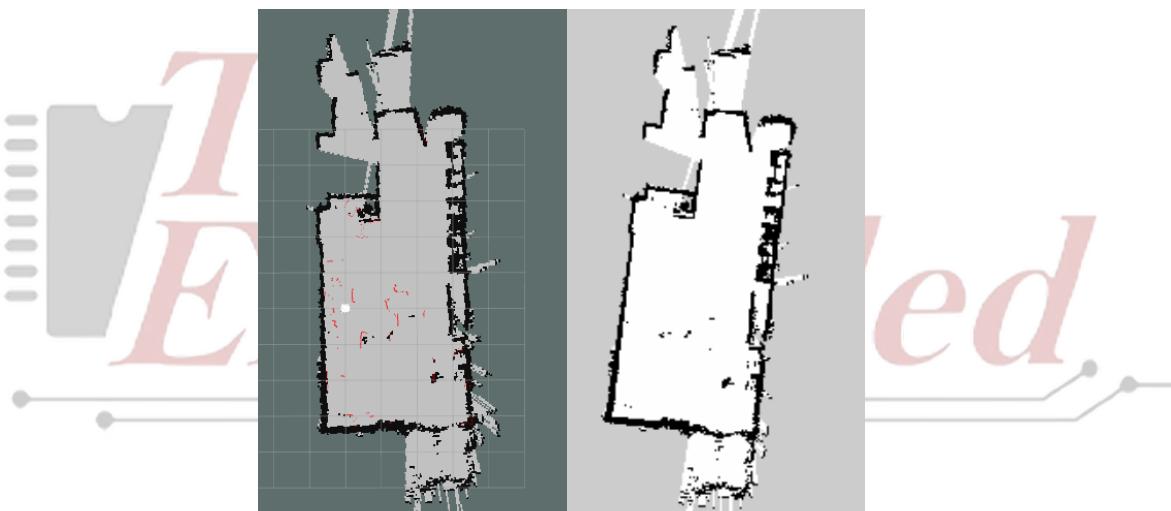
1. Open the new Remote terminal and type “rosrun map\_server map\_saver -f hector\_map”

```
Last login: Fri May 20 14:20:31 2022 from 192.168.1.20
iron-X ip address: 192.168.1.21
pi@ironx:~ $ rosrun map_server map_saver -f hector_map
[ INFO] [1653035040.944652928]: Waiting for the map
[ INFO] [1653035041.836053836]: Received a 2048 X 2048 map @ 0.050 m/pix
[ INFO] [1653035041.836298223]: Writing map occupancy data to hector_map.pgm
[ INFO] [1653035042.692920561]: Writing map occupancy data to hector_map.yaml
[ INFO] [1653035042.694223123]: Done
```

2. You will get the map file as "hector\_map.pgm" and "hector\_map.yaml".



You can see the hector\_map.pgm is as same as map from Rviz.



## The comparison between Hector SLAM with gmapping SLAM

Hector slam is uses scan-matching algorithm. So, it's more efficient to use in the complex area that mobile robot is difficult or require a long time to transport.

Pros.

- Use only LaserScan Data.
- Require low computational resources.
- More flexible to travel around the place with human's help.

Cons.

- The map is misplaced and get noise easily because it uses only LaserData.
- The map shape can be completely broken by even one mistake.
- Map's detail may not real as same as robot transportation.



# Cartographer SLAM

pi@ironx: ~

```
pi@ironx:~ $ roslaunch tesr_ros_ironx_navigation_pkg ironx_slam.launch slam_methods:=cartographer
```

reny@tesr-9939: ~

```
reny@tesr-9939:~ $ rviz
```

reny@tesr-9939: ~

```
reny@tesr-9939:~ $ rosrun teleop_twist_keyboard teleop_twist_keyboard.py
```

## (On Remote Terminal)

Type “`roslaunch tesr_ros_ironx_navigation_pkg ironx_slam.launch slam_methods:=cartographer`”  
(wait until “Publishing combined odometry on /odom\_ekf” is show)

## (On PC/Laptop Terminal)

And then, run Keyboard and Rviz

Type “rosrun teleop\_twist\_keyboard teleop\_twist\_keyboard.py”

**Or** you may use joystick instead of a keyboard

Type “rviz”

```
/home/pl/ros_catkin_ws/src/tesr_ros_ironx_navigation_pkg...          rengy@tesr-9939: ~
robot_pose_ekf (robot_pose_ekf/robot_pose_ekf)
robot_state_publisher (robot_state_publisher/robot_state_publisher)
rplidarNode (rplidar_ros/rplidarNode)

ROS_MASTER_URI=http://192.168.1.20:11311

process[ironx_driver-1]: started with pid [1991]
process[robot_pose_ekf-2]: started with pid [1992]
process[odom_ekf_node-3]: started with pid [1993]
process[robot_state_publisher-4]: started with pid [1994]
process[base_footprint_to_tmu-5]: started with pid [1998]
process[rplidar_node-6]: started with pid [2000]
process[base_footprint_to_laser-7]: started with pid [2002]
[ INFO] [1653277355.642631083]: output frame: odom
[ INFO] [1653277355.665089990]: base frame: base_footprint
process[laser_filter-8]: started with pid [2004]
process[cartographer_node-9]: started with pid [2012]
process[cartographer_occupancy_grid_node-10]: started with pid [2022]
[ INFO] [1653277356.027066354]: RPLIDAR running on ROS package rplidar_ros, SDK Version:2.0.0
[ INFO] [1653277356.106429269]: RPLIDAR S/N: 42829A87C5E392D2A5E49EF0D2573D64
[ INFO] [1653277356.106718806]: Firmware Ver: 1.25
[ INFO] [1653277356.106912343]: Hardware Rev: 5
[ INFO] [1653277356.158522003]: RPLidar health status : OK.
[ INFO] [1653277356.473914258]: current scan mode: Boost, sample rate: 8 Khz, max_distance: 12.0 m, scan frequency:10.0 Hz,
[ INFO] [1653277364.180277559]: Initializing Odom sensor
[ INFO] [1653277364.181668280]: Initializing Imu sensor
[ INFO] [1653277364.213265383]: Imu sensor activated
[ INFO] [1653277364.213577401]: Odom sensor activated
[ INFO] [1653277364.233389885]: Kalman filter initialized with odom measurement
[ INFO] [1653277364.311348]: Publishing combined odometry on /odom_ekf

rengy@tesr-9939:~$ rviz
[ INFO] [1653277582.248177816]: rviz version 1.14.14
[ INFO] [1653277582.248239193]: compiled against Qt version 5.12.8
[ INFO] [1653277582.249248210]: compiled against OGRE version 1.9.0 (Chadamon)
[ INFO] [1653277582.254117968]: Forcing OpenGL version 0.
[ INFO] [1653277582.427887315]: Stereo is NOT SUPPORTED
[ INFO] [1653277582.427943622]: OpenGL device: AMD RENOIR (DRM 3.42.0, 5.14.0-108-oem, LLVM 12.0.0)
[ INFO] [1653277582.427956105]: OpenGL version: 4.6 (GLSL 4.6) limited to GLSL 1.4 on Mesa system.

For Holonomic mode (strafing), hold down the shift key:
-----
U I O
J K L
M < >

t: up (+z)
b: down (-z)

anything else : stop

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
CTRL-C to quit

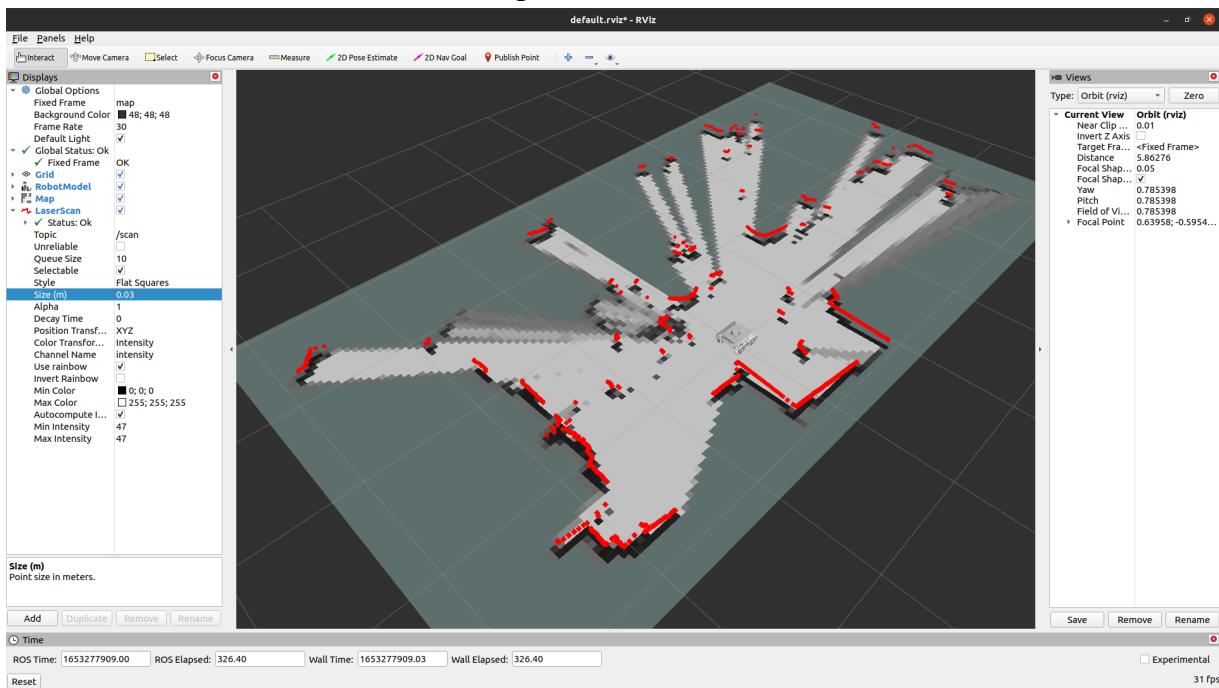
currently:      speed 0.5      turn 1.0
```

**Setup configuration to show RobotModel, Map and LaserScan in Rviz**

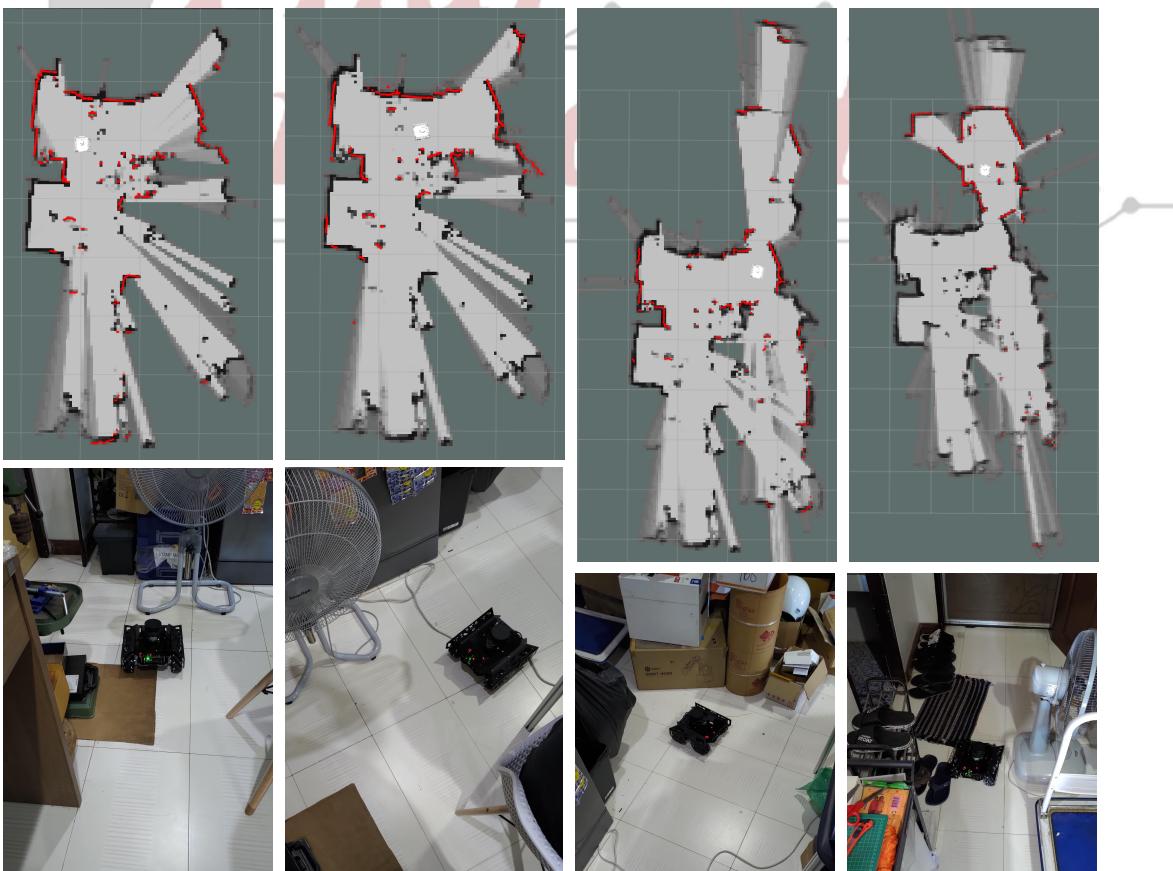
1. Add “RobotModel” from select topic in list of “By display type”
  2. Add “Map” by select from “By topic” > “/map” > Map
  3. Add “LaserScan” by select from “By topic” > “/scan” > LaserScan

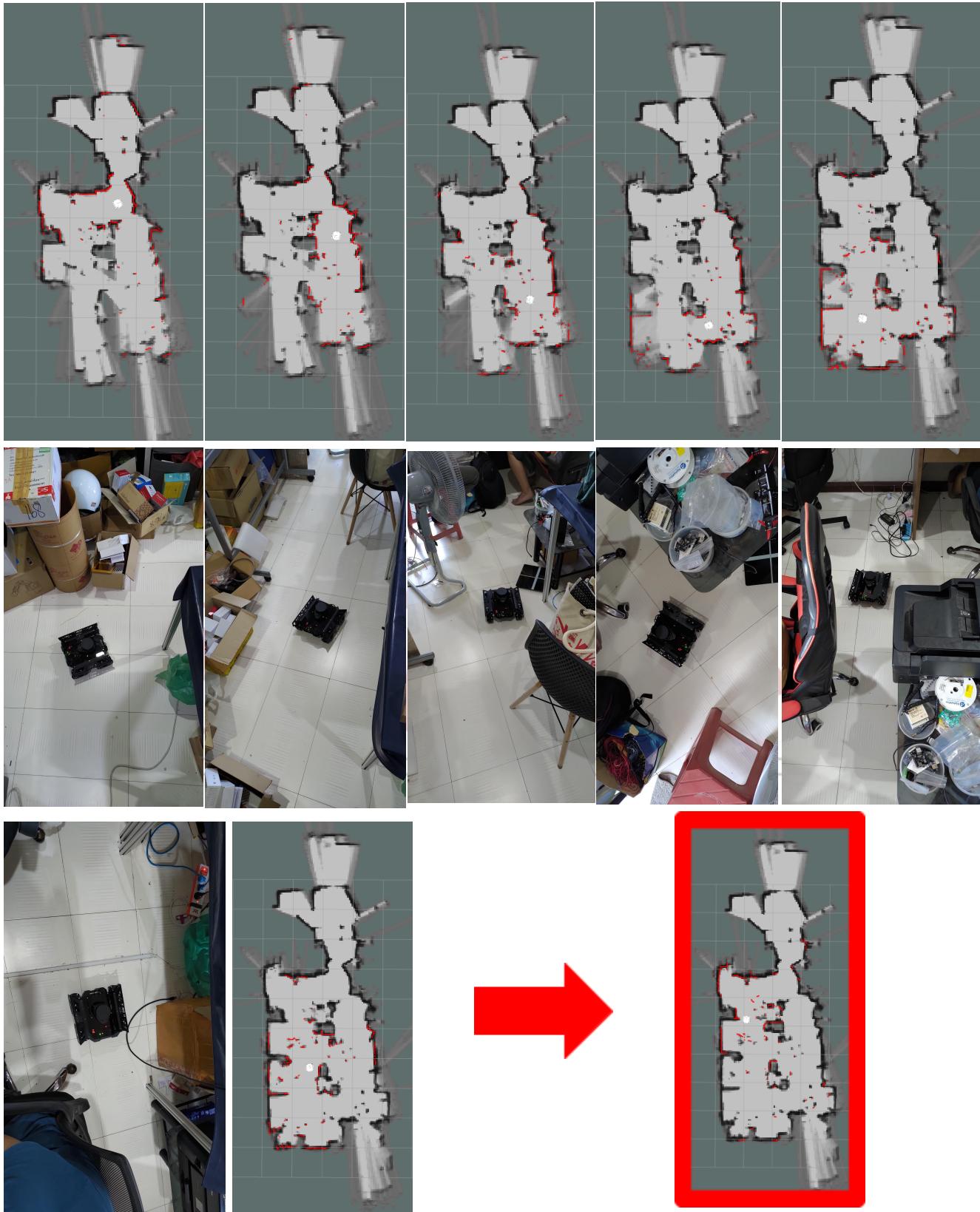


## After Setup, This is Result on Rviz



And this is an example of exploring the room. The cartographer uses odometry data, imu data, and laser scan data. So, the precision of the map is quite good.





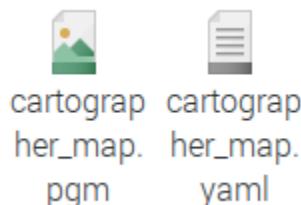
## Save map from cartographer SLAM

### (On Remote Terminal)

1. Open the new Remote terminal and type “rosrun map\_server map\_saver -f cartographer\_map”

```
Last login: Mon May 23 10:38:02 2022 from 192.168.1.20
iron-X ip address: 192.168.1.21
pi@ironx:~ $ rosrun map_server map_saver -f cartographer_map
[ INFO] [1653280703.410352398]: Waiting for the map
[ INFO] [1653280703.652752506]: Received a 321 X 170 map @ 0.050 m/pix
[ INFO] [1653280703.652966079]: Writing map occupancy data to cartographer_map.pgm
[ INFO] [1653280703.669747552]: Writing map occupancy data to cartographer_map.yaml
[ INFO] [1653280703.671779480]: Done
```

2. You will get the map file as "cartographer\_map.pgm" and "cartographer\_map.yaml".



You can see the cartographer\_map.pgm is as same as map from Rviz.



## The comparison between cartographer SLAM with gmapping SLAM

Cartographer slam's algorithm use scan-matcing and loop detection while gmapping is uses particle filter pairing algorithm. In term of data analysis, the cartographer will use odometry -data, imu data and laser scan while gmapping does not use imu data. Not only that, cartographer is capable to use RGB-D algorithm is an algorithm for mapping using depth images too.

Pros.

- More Precision and Accuracy by use combined data
- Harder to misplace when compare with gmapping
- By loop detection algorithm slam can clear occupied from noise or error.

Cons.

- Require high computational resources.
- Can be misplace with unexpect error from some data
- Can get a noise from changed environment while loop detection.



## Navigation

### Launch ironx\_nav.launch

#### (On Remote Terminal)

- Type “roslauch tesr\_ros\_ironx\_navigation\_pkg ironx\_nav.launch map\_file:=/home/pi/cartographer\_map.yaml”  
 (wait until “odom received!” showup.)

```
[ INFO] [1653283337.114945129]: Recovery behavior will clear layer 'obstacles'
[ INFO] [1653283337.184110924]: Recovery behavior will clear layer 'obstacles'
[ INFO] [1653283337.600162081]: odom received!
```

#### (On PC/Laptop Terminal)

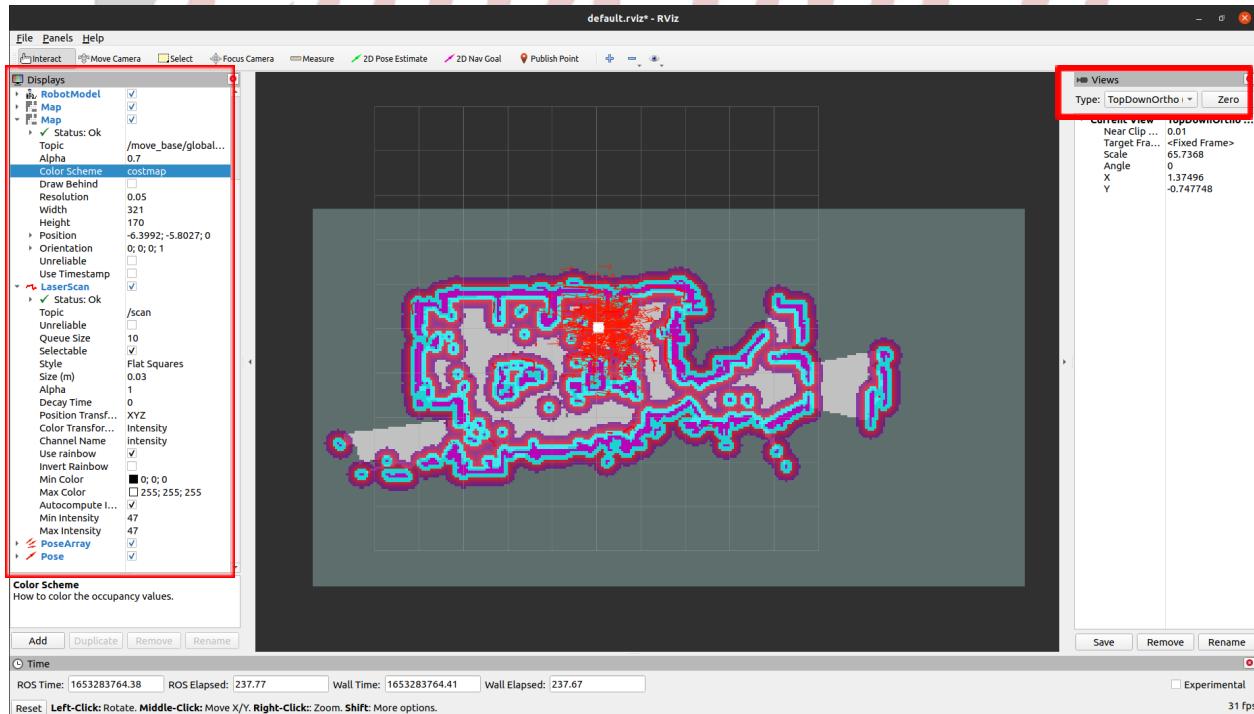
- Type “rviz” or “rosrun rviz rviz”

### Setup configuration to show RobotModel, Map and LaserScan in Rviz

- Add “RobotModel” from “By display type”
- Add topic from “By topic” following as: “/map > Map”,  
 “/move\_base > /global\_costmap > /costmap > Map” (costmap),  
 (“Color scheme” should be: costmap)  
 “/scan > LaserScan”,  
 “/particlecloud > PoseArray”,  
 “/move\_base\_simple > /goal > Pose”

\*At the top right of rviz panel you can change the point of view for different view of Rviz’s interface (“TopDownOrtho(rviz)” is Recommended for Navigation application \*default is “Orbit(rviz)”)

### After Setup, This is Result on Rviz



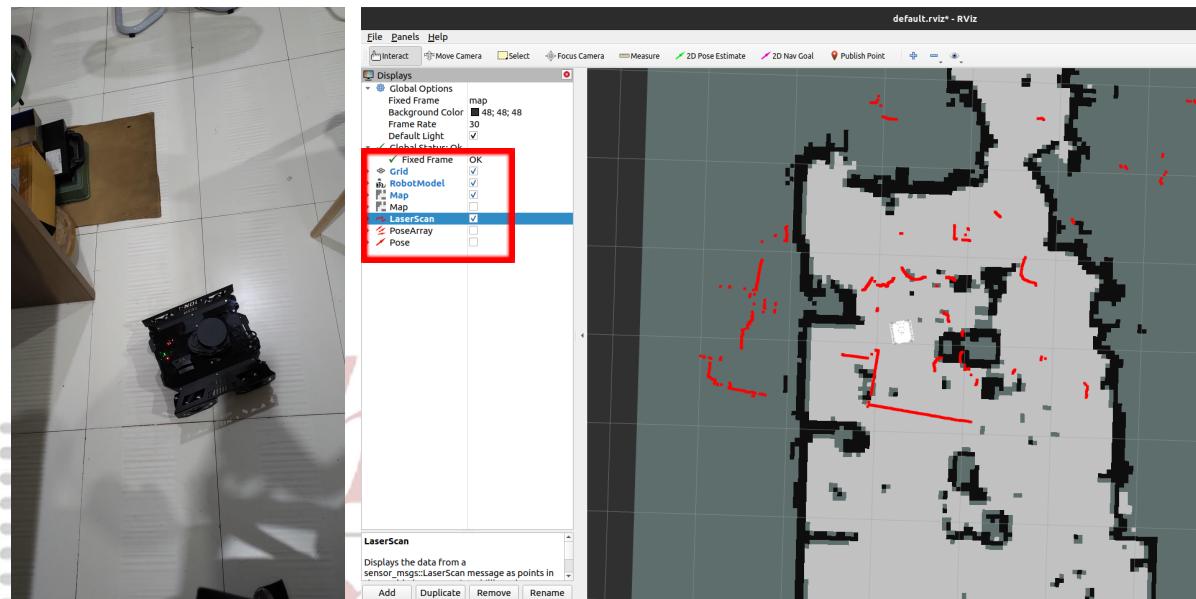
## 2D Pose Estimate

After setting, check if your robot are already at same position on the map. You can re-estimate pose of robot model on the map by click to “2D Pose Estimate”.



You can see if your robot is misplace from the map by observe to laser data or observe from posture of your robot. Compare to the posture in rviz screen.

\*you may hide Map>costmap and Pose Array to make you see the different clearly



And then, re-estimate pose by press and hold on the map the set position and drag to set the direction for robot on the map. After re-estimate pose, the position and posture will be more the same as reality.

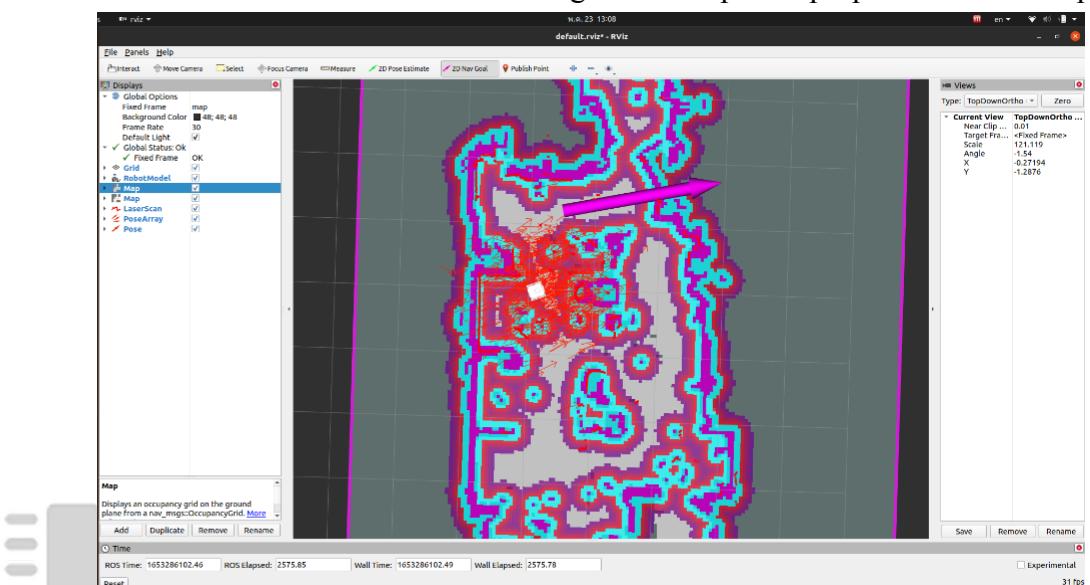


## 2D Nav Goal

Now, your iron-X will be able to use the navigation for moving around your place or for according to your map\_file by “2D Nav Goal” on tools bar of Rviz.



Click “2D Nav Goal” and Click and Drag on the map. The purple arrow will display.



The iron-X will navigate and move to the arrow’s position.



```
[ INFO] [1653286140.777954909]: Got new plan
[ INFO] [1653286140.778294665]: Goal reached
```

**Remote Terminal**



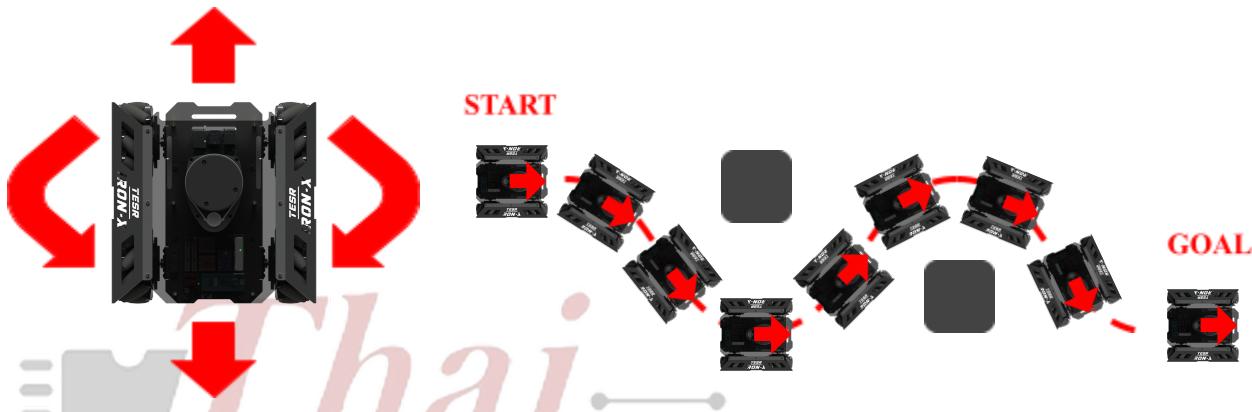
## iron-X's move\_base type

iron-X is designed to capable for move in 2 system are holonomic and non-holonomic. It can give the definition of move\_base\_type as:

- Differential drive(non-holonomic)
- Omni drive(holonomic)

### Differential Drive(non-holonomic)

Differential Drive or diff-drive is the default move\_base\_type of iron-X. This move\_base is move as car-like in use case is suitable to transport the objects more than omni-drive.



### Omni Drive(holonomic)

Omni Drive is move in holonomic system. So, it give a priority to the motions to goal more than diff-drive. It will suitable to use with robot that needed high mobility.

You can add argument “move\_base\_omni:=true” in command line to change movement of iron-X to holonomic or “omni-drive” same as below:

```
pi@ironx: ~ $ roslaunch tesr_ros_ironx_navigation_pkg ironx_nav.launch move_base_omni:=true
map_file:=~/home/pi/cartographer_map.yaml
```

“roslaunch tesr\_ros\_ironx\_navigation\_pkg ironx\_nav.launch **move\_base\_omni:=true**  
map\_file:=~/home/pi/cartographer\_map.yaml”

