

ROS gui based

<https://wiki.ros.org/blockly>

<https://wiki.ros.org/blockly>

You said:

```
triggers for man-db (2.9.1-1) ... (yoloenv) (base) sakthees-monk@saktheesmonk-desktop:~$ sudo
apt update Hit:2 http://in.archive.ubuntu.com/ubuntu focal InRelease Get:3
http://packages.ros.org/ros/ubuntu focal InRelease [4,679 B] Hit:4
http://security.ubuntu.com/ubuntu focal-security InRelease Get:5
http://in.archive.ubuntu.com/ubuntu focal-updates InRelease [128 kB] Err:3
http://packages.ros.org/ros/ubuntu focal InRelease The following signatures couldn't be verified
because the public key is not available: NO_PUBKEY F42ED6FBAB17C654 Hit:6
http://in.archive.ubuntu.com/ubuntu focal-backports InRelease Hit:7
http://in.archive.ubuntu.com/ubuntu focal-proposed InRelease Get:1
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64 InRelease [1,581
B] Err:1 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64 InRelease
The following signatures couldn't be verified because the public key is not available: NO_PUBKEY
A4B469963BF863CC Reading package lists... Done W: GPG error:
http://packages.ros.org/ros/ubuntu focal InRelease: The following signatures couldn't be verified
because the public key is not available: NO_PUBKEY F42ED6FBAB17C654 E: The repository
'http://packages.ros.org/ros/ubuntu focal InRelease' is not signed. N: Updating from such a
repository can't be done securely, and is therefore disabled by default. N: See apt-secure(8)
manpage for repository creation and user configuration details. W: GPG error:
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64 InRelease: The
following signatures couldn't be verified because the public key is not available: NO_PUBKEY
A4B469963BF863CC E: The repository
'http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64 InRelease' is not
signed. N: Updating from such a repository can't be done securely, and is therefore disabled by
default. N: See apt-secure(8) manpage for repository creation and user configuration details.
(yoloenv) (base) sakthees-monk@saktheesmonk-desktop:~$ sudo apt install ros-noetic-desktop-
full Reading package lists... Done Building dependency tree Reading state information... Done E:
Unable to locate package ros-noetic-desktop-full (yoloenv) (base) sakthees-monk@saktheesmonk-
desktop:~$
```

ChatGPT said:

You're facing two main issues here:

Issue 1: Missing GPG keys

This is causing the following repositories to be considered **untrusted**:

- `http://packages.ros.org/ros/ubuntu`

- https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64

You need to **add the missing public keys** to fix this.

Fix for ROS (missing key F42ED6FBAB17C654)

```
sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key F42ED6FBAB17C654
```

Fix for CUDA (missing key A4B469963BF863CC)

```
sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key A4B469963BF863CC
```

Alternatively, using `curl` (sometimes more reliable):

```
# ROS key
curl -sSL 'http://keyserver.ubuntu.com/pks/lookup?op=get&search=0xF42ED6FBAB17C654' | sudo apt-key add -

# CUDA key
curl -sSL 'http://keyserver.ubuntu.com/pks/lookup?op=get&search=0xA4B469963BF863CC' | sudo apt-key add -
```

Then run:

```
sudo apt update
```

Issue 2: ros-noetic-desktop-full not found

This happens usually because:

1. The **ROS repository isn't set up correctly**, or
2. `apt update` failed earlier due to the GPG key error.

Full Fix: ROS Noetic Installation for Ubuntu 20.04 (Focal)

Follow these steps **after fixing the keys**:

1. Add the ROS repository:

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu focal main" > /etc/apt/sources.list.d/ros1-latest.list'
```

2. Add the key again if not already:

```
sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key F42ED6FBAB17C654
```

3. Update package index:

```
sudo apt update
```

4. Install ROS Noetic Desktop Full:

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

Pro Tip:

If you're using **Ubuntu 22.04 or newer**, `ros-noetic` will not work — you'll need **ROS 2** (like **Humble** or **Iron**). Noetic is only officially supported on **Ubuntu 20.04**.

Would you like a full ROS Noetic setup script (including `rosdep`, `catkin`, etc.)?

You said:

```
nk@saktheesmonk-desktop:~$ sudo apt install ros-noetic-PACKAGE Reading package lists... Done
Building dependency tree Reading state information... Done E: Unable to locate package ros-noetic-
PACKAGE
```

ChatGPT said:

The error:

```
E: Unable to locate package ros-noetic-PACKAGE
```

is because `PACKAGE` is just a **placeholder**, not an actual ROS package name.

Correct Usage

If you want to install a specific ROS Noetic package, replace `PACKAGE` with a **real package name**, like:

Example:

```
sudo apt install ros-noetic-turtlesim
```

Or, if you want **everything**, install:

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

How to find available ROS Noetic packages:

You can search for available packages like this:

```
apt-cache search ros-noetic
```

This will list all available ROS Noetic packages in the repository.

Full ROS Noetic Setup (if not done yet)

1. Add the ROS repo (if not already done):

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu focal main" >
/etc/apt/sources.list.d/ros1-latest.list'
```

2. Add the ROS GPG key:

```
sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key
F42ED6FBAB17C654
```

3. Update the package list:

```
sudo apt update
```

4. Install desktop-full version:

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

5. Initialize rosdep:

```
sudo rosdep init
rosdep update
```

6. Add source to your shell:

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

Let me know if you want help installing a **specific package** or doing a **workspace (catkin) setup**.

You said:

```
sudo apt install python3-rosdep2 give cionda version
```

ChatGPT said:

To install rosdep for ROS Noetic (Python 3 version) **with Conda compatibility**, you need to **avoid conflicts between system Python (used by ROS) and Conda Python**.

Problem with Your Command:

```
sudo apt install python3-rosdep2
```

This tries to install a system package named `python3-rosdep2`, which **doesn't exist**. The correct system package name is:

```
sudo apt install python3-rosdep
```

But this installs it into the **system Python, not Conda**.

If You're Using Conda:

If you want rosdep inside a **Conda environment**, follow this method:

Step 1: Activate your conda environment

```
conda activate yoloenv # or whatever your env is
```

Step 2: Install dependencies via pip (inside conda)

```
pip install rosdep
```

But note: `rosdep` needs system tools (`apt`, `sudo`, etc.) and root access — even when installed in Conda.

Best Practice (Hybrid Approach):

ROS tooling is designed around system Python. Mixing with Conda **can break things**.

So, recommended:

- Use **system Python** for all ROS-related tools like `rosdep`, `roscpp`, `catkin_tools`, etc.
- Use Conda for **ML/CV libraries** like PyTorch, OpenCV, YOLO, etc.

You can bridge them via ROS Python APIs or messaging (`rospy`, `cv_bridge`, etc.).

Summary of Fixes:

Goal	Command
Install <code>rosdep</code> system-wide	<code>sudo apt install python3-rosdep</code>
Initialize <code>rosdep</code>	<code>sudo rosdep init && rosdep update</code>
Install in Conda (optional)	<code>pip install rosdep</code> (but not recommended for full ROS setups)
Search package	<code>apt-cache search rosdep</code>

Let me know your goal:

- Do you want **ROS fully inside Conda**?
- Or just use Conda for some parts and ROS system-wide?

I can guide setup accordingly.

You said:

If You're Using Conda: If you want `rosdep` inside a Conda environment, follow this method:

Step 1: Activate your conda environment `conda activate yoloenv` # or whatever your env is
Step 2: Install dependencies via pip (inside conda) `pip install rosdep`
But note: `rosdep` needs system tools (`apt`, `sudo`, etc.) and root access — even when installed in Conda.
Best Practice (Hybrid Approach): ROS tooling is designed around system Python. Mixing with Conda can break things.
So, recommended: Use system Python for all ROS-related tools like `rosdep`, `roscpp`, `catkin_tools`, etc. Use Conda for ML/CV libraries like PyTorch, OpenCV, YOLO, etc. You can bridge them via

ROS Python APIs or messaging (rospy, cv_bridge, etc.). Summary of Fixes: Goal Command
Install rosdep system-wide sudo apt install python3-rosdep Initialize rosdep sudo rosdep init &&
rosdep update Install in Conda (optional) pip install rosdep (but not recommended for full ROS
setups) Search package apt-cache search rosdep Let me know your goal: Do you want ROS fully
inside Conda? Or just use Conda for some parts and ROS system-wide? I can guide setup
accordingly. create env initialise

ChatGPT said:

Great! Since you're asking to **create the environment and initialize it**, I'll assume you want to:

- Use **ROS Noetic** and its core tools system-wide (as recommended),
 - Use **Conda for machine learning, OpenCV, and YOLO** work,
 - Connect both environments **smoothly** (hybrid setup).
-

Step-by-Step: Create & Initialize Hybrid Conda + ROS Environment

Step 1: Create the Conda Environment (Python 3.8 — compatible with ROS Noetic)

```
conda create -n yolovenv python=3.8 -y
conda activate yolovenv
```

ROS Noetic works best with Python 3.8.

Step 2: Install ML/CV Libraries inside Conda

You can install these inside the yolovenv:

```
# PyTorch + OpenCV + YOLO + other tools
conda install pytorch torchvision torchaudio -c pytorch
conda install opencv
pip install ultralytics # for YOLOv8+
```

Step 3: Install ROS-related tools system-wide (do NOT install them in conda)

3.1 Install rosdep:

```
sudo apt update
sudo apt install python3-rosdep
```

3.2 Initialize rosdep (first-time only):

```
sudo rosdep init
rosdep update
```

Step 4: Bridge Conda & ROS

Now add ROS setup to your shell (optional):

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

Then, whenever needed:

```
# Activate Conda for ML work
conda activate yoloenv
```

```
# Source ROS separately if not already in bashrc
source /opt/ros/noetic/setup.bash
```

Now you can run `rospy`, `roslaunch`, etc., and also import ML models inside the same terminal session.

You're Ready!

To test:

- Run:
 - `roscore`
 - In another terminal:
 - ```
conda activate yoloenv
source /opt/ros/noetic/setup.bash
python3
>>> import rospy
>>> print("ROS & Conda working together!")
```
- 

Would you like a **shell script** to automate all these steps for future machines or VMs?

### You said:

give rosudemy for yoloenv env name \

ChatGPT said:

Certainly! Here's a shell script that will:

```
Create a Conda environment named rosudemy
Install Python 3.8 (compatible with ROS Noetic)
Install ML/CV tools inside Conda (PyTorch, OpenCV, YOLO)
Set up ROS Noetic system-wide (not in Conda)
Bridge ROS with your Conda environment
```

---

## Shell Script: `setup_rosudemy.sh`

```
#!/bin/bash
```

```

echo " Creating Conda environment: rosudemy (Python 3.8)..."
conda create -n rosudemy python=3.8 -y

echo " Activating rosudemy..."
conda activate rosudemy

echo " Installing ML and CV libraries into rosudemy..."
conda install pytorch torchvision torchaudio -c pytorch -y
conda install opencv -y
pip install ultralytics

echo " Installing rosdep system-wide (Ubuntu)..."
sudo apt update
sudo apt install python3-rosdep -y

echo " Initializing rosdep (if first time)..."
sudo rosdep init 2>/dev/null || echo "rosdep already initialized"
rosdep update

echo " Sourcing ROS Noetic in ~/.bashrc if not already there..."
grep -qxF 'source /opt/ros/noetic/setup.bash' ~/.bashrc || echo 'source
/opt/ros/noetic/setup.bash' >> ~/.bashrc
source ~/.bashrc

echo " Setup complete!"

echo -e "\n Use this in your terminal when working:\n"
echo "conda activate rosudemy"
echo "source /opt/ros/noetic/setup.bash"

```

---

## How to Use:

### 1. Save it:

```

nano setup_rosudemy.sh
Paste the code and save (Ctrl+O, Enter, Ctrl+X)

```

### 2. Make it executable:

```

chmod +x setup_rosudemy.sh

```

### 3. Run it:

```

./setup_rosudemy.sh

```

---

Would you like me to add optional support for **Jupyter, matplotlib, or catkin workspace setup** inside the **rosudemy** environment as well?

#### You said:

```

(base) sakthees-monk@saktheesmonk-desktop:~$ conda create -n rosudemy python=3.8 -y
CondaToSNonInteractiveError: Terms of Service have not been accepted for the following
channels. Please accept or remove them before proceeding: • https://repo.anaconda.com/pkgs/main •
https://repo.anaconda.com/pkgs/r To accept a channel's Terms of Service, run the following and
replace CHANNEL with the channel name/URL: ► conda tos accept --override-channels --channel

```



CHANNEL To remove channels with rejected Terms of Service, run the following and replace CHANNEL with the channel name/URL: ▶ `conda config --remove channels CHANNEL (base)`  
sakthees-monk@saktheesmonk-desktop:~\$

ChatGPT said:

You're seeing this error because **Conda now requires explicit acceptance of the Terms of Service (ToS)** for some official channels (like `https://repo.anaconda.com/pkg/main`).

---

## To Fix the Error

You need to manually **accept the ToS** for the required channels.

Run the following commands **exactly**:

```
conda tos accept --override-channels --channel
https://repo.anaconda.com/pkg/main
conda tos accept --override-channels --channel https://repo.anaconda.com/pkg/r
```

---

## Then retry:

```
conda create -n rosudemy python=3.8 -y
```

---

## Optional (View all ToS-required channels):

```
conda tos list
```

This shows channels that still need your approval.

---

Let me know if you want to install from only community channels (like `conda-forge`) and avoid Anaconda official ones.

**You said:**

```
usage: conda tos [-V] [-c CHANNEL] [--override-channels] [-n ENVIRONMENT | -p PATH] [--
site | --system | --user | --env | --tos-root TOS_ROOT] [--cache-timeout CACHE_TIMEOUT | --
ignore-cache] [--json] [-v] [-h] {accept,reject,view,interactive,info,clean} ... conda tos: error:
argument cmd: invalid choice: 'list' (choose from accept, reject, view, interactive, info, clean)
```

ChatGPT said:

You're right — `conda tos` does **not** have a `list` command. That was a mistake on my part. Thanks for catching it.

---

## To proceed, just do this:

Run these two commands **exactly** to accept the required channels:

```
conda tos accept --override-channels --channel
https://repo.anaconda.com/pkgs/main
conda tos accept --override-channels --channel https://repo.anaconda.com/pkgs/r
```

Then try:

```
conda create -n rosudemy python=3.8 -y
```

---

### Optional: View accepted ToS (for verification)

You can use the following to **view the current status**:

```
conda tos info
```

It will show which channels' Terms of Service have been accepted or rejected.

---

Let me know if you'd prefer to switch to only using `conda-forge` (no Anaconda ToS required).

**You said:**

```
#!/bin/bash echo " Creating Conda environment: rosudemy (Python 3.8)..." conda create -n
rosudemy python=3.8 -y echo " Activating rosudemy..." conda activate rosudemy echo "
Installing ML and CV libraries into rosudemy..." conda install pytorch torchvision torchaudio -c
pytorch -y conda install opencv -y pip install ultralytics echo " Installing rosdep system-wide
(Ubuntu)..." sudo apt update sudo apt install python3-rosdep -y echo " Initializing rosdep (if first
time)..." sudo rosdep init 2>/dev/null || echo "rosdep already initialized" rosdep update echo "
Sourcing ROS Noetic in ~/.bashrc if not already there..." grep -qxF 'source
/opt/ros/noetic/setup.bash' ~/.bashrc || echo 'source /opt/ros/noetic/setup.bash' >> ~/.bashrc source
~/.bashrc echo " Setup complete!" echo -e "\n Use this in your terminal when working:\n"
echo "conda activate rosudemy" echo "source /opt/ros/noetic/setup.bash" for yolo v5 yolov11 create
```

**ChatGPT said:**

Thanks! You're asking to modify the script for **YOLOv5 and YOLOv11** inside the same **rosudemy** Conda environment. Since **YOLOv11 is unofficial** (likely just a future/forked version), the goal is:

- Install everything required for **YOLOv5** (Ultralytics repo, PyTorch, OpenCV, etc.)
  - Make the setup extensible for **future YOLO versions** like YOLOv11 (assuming you'll clone/run it manually)
- 

### Updated Script for YOLOv5 + Future YOLO (e.g., YOLOv11)

```
#!/bin/bash

echo " Creating Conda environment: rosudemy (Python 3.8)..."
conda create -n rosudemy python=3.8 -y

echo " Activating rosudemy..."
Use conda shell integration to activate env in script
eval "$(conda shell.bash hook)"
```

```

conda activate rosudemy

echo " Installing ML and CV libraries into rosudemy..."
conda install pytorch torchvision torchaudio -c pytorch -y
conda install opencv -y

echo " Installing YOLOv5 tools (Ultralytics)..."
pip install ultralytics # Includes YOLOv5 CLI

echo " Cloning YOLOv5 for custom usage..."
git clone https://github.com/ultralytics/yolov5.git
cd yolov5
pip install -r requirements.txt
cd ..

echo " (Optional) Clone YOLOv11 manually if available/forked."

echo " Installing rosdep system-wide (Ubuntu)..."
sudo apt update
sudo apt install python3-rosdep -y

echo " Initializing rosdep (if first time)..."
sudo rosdep init 2>/dev/null || echo "rosdep already initialized"
rosdep update

echo " Sourcing ROS Noetic in ~/.bashrc if not already there..."
grep -qxF 'source /opt/ros/noetic/setup.bash' ~/.bashrc || echo 'source /opt/ros/noetic/setup.bash' >> ~/.bashrc
source ~/.bashrc

echo " Setup complete!"

echo -e "\n Use this in your terminal when working:\n"
echo "conda activate rosudemy"
echo "source /opt/ros/noetic/setup.bash"
echo "cd yolov5 && python detect.py --source 0 # (for webcam test)"

```

---

### What this sets up:

- **YOLOv5** via `ultralytics` pip module + full repo
  - Ready to clone and run future versions like YOLOv11
  - System-wide ROS tools (Noetic + rosdep)
  - Conda Python 3.8 for model compatibility
  - Easy terminal re-entry setup
- 

Let me know if you also want:

- Automatic YOLOv11 fork from GitHub if available
- Jupyter Notebook setup in rosudemy
- `cv_bridge` installation between OpenCV and ROS

**You said:**

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" Skip end-of-life distro "foxy" Skip end-of-life distro  
"galactic" Skip end-of-life distro "groovy" Add distro "humble" Skip end-of-life distro "hydro" Skip  
end-of-life distro "indigo" Skip end-of-life distro "iron" Skip end-of-life distro "jade" Add distro  
"jazzy" Add distro "kilted" Skip end-of-life distro "kinetic" Skip end-of-life distro "lunar" Skip end-  
of-life distro "melodic" Skip end-of-life distro "noetic" Add distro "rolling" which to add noetic  
install how to check ros properly with desktop full packages installed

**ChatGPT said:**

You're asking two things:

---

## 1. You ran `rosdep update` and saw:

Skip end-of-life distro "noetic"

This is **normal** — ROS considers Noetic **end-of-life upstream (May 2025)**, so `rosdep` doesn't actively pull dependency updates for it anymore, but it still works if already installed.

**If your system already has Noetic installed**, it's fine to continue using it.

---

## 2. How to check if ROS Noetic Desktop Full is installed correctly?

Run the following checks one by one:

---

### A. Check installation directory:

```
ls /opt/ros/noetic
```

You should see folders like `bin`, `lib`, `share`, `setup.bash`, etc.

---

### B. Check if core ROS command works:

```
roscore
```

You should see `roscore` running with logs like:

```
... started core service [/rosout]
```

---

### C. Check if some key packages exist (e.g., `rviz`, `rqt`, etc.)

```
roslaunch rviz rviz
roslaunch rqt_gui rqt_gui
```

If you see GUI windows opening, desktop packages are installed.

---

### D. Check with `dpkg` if full package is installed:

```
dpkg -l | grep ros-noetic-desktop-full
```

If installed, you'll see output like:

```
ii ros-noetic-desktop-full 1.15.11-1focal all ROS Desktop Full install
```

---

### E. Confirm with `rosversion`:

```
rosversion -d
```

Should return:

```
noetic
```

---

### F. Run a test package:

Try this:

```
sudo apt install ros-noetic-turtlesim
roslaunch turtlesim turtlesim_node
```

You should see a turtle simulation window.

---

### If `ros-noetic-desktop-full` was not installed:

Run:

```
sudo apt update
sudo apt install ros-noetic-desktop-full
```

---

Let me know if you want a command to list **all installed ROS Noetic packages**, or verify `catkin/roslaunch/rospy` etc.

**You said:**

```
osrun rqt_gui rqt_gui Traceback (most recent call last): File "/opt/ros/noetic/lib/rqt_gui/rqt_gui",
line 6, in <module> import rospkg ModuleNotFoundError: No module named 'rospkg'
"ModuleNotFoundError: No module named 'rospkg' (rosudemey) sakthees-monk@saktheesmonk-
desktop:~$ dpkg -l | grep ros-noetic-desktop-full ii ros-noetic-desktop-full 1.5.0-
```

1focal.20250521.014741 amd64 A metapackage to aggregate several packages. (rosudemy)  
sakthees-monk@saktheesmonk-desktop:~\$

ChatGPT said:

Great — you've confirmed that:

**ros-noetic-desktop-full** is installed (via dpkg)

But you're seeing an error:

ModuleNotFoundError: No module named 'rospkg'

This means that when running `roslaunch rqt_gui rqt_gui`, it's using your **Conda Python**, which doesn't have the ROS Python packages.

---

## Why This Happens

You're inside the `rosudemy` Conda environment:

(rosudemy) sakthees-monk@saktheesmonk-desktop:~\$

So when you run a script, it's using Conda's Python (e.g., `/home/.../miniconda3/envs/rosudemy/bin/python`) — which doesn't have `rospkg`, `rospy`, etc.

---

## Solution: Use System Python for ROS commands

To properly run `roslaunch`, `rqt`, `roscore`, etc., you should **exit Conda**, or avoid letting Conda override `PYTHONPATH`.

### Option 1: Deactivate Conda before running ROS

```
conda deactivate
roslaunch rqt_gui rqt_gui
```

This will use the system Python from `/opt/ros/noetic`, where `rospkg` and other ROS libraries are properly installed.

---

### Option 2: Add system Python to `rosudemy` (not recommended, but possible)

If you absolutely want to run ROS code inside Conda (not just interact with it), you'll need to install ROS Python packages into Conda (but mixing can break things):

```
pip install rospkg catkin_pkg roscpp rosinstall rosinstall_generator wstool
```

But even with this, GUI tools like `rqt` or `rviz` are compiled C++ binaries, and still rely on `/opt/ros/noetic`.

---

## Quick Test:

To confirm you're using system Python, run:

```
which python
```

Expected output when **not** in Conda:

```
/usr/bin/python3
```

Expected output in Conda:

```
/home/youruser/miniconda3/envs/rosudemy/bin/python
```

---

## Best Practice Recap:

| Task                                   | Environment          | How to run                        |
|----------------------------------------|----------------------|-----------------------------------|
| roscore, rosrun, rqt_gui               | System shell         | conda deactivate, then run        |
| YOLO, ML models                        | rosudemy Conda env   | conda activate rosudemy           |
| Use both                               | System + Conda combo | Source ROS after activating Conda |
| conda activate rosudemy                |                      |                                   |
| source /opt/ros/noetic/setup.bash      |                      |                                   |
| python3 # use rospy, etc. inside Conda |                      |                                   |

---

Would you like a helper script that automates switching between ROS and Conda contexts cleanly?

**You said:**

now how to check everthing workig properly

ChatGPT said:

Great! Now let's **fully verify** that your **ROS Noetic + Conda (rosudemy)** hybrid environment is working as expected.

---

## Full Checklist to Confirm Setup

### Part A: ROS Core and GUI tools

#### 1. ROS Environment Check

```
source /opt/ros/noetic/setup.bash
printenv | grep ROS
```

You should see variables like ROS\_VERSION, ROS\_DISTRO, etc.

#### 2. Start roscore

```
roscore
```

You should see:

```
... started core service [/rosout]
```

### 3. Test rqt\_gui

Open another terminal and run:

```
source /opt/ros/noetic/setup.bash
roslaunch rqt_gui rqt_gui
```

It should open the GUI.

---

## Part B: rosudemy Conda Env Check

### 4. Activate and Use Conda + ROS Python

```
conda activate rosudemy
source /opt/ros/noetic/setup.bash
python3
```

Then inside Python:

```
import rospy
print(rospy.__file__) # Should show path from /opt/ros/noetic
```

If this works, ROS Python is accessible inside Conda!

---

## Part C: YOLO + OpenCV Check (Inside Conda)

### 5. Check OpenCV

```
conda activate rosudemy
python3
```

Then:

```
import cv2
print(cv2.__version__)
```

### 6. Test YOLOv5 from ultralytics

```
yolo task=detect mode=predict model=yolov5s.pt source=0
```

This should activate your webcam and run live detection.

---

## Optional: Test turtlesim

```
sudo apt install ros-noetic-turtlesim
roscore
In a new terminal
source /opt/ros/noetic/setup.bash
roslaunch turtlesim turtlesim_node
```

You should see a turtle window!



---

## Everything is Working If:

| Test                         | Expected Outcome                   |
|------------------------------|------------------------------------|
| roscore                      | Starts successfully                |
| rqt_gui                      | GUI opens                          |
| import rospy in Conda        | Works without error                |
| cv2.__version__ in Conda     | Prints OpenCV version              |
| yolo detect from ultralytics | Webcam window with detection boxes |
| turtlesim_node               | Shows animated turtle in GUI       |

---

Would you like me to generate a `check_rosudemy.sh` script that runs all these checks automatically for you?

**You said:**

```
(rosudemy) sakthees-monk@saktheesmonk-desktop:~$ printenv | grep ROS ROS_VERSION=1
ROS_PYTHON_VERSION=3 ROS_PACKAGE_PATH=/opt/ros/noetic/share
ROSLISP_PACKAGE_DIRECTORIES= ROS_ETC_DIR=/opt/ros/noetic/etc/ros
ROS_MASTER_URI=http://localhost:11311 ROS_ROOT=/opt/ros/noetic/share/ros
ROS_DISTRO=noetic (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ roscore ... logging to
/home/sakthees-monk/.ros/log/882518a6-63cf-11f0-9b6f-e93e91f5950d/roslaunch-saktheesmonk-
desktop-18938.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://saktheesmonk-desktop:45869/ ros_comm version 1.17.4 SUMMARY =====
PARAMETERS * /rostdistro: noetic * /rosversion: 1.17.4 NODES RLException: roscore cannot run
as another roscore/master is already running. Please kill other roscore/master processes before
relaunching. The ROS_MASTER_URI is http://saktheesmonk-desktop:11311/ The traceback for the
exception was written to the log file (rosudemy) sakthees-monk@saktheesmonk-desktop:~$
source /opt/ros/noetic/setup.bash (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ rosrn
rqt_gui rqt_gui Traceback (most recent call last): File "/opt/ros/noetic/lib/rqt_gui/rqt_gui", line 13,
in <module> sys.exit(main.main()) File
"/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/main.py", line 61, in main return super(File
"/opt/ros/noetic/lib/python3/dist-packages/qt_gui/main.py", line 406, in main from
python_qt_binding import QT_BINDING File
"/opt/ros/noetic/lib/python3/dist-packages/python_qt_binding/__init__.py", line 55, in <module>
from .binding_helper import loadUi, QT_BINDING, QT_BINDING_MODULES,
QT_BINDING_VERSION # @UnusedImport File
"/opt/ros/noetic/lib/python3/dist-packages/python_qt_binding/binding_helper.py", line 274, in
<module> _select_qt_binding(File
"/opt/ros/noetic/lib/python3/dist-packages/python_qt_binding/binding_helper.py", line 118, in
_select_qt_binding raise ImportError(ImportError: Could not find Qt binding (looked for: 'pyqt',
'pyside'): ImportError for 'pyqt': No module named 'PyQt5' Traceback (most recent call last): File
"/opt/ros/noetic/lib/python3/dist-packages/python_qt_binding/binding_helper.py", line 108, in
_select_qt_binding QT_BINDING_VERSION = binding_loader(required_modules,
```

```
optional_modules) File
"/opt/ros/noetic/lib/python3/dist-packages/python_qt_binding/binding_helper.py", line 153, in
_load_pyqt _named_import('PyQt5.%s' % module_name) File "/opt/ros/noetic/lib/python3/dist-
packages/python_qt_binding/binding_helper.py", line 133, in _named_import module =
builtins.__import__(name) ModuleNotFoundError: No module named 'PyQt5' ImportError for
'pyside': No module named 'PySide2' Traceback (most recent call last): File
"/opt/ros/noetic/lib/python3/dist-packages/python_qt_binding/binding_helper.py", line 108, in
_select_qt_binding QT_BINDING_VERSION = binding_loader(required_modules,
optional_modules) File
"/opt/ros/noetic/lib/python3/dist-packages/python_qt_binding/binding_helper.py", line 185, in
_load_pyside _named_import('PySide2.%s' % module_name) File "/opt/ros/noetic/lib/python3/dist-
packages/python_qt_binding/binding_helper.py", line 133, in _named_import module =
builtins.__import__(name) ModuleNotFoundError: No module named 'PySide2' (rosudemey)
sakthees-monk@saktheesmonk-desktop:~$
```

ChatGPT said:

You're **very close to a complete working ROS + Conda setup!**

Here's exactly what's happening and how to fix it.

---

## Problem Summary

You're running:

```
roslaunch rqt_gui rqt_gui
```

...but getting this:

```
ImportError: Could not find Qt binding (looked for: 'pyqt', 'pyside'):
No module named 'PyQt5'
No module named 'PySide2'
```

This happens because **rqt\_gui** is a **Python GUI tool**, and it can't find a **Qt Python binding (PyQt5 or PySide2)** — because you're inside a **Conda environment**, which overrides the default system Python paths.

---

## Solution

### Option 1: Install PyQt5 and/or PySide2 in Conda

Since you're inside the **rosudemey** Conda environment, install the Qt bindings there:

```
conda activate rosudemey
conda install pyqt -y # For PyQt5
OR
conda install pyside2 -y # For PySide2
```

Then retry:

```
roslaunch rqt_gui rqt_gui
```

---

## Option 2: Use ROS Outside of Conda (Recommended for GUI tools)

ROS tools like `rqt_gui`, `rviz`, `turtlesim`, etc., are **built for system Python**, not Conda. So for these, **deactivate Conda**:

```
conda deactivate
source /opt/ros/noetic/setup.bash
roslaunch rqt_gui rqt_gui
```

This should now launch without any missing PyQt5 errors, since ROS Noetic installs the correct versions system-wide.

---

## Best Practice Recap (Final Setup Strategy)

| Task                                                                 | Where to Run                                                   | Notes                                    |
|----------------------------------------------------------------------|----------------------------------------------------------------|------------------------------------------|
| <code>roscore</code> , <code>roslaunch</code> , <code>rqt_gui</code> | Outside Conda (system shell)                                   | Avoid Conda interference                 |
| YOLOv5, ML/CV                                                        | Inside Conda (rosudemy)                                        | Use <code>conda activate rosudemy</code> |
| Bridge between both                                                  | <code>source /opt/ros/noetic/setup.bash</code><br>inside Conda | Needed for <code>rospy</code> , etc.     |

---

## Want to Fully Test?

Here's a full **system + Conda ROS sanity test**:

```
In terminal 1 (system shell)
conda deactivate
source /opt/ros/noetic/setup.bash
roscore

In terminal 2
conda deactivate
source /opt/ros/noetic/setup.bash
roslaunch rqt_gui rqt_gui

In terminal 3
conda activate rosudemy
source /opt/ros/noetic/setup.bash
python3 -c "import rospy; print('rospy is working in Conda!')"
```

---

Would you like a script that runs all these checks and confirms your setup automatically?

**You said:**

```
roslaunch rqt_gui rqt_gui RosPluginProvider.load(qt_gui_cpp/CppPluginProvider) exception raised in
__builtin__.__import__(qt_gui_cpp.cpp_plugin_provider, [CppPluginProvider]): Traceback (most
recent call last): File "/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/ros_plugin_provider.py",
```

```

line 79, in load module = __builtin__.__import__(File
"/opt/ros/noetic/lib/python3/dist-packages/qt_gui_cpp/cpp_plugin_provider.py", line 33, in
<module> from .cpp_binding_helper import qt_gui_cpp File "/opt/ros/noetic/lib/python3/dist-
packages/qt_gui_cpp/cpp_binding_helper.py", line 43, in <module> from . import
libqt_gui_cpp_sip ValueError: PyCapsule_GetPointer called with incorrect name
RecursivePluginProvider.discover() loading plugin "qt_gui_cpp/CppPluginProvider" failed:
Traceback (most recent call last): File
"/opt/ros/noetic/lib/python3/dist-packages/qt_gui/recursive_plugin_provider.py", line 60, in
discover instance = self._plugin_provider.load(plugin_descriptor.plugin_id(), None) File
"/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/ros_plugin_provider.py", line 90, in load raise e
File "/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/ros_plugin_provider.py", line 79, in load
module = __builtin__.__import__(File
"/opt/ros/noetic/lib/python3/dist-packages/qt_gui_cpp/cpp_plugin_provider.py", line 33, in
<module> from .cpp_binding_helper import qt_gui_cpp File "/opt/ros/noetic/lib/python3/dist-
packages/qt_gui_cpp/cpp_binding_helper.py", line 43, in <module> from . import
libqt_gui_cpp_sip ValueError: PyCapsule_GetPointer called with incorrect name (rosudemey)
sakthees-monk@saktheesmonk-desktop:~$ error

```

ChatGPT said:

The error you're seeing:

`ValueError: PyCapsule_GetPointer called with incorrect name`

...comes from a mismatch between **Python versions** and **compiled C++ Qt bindings (SIP bindings)** used by `rqt_gui` in ROS Noetic.

This is **very common** when you're running ROS Noetic inside a custom **Conda environment** (like your `(rosudemey)` one), because ROS is built to work with the **system's default Python 3.8 (on Ubuntu 20.04), not Conda's Python interpreter**.

## Here's how to fix this:

### Option 1: Run `rqt_gui` outside conda (recommended for GUI)

ROS tools like `rqt_gui`, `rviz`, etc., expect the system's Python and Qt setup.

```
First, deactivate your conda env:
conda deactivate
```

```
Then run:
roslaunch rqt_gui rqt_gui
```

If it works now, the issue was **conda conflicting with ROS's Qt/Python bindings** — which it almost always does with GUI tools.

### Option 2: If you *must* use Conda, set environment variables carefully (not always stable)

You need to make sure that the Conda environment does **not override** system Qt, SIP, and Python bindings used by ROS.

Before running `rqt_gui`, set the following in your Conda environment:

```
export PYTHONPATH=/opt/ros/noetic/lib/python3/dist-packages:$PYTHONPATH
export LD_LIBRARY_PATH=/opt/ros/noetic/lib:$LD_LIBRARY_PATH
export QT_QPA_PLATFORM_PLUGIN_PATH=/opt/ros/noetic/lib/qt5/plugins
```

Then run:

```
roslaunch rqt_gui rqt_gui
```

Still, this may not fix it completely due to deeper SIP mismatch between Conda Python and ROS's system SIP modules.

---

## Best Practice

- **Use Conda only for ML/CV tasks** (like YOLO training, OpenCV, etc.)
  - **Use system Python for ROS GUI tools** (`rqt_gui`, `rviz`, etc.)
  - You can have both worlds by launching ROS core/system tools from a regular terminal and running CV models inside the conda env with ROS bindings (if needed).
- 

## To Check if ROS Itself is Working Fine (without GUI)

Use these simple checks:

```
Open one terminal
roscore
```

In another terminal:

```
roslaunch turtlesim turtlesim_node
```

Then:

```
roslaunch turtlesim turtle_teleop_key
```

If the turtle moves with your keyboard, **ROS core works fine!**

Let me know if you'd like help testing YOLO or ROS-OpenCV nodes next.

**You said:**

```
First, deactivate your conda env: (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ conda
deactivate (base) sakthees-monk@saktheesmonk-desktop:~$ (base) sakthees-
monk@saktheesmonk-desktop:~$ # Then run: (base) sakthees-monk@saktheesmonk-desktop:~$
roslaunch rqt_gui rqt_gui Traceback (most recent call last): File "/opt/ros/noetic/lib/rqt_gui/rqt_gui",
line 6, in <module> import rospkg ModuleNotFoundError: No module named 'rospkg' (base)
sakthees-monk@saktheesmonk-desktop:~$
```

ChatGPT said:

You are encountering the `ModuleNotFoundError: No module named 'rospkg'` even outside your Conda environment, which means your **global Python environment is missing**

**ROS Python packages**—likely because of conflicts introduced by Conda or missing environment sourcing.

---

## Let's fix this step by step:

### Step 1: Source ROS setup properly

Ensure your shell sources ROS Noetic on every new terminal.

```
source /opt/ros/noetic/setup.bash
```

You can also add this to your `~/.bashrc`:

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

---

### Step 2: Reinstall missing Python packages for ROS

You need to install `rospkg` and other dependencies in the **system Python** (not Conda).

Run:

```
sudo apt install python3-rospkg python3-rospkg-modules python3-catkin-pkg-modules python3-catkin-pkg
```

---

### Step 3: Try running `rqt_gui` again

Now, try:

```
roslaunch rqt_gui rqt_gui
```

---

## Optional: Test if ROS is working properly

You can test a simple publisher-subscriber communication:

```
roscore
```

In a new terminal:

```
source /opt/ros/noetic/setup.bash
roslaunch turtlesim turtlesim_node
```

In another terminal:

```
source /opt/ros/noetic/setup.bash
roslaunch turtlesim turtle_teleop_key
```

If the turtle moves, your ROS installation works.

---

Let me know if `rqt_gui` still throws the `libqt_gui_cpp_sip` error—we can then rebuild that package from source if needed.

**You said:**

```
base) sakthees-monk@saktheesmonk-desktop:~$ conda activate rosudemy (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ rosrn rqt_gui rqt_gui
RosPluginProvider.load(qt_gui_cpp/CppPluginProvider) exception raised in
__builtin__.__import__(qt_gui_cpp.cpp_plugin_provider, [CppPluginProvider]): Traceback (most recent call last): File "/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/ros_plugin_provider.py", line 79, in load module = __builtin__.__import__(File "/opt/ros/noetic/lib/python3/dist-packages/qt_gui_cpp/cpp_plugin_provider.py", line 33, in <module> from .cpp_binding_helper import qt_gui_cpp File "/opt/ros/noetic/lib/python3/dist-packages/qt_gui_cpp/cpp_binding_helper.py", line 43, in <module> from . import libqt_gui_cpp_sip ValueError: PyCapsule_GetPointer called with incorrect name RecursivePluginProvider.discover() loading plugin "qt_gui_cpp/CppPluginProvider" failed: Traceback (most recent call last): File "/opt/ros/noetic/lib/python3/dist-packages/qt_gui/recursive_plugin_provider.py", line 60, in discover instance = self._plugin_provider.load(plugin_descriptor.plugin_id(), None) File "/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/ros_plugin_provider.py", line 90, in load raise e File "/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/ros_plugin_provider.py", line 79, in load module = __builtin__.__import__(File "/opt/ros/noetic/lib/python3/dist-packages/qt_gui_cpp/cpp_plugin_provider.py", line 33, in <module> from .cpp_binding_helper import qt_gui_cpp File "/opt/ros/noetic/lib/python3/dist-packages/qt_gui_cpp/cpp_binding_helper.py", line 43, in <module> from . import libqt_gui_cpp_sip ValueError: PyCapsule_GetPointer called with incorrect name (rosudemy) sakthees-monk@saktheesmonk-desktop:~$
```

**ChatGPT said:**

You're running into a common compatibility issue when using **ROS Noetic** with **Python virtual environments** like Conda, especially when launching `rqt_gui`. Here's a breakdown of the problem and a step-by-step fix:

---

## Problem Summary

- ROS Noetic uses **Python 3.8** (on Ubuntu 20.04).
- `rqt_gui` uses compiled C++-Python bindings (`libqt_gui_cpp_sip`) which are **very sensitive to Python environment mismatches**.
- When you use `conda activate`, you're switching to a **different Python interpreter**, which **breaks the bindings**.

---

## Error Breakdown

ValueError: PyCapsule\_GetPointer called with incorrect name

This error indicates that the compiled Python module (`libqt_gui_cpp_sip`) was built for the system Python, **not** for the one in your Conda environment.

---

## Recommended Fix

**Never run GUI-based ROS tools like `rqt_gui`, `rviz`, etc., from inside a Conda environment.**

### Steps to Reset and Run `rqt_gui` Properly:

#### 1. Exit conda:

- `conda deactivate`

#### • Ensure you're using system Python:

- `which python3`

# Output should be: `/usr/bin/python3`

#### • Source ROS environment:

- `source /opt/ros/noetic/setup.bash`

#### • Then run:

4. `roslaunch rqt_gui rqt_gui`

---

## Check Everything Works

To verify ROS Noetic is properly installed and working:

### 1. Check `roscore`:

`roscore`

You should see: `started core service [/rosout]`

### 2. In a new terminal:

```
source /opt/ros/noetic/setup.bash
roslaunch turtlesim turtlesim_node
```

### 3. In another terminal:

```
source /opt/ros/noetic/setup.bash
roslaunch turtlesim turtle_teleop_key
```

→ You should be able to **control the turtle** with arrow keys.

---



## Summary

- `rqt_gui` fails inside Conda because ROS was built with system Python.
- Run all GUI tools **outside** Conda.
- Use Conda only for custom scripts, ML models, or packages that don't rely on ROS GUI binaries.

Let me know if you want to set up a **ROS+Conda workflow** where only custom Python scripts run inside Conda, but the rest use system ROS.

### You said:

```
(rosudemey) sakthees-monk@saktheesmonk-desktop:~$ which python3
/home/sakthees-monk/Downloads/yes/envs/rosudemey/bin/python3 (rosudemey) sakthees-
monk@saktheesmonk-desktop:~$ # Output should be: /usr/bin/python3 (rosudemey) sakthees-
monk@saktheesmonk-desktop:~$ source /opt/ros/noetic/setup.bash (rosudemey) sakthees-
monk@saktheesmonk-desktop:~$ rosrun rqt_gui rqt_gui
RosPluginProvider.load(qt_gui_cpp/CppPluginProvider) exception raised in
__builtin__.__import__(qt_gui_cpp.cpp_plugin_provider, [CppPluginProvider]): Traceback (most
recent call last): File "/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/ros_plugin_provider.py",
line 79, in load module = __builtin__.__import__(File
"/opt/ros/noetic/lib/python3/dist-packages/qt_gui_cpp/cpp_plugin_provider.py", line 33, in
<module> from .cpp_binding_helper import qt_gui_cpp File "/opt/ros/noetic/lib/python3/dist-
packages/qt_gui_cpp/cpp_binding_helper.py", line 43, in <module> from . import
libqt_gui_cpp_sip ValueError: PyCapsule_GetPointer called with incorrect name
RecursivePluginProvider.discover() loading plugin "qt_gui_cpp/CppPluginProvider" failed:
Traceback (most recent call last): File
"/opt/ros/noetic/lib/python3/dist-packages/qt_gui/recursive_plugin_provider.py", line 60, in
discover instance = self._plugin_provider.load(plugin_descriptor.plugin_id(), None) File
"/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/ros_plugin_provider.py", line 90, in load raise e
File "/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/ros_plugin_provider.py", line 79, in load
module = __builtin__.__import__(File
"/opt/ros/noetic/lib/python3/dist-packages/qt_gui_cpp/cpp_plugin_provider.py", line 33, in
<module> from .cpp_binding_helper import qt_gui_cpp File "/opt/ros/noetic/lib/python3/dist-
packages/qt_gui_cpp/cpp_binding_helper.py", line 43, in <module> from . import
libqt_gui_cpp_sip ValueError: PyCapsule_GetPointer called with incorrect name (rosudemey)
sakthees-monk@saktheesmonk-desktop:~$ roscore ... logging to
/home/sakthees-monk/.ros/log/882518a6-63cf-11f0-9b6f-e93e91f5950d/roslaunch-saktheesmonk-
desktop-19650.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://saktheesmonk-desktop:39969/ ros_comm version 1.17.4 SUMMARY =====
PARAMETERS * /rostdistro: noetic * /rosversion: 1.17.4 NODES RLException: roscore cannot run
as another roscore/master is already running. Please kill other roscore/master processes before
relaunching. The ROS_MASTER_URI is http://saktheesmonk-desktop:11311/ The traceback for the
exception was written to the log file give pip of ros dep
```

ChatGPT said:

You're very close to a working ROS Noetic setup inside your `rosudemy` Conda environment. Let's handle your questions and errors one by one:

---

## You've done correctly:

- Created a Conda env `rosudemy`.
  - ROS is sourced via `source /opt/ros/noetic/setup.bash`.
  - `roscore` started properly (though it says another instance is running).
- 

## Error in `roslaunch rqt_gui rqt_gui`

### Error:

```
ValueError: PyCapsule_GetPointer called with incorrect name
```

### Cause:

This error usually occurs when ROS tools like `rqt_gui` are run from a Conda environment with a **different Python version** or **incompatible SIP bindings** than what ROS expects.

ROS Noetic expects **Python 3.8** (the system Python on Ubuntu 20.04), and specifically **not** a Conda Python or custom SIP install.

---

## Your current Python is:

```
which python3
/home/sakthees-monk/Downloads/yes/envs/rosudemy/bin/python3
```

This is Conda's Python, **not** system Python (`/usr/bin/python3`). That is the issue.

---

## Fix Options:

### Option 1: Run ROS tools *outside* Conda

Just use the terminal outside your Conda environment:

```
conda deactivate
source /opt/ros/noetic/setup.bash
roslaunch rqt_gui rqt_gui
```

**Why?:** Tools like `rqt_gui` and `rviz` must use system Python and native bindings, not Conda.

---

## Option 2: Keep using Conda for code, not for GUI tools

Use `rosudemmy` environment **only for your scripts**, training, or YOLO pipelines, not for `rqt_gui`, `rviz`, etc.

---

## Installing `rosdep` using `pip` in Conda

If you're asking how to install `rosdep` inside Conda:

```
pip install rosdep
```

But it's better to install it system-wide using:

```
sudo apt install python3-rosdep
```

Then initialize once:

```
sudo rosdep init
rosdep update
```

---

## Check that everything works:

### 1. Check ROS setup:

```
source /opt/ros/noetic/setup.bash
roscore
```

### 2. New terminal: Run a basic talker/listener test:

```
roslaunch rospym_tutorials talker
```

Another terminal:

```
roslaunch rospym_tutorials listener
```

### 3. YOLO or your training pipeline: Run it from Conda `rosudemmy` env — works best for training.

---

## Summary:

| Task                                                            | Run in Conda? | Run Outside Conda? |
|-----------------------------------------------------------------|---------------|--------------------|
| <code>roscore</code> , <code>rqt_gui</code> , <code>rviz</code> | No            | Yes                |
| Python training (YOLO, etc)                                     | Yes           | No                 |
| ROS <code>rosdep</code> setup                                   | No (system)   | Yes                |

---

If you want help verifying your `rosdep` is working or need to test YOLO inside the ROS node, I can guide you further.

### You said:

Skip to content Chat history You said: triggers for man-db (2.9.1-1) ... (yoloven) (base) sakthees-monk@saktheesmonk-desktop:~\$ sudo apt update Hit:2 http://in.archive.ubuntu.com/ubuntu focal InRelease Get:3 http://packages.ros.org/ros/ubuntu focal InRelease [4,679 B] Hit:4 http://security.ubuntu.com/ubuntu focal-security InRelease Get:5 http://in.archive.ubuntu.com/ubuntu focal-updates InRelease [128 kB] Err:3 http://packages.ros.org/ros/ubuntu focal InRelease The following signatures couldn't be verified because the public key is not available: NO\_PUBKEY F42ED6FBAB17C654 Hit:6 http://in.archive.ubuntu.com/ubuntu focal-backports InRelease Hit:7 http://in.archive.ubuntu.com/ubuntu focal-proposed InRelease Get:1 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86\_64 InRelease [1,581 B] Err:1 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86\_64 InRelease The following signatures couldn't be verified because the public key is not available: NO\_PUBKEY A4B469963BF863CC Reading package lists... Done W: GPG error: http://packages.ros.org/ros/ubuntu focal InRelease: The following signatures couldn't be verified because the public key is not available: NO\_PUBKEY F42ED6FBAB17C654 E: The repository 'http://packages.ros.org/ros/ubuntu focal InRelease' is not signed. N: Updating from such a repository can't be done securely, and is therefore disabled by default. N: See apt-secure(8) manpage for repository creation and user configuration details. W: GPG error: https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86\_64 InRelease: The following signatures couldn't be verified because the public key is not available: NO\_PUBKEY A4B469963BF863CC E: The repository 'http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86\_64 InRelease' is not signed. N: Updating from such a repository can't be done securely, and is therefore disabled by default. N: See apt-secure(8) manpage for repository creation and user configuration details. (yoloven) (base) sakthees-monk@saktheesmonk-desktop:~\$ sudo apt install ros-noetic-desktop-full Reading package lists... Done Building dependency tree Reading state information... Done E: Unable to locate package ros-noetic-desktop-full (yoloven) (base) sakthees-monk@saktheesmonk-desktop:~\$ ChatGPT said: You're facing two main issues here: Issue 1: Missing GPG keys This is causing the following repositories to be considered untrusted: http://packages.ros.org/ros/ubuntu https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86\_64 You need to add the missing public keys to fix this. Fix for ROS (missing key F42ED6FBAB17C654) sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key F42ED6FBAB17C654 Fix for CUDA (missing key A4B469963BF863CC) sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key A4B469963BF863CC Alternatively, using curl (sometimes more reliable): # ROS key curl -sSL 'http://keyserver.ubuntu.com/pks/lookup?op=get&search=0xF42ED6FBAB17C654' | sudo apt-key add - # CUDA key curl -sSL 'http://keyserver.ubuntu.com/pks/lookup?op=get&search=0xA4B469963BF863CC' | sudo apt-key add - Then run: sudo apt update Issue 2: ros-noetic-desktop-full not found This happens usually because: The ROS repository isn't set up correctly, or apt update failed earlier due to the GPG key error. Full Fix: ROS Noetic Installation for Ubuntu 20.04 (Focal) Follow these steps after fixing the keys: 1. Add the ROS repository: sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu focal main" > /etc/apt/sources.list.d/ros1-latest.list' 2. Add the key again if not already: sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key F42ED6FBAB17C654 3. Update package index: sudo apt update 4. Install ROS Noetic Desktop Full: sudo apt install ros-noetic-desktop-full Pro Tip: If you're using Ubuntu 22.04 or newer, ros-noetic will not work — you'll need

ROS 2 (like Humble or Iron). Noetic is only officially supported on Ubuntu 20.04. Would you like a full ROS Noetic setup script (including rosdep, catkin, etc.)? You said: nk@saktheesmonk-desktop:~\$ sudo apt install ros-noetic-PACKAGE Reading package lists... Done Building dependency tree Reading state information... Done E: Unable to locate package ros-noetic-PACKAGE ChatGPT said: The error: E: Unable to locate package ros-noetic-PACKAGE is because PACKAGE is just a placeholder, not an actual ROS package name.      Correct Usage If you want to install a specific ROS Noetic package, replace PACKAGE with a real package name, like: Example: sudo apt install ros-noetic-turtlesim Or, if you want everything, install: sudo apt install ros-noetic-desktop-full      How to find available ROS Noetic packages: You can search for available packages like this: apt-cache search ros-noetic This will list all available ROS Noetic packages in the repository.      Full ROS Noetic Setup (if not done yet) Add the ROS repo (if not already done): sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu focal main" > /etc/apt/sources.list.d/ros1-latest.list' Add the ROS GPG key: sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key F42ED6FBAB17C654 Update the package list: sudo apt update Install desktop-full version: sudo apt install ros-noetic-desktop-full Initialize rosdep: sudo rosdep init rosdep update Add source to your shell: echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc source ~/.bashrc Let me know if you want help installing a specific package or doing a workspace (catkin) setup. You said: sudo apt install python3-rosdep2 give cionda version ChatGPT said: To install rosdep for ROS Noetic (Python 3 version) with Conda compatibility, you need to avoid conflicts between system Python (used by ROS) and Conda Python.      Problem with Your Command: sudo apt install python3-rosdep2 This tries to install a system package named python3-rosdep2, which doesn't exist. The correct system package name is: sudo apt install python3-rosdep But this installs it into the system Python, not Conda.      If You're Using Conda: If you want rosdep inside a Conda environment, follow this method: Step 1: Activate your conda environment conda activate yoloenv # or whatever your env is Step 2: Install dependencies via pip (inside conda) pip install rosdep But note: rosdep needs system tools (apt, sudo, etc.) and root access — even when installed in Conda.      Best Practice (Hybrid Approach): ROS tooling is designed around system Python. Mixing with Conda can break things. So, recommended: Use system Python for all ROS-related tools like rosdep, rosinstall, catkin\_tools, etc. Use Conda for ML/CV libraries like PyTorch, OpenCV, YOLO, etc. You can bridge them via ROS Python APIs or messaging (rospy, cv\_bridge, etc.).      Summary of Fixes: Goal Command Install rosdep system-wide sudo apt install python3-rosdep Initialize rosdep sudo rosdep init && rosdep update Install in Conda (optional) pip install rosdep (but not recommended for full ROS setups) Search package apt-cache search rosdep Let me know your goal: Do you want ROS fully inside Conda? Or just use Conda for some parts and ROS system-wide? I can guide setup accordingly. You said:      If You're Using Conda: If you want rosdep inside a Conda environment, follow this method: Step 1: Activate your conda environment conda activate yoloenv # or whatever your env is Step 2: Install dependencies via pip (inside conda) pip install rosdep But note: rosdep needs system tools (apt, sudo, etc.) and root access — even when installed in Conda.      Best Practice (Hybrid Approach): ROS tooling is designed around system Python. Mixing with Conda can break things. So, recommended: Use system Python for all ROS-related tools like rosdep, rosinstall, catkin\_tools, etc. Use Conda for ML/CV libraries like PyTorch, OpenCV, YOLO, etc. You can bridge them via ROS Python APIs or messaging (rospy, cv\_bridge, etc.).      Summary of Fixes: Goal Command Install rosdep system-wide sudo apt install python3-rosdep Initialize rosdep sudo rosdep init && rosdep update Install in Conda (optional) pip install rosdep (but not recommended for full ROS setups) Search package apt-cache search rosdep Let me know your goal: Do you want ROS fully inside Conda? Or just use Conda for

some parts and ROS system-wide? I can guide setup accordingly. create env initialise ChatGPT said: Great! Since you're asking to create the environment and initialize it, I'll assume you want to: Use ROS Noetic and its core tools system-wide (as recommended), Use Conda for machine learning, OpenCV, and YOLO work, Connect both environments smoothly (hybrid setup).

Step-by-Step: Create & Initialize Hybrid Conda + ROS Environment

Step 1: Create the Conda Environment (Python 3.8 — compatible with ROS Noetic) `conda create -n yolovenv python=3.8 -y`  
`conda activate yolovenv` ROS Noetic works best with Python 3.8.

Step 2: Install ML/CV Libraries inside Conda You can install these inside the yolovenv: # PyTorch + OpenCV + YOLO + other tools `conda install pytorch torchvision torchaudio -c pytorch` `conda install opencv` `pip install ultralytics` # for YOLOv8+

Step 3: Install ROS-related tools system-wide (do NOT install them in conda)

3.1 Install rosdep: `sudo apt update` `sudo apt install python3-rosdep`

3.2 Initialize rosdep (first-time only): `sudo rosdep init` `rosdep update`

Step 4: Bridge Conda & ROS Now add ROS setup to your shell (optional): `echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc` `source ~/.bashrc` Then, whenever needed: # Activate Conda for ML work `conda activate yolovenv` # Source ROS separately if not already in bashrc `source /opt/ros/noetic/setup.bash` Now you can run rospy, roslaunch, etc., and also import ML models inside the same terminal session.

You're Ready! To test: Run: `roscore` In another terminal: `conda activate yolovenv` `source /opt/ros/noetic/setup.bash` `python3 >>> import rospy >>> print("ROS & Conda working together!")` Would you like a shell script to automate all these steps for future machines or VMs? You said: give rosudemy for yolovenv env name \

ChatGPT said: Certainly! Here's a shell script that will:

Create a Conda environment named rosudemy

Install Python 3.8 (compatible with ROS Noetic)

Install ML/CV tools inside Conda (PyTorch, OpenCV, YOLO)

Set up ROS Noetic system-wide (not in Conda)

Bridge ROS with your Conda environment

Shell Script: `setup_rosudemy.sh`

```
#!/bin/bash
echo "Creating Conda environment: rosudemy (Python 3.8)..."
conda create -n rosudemy python=3.8 -y
echo "Activating rosudemy..."
conda activate rosudemy
echo "Installing ML and CV libraries into rosudemy..."
conda install pytorch torchvision torchaudio -c pytorch -y
conda install opencv -y
pip install ultralytics
echo "Installing rosdep system-wide (Ubuntu)..."
sudo apt update
sudo apt install python3-rosdep -y
echo "Initializing rosdep (if first time)..."
sudo rosdep init 2>/dev/null || echo "rosdep already initialized"
rosdep update
echo "Sourcing ROS Noetic in ~/.bashrc if not already there..."
grep -qxF 'source /opt/ros/noetic/setup.bash' ~/.bashrc || echo 'source /opt/ros/noetic/setup.bash' >> ~/.bashrc
source ~/.bashrc
echo "Setup complete!"
echo -e "\n Use this in your terminal when working:\n"
echo "conda activate rosudemy"
echo "source /opt/ros/noetic/setup.bash"
```

How to Use: Save it: `nano setup_rosudemy.sh` # Paste the code and save (Ctrl+O, Enter, Ctrl+X) Make it executable: `chmod +x setup_rosudemy.sh` Run it: `./setup_rosudemy.sh` Would you like me to add optional support for Jupyter, matplotlib, or catkin workspace setup inside the rosudemy environment as well? You said: (base) sakthees-monk@saktheesmonk-desktop:~\$ `conda create -n rosudemy python=3.8 -y` CondaToSNonInteractiveError: Terms of Service have not been accepted for the following channels. Please accept or remove them before proceeding: • <https://repo.anaconda.com/pkgs/main> • <https://repo.anaconda.com/pkgs/r> To accept a channel's Terms of Service, run the following and replace CHANNEL with the channel name/URL: ▶ `conda tos accept --override-channels --channel CHANNEL` To remove channels with rejected Terms of Service, run the following and replace CHANNEL with the channel name/URL: ▶ `conda config --remove channels CHANNEL` (base) sakthees-monk@saktheesmonk-desktop:~\$

ChatGPT said: You're seeing this error because Conda now requires explicit acceptance of the Terms of Service (ToS) for some official channels (like <https://repo.anaconda.com/pkgs/main>). To Fix the Error

You need to manually accept the ToS for the required channels. Run the following commands exactly: `conda tos accept --override-channels --channel https://repo.anaconda.com/pkgs/main` `conda tos accept --override-channels --channel https://repo.anaconda.com/pkgs/r` Then retry: `conda create -n rosudemy python=3.8 -y` Optional (View all ToS-required channels): `conda tos list` This shows channels that still need your approval. Let me know if you want to install from only community channels (like conda-forge) and avoid Anaconda official ones. You said: usage: `conda tos [-V] [-c CHANNEL] [--override-channels] [-n ENVIRONMENT | -p PATH] [--site | --system | --user | --env | --tos-root TOS_ROOT] [--cache-timeout CACHE_TIMEOUT | --ignore-cache] [--json] [-v] [-h] {accept,reject,view,interactive,info,clean} ...` `conda tos: error: argument cmd: invalid choice: 'list' (choose from accept, reject, view, interactive, info, clean)` ChatGPT said: You're right — `conda tos` does not have a list command. That was a mistake on my part. Thanks for catching it.

To proceed, just do this: Run these two commands exactly to accept the required channels: `conda tos accept --override-channels --channel https://repo.anaconda.com/pkgs/main` `conda tos accept --override-channels --channel https://repo.anaconda.com/pkgs/r` Then try: `conda create -n rosudemy python=3.8 -y` Optional: View accepted ToS (for verification) You can use the following to view the current status: `conda tos info` It will show which channels' Terms of Service have been accepted or rejected. Let me know if you'd prefer to switch to only using conda-forge (no Anaconda ToS required). You said: `#!/bin/bash echo " Creating Conda environment: rosudemy (Python 3.8)..." conda create -n rosudemy python=3.8 -y echo " Activating rosudemy..." conda activate rosudemy echo " Installing ML and CV libraries into rosudemy..." conda install pytorch torchvision torchaudio -c pytorch -y conda install opencv -y pip install ultralytics echo " Installing rosdep system-wide (Ubuntu)..." sudo apt update sudo apt install python3-rosdep -y echo " Initializing rosdep (if first time)..." sudo rosdep init 2>/dev/null || echo "rosdep already initialized" rosdep update echo " Sourcing ROS Noetic in ~/.bashrc if not already there..." grep -qxF 'source /opt/ros/noetic/setup.bash' ~/.bashrc || echo 'source /opt/ros/noetic/setup.bash' >> ~/.bashrc source ~/.bashrc echo " Setup complete!" echo -e "\n Use this in your terminal when working:\n" echo "conda activate rosudemy" echo "source /opt/ros/noetic/setup.bash" for yolov5 yolov11 create` ChatGPT said: Thanks! You're asking to modify the script for YOLOv5 and YOLOv11 inside the same rosudemy Conda environment. Since YOLOv11 is unofficial (likely just a future/forked version), the goal is: Install everything required for YOLOv5 (Ultralytics repo, PyTorch, OpenCV, etc.) Make the setup extensible for future YOLO versions like YOLOv11 (assuming you'll clone/run it manually) Updated Script for YOLOv5 + Future YOLO (e.g., YOLOv11) `#!/bin/bash echo " Creating Conda environment: rosudemy (Python 3.8)..." conda create -n rosudemy python=3.8 -y echo " Activating rosudemy..." # Use conda shell integration to activate env in script eval "$(conda shell.bash hook)" conda activate rosudemy echo " Installing ML and CV libraries into rosudemy..." conda install pytorch torchvision torchaudio -c pytorch -y conda install opencv -y echo " Installing YOLOv5 tools (Ultralytics)..." pip install ultralytics # Includes YOLOv5 CLI echo " Cloning YOLOv5 for custom usage..." git clone https://github.com/ultralytics/yolov5.git cd yolov5 pip install -r requirements.txt cd .. echo " (Optional) Clone YOLOv11 manually if available/forked." echo " Installing rosdep system-wide (Ubuntu)..." sudo apt update sudo apt install python3-rosdep -y echo " Initializing rosdep (if first time)..." sudo rosdep init 2>/dev/null || echo "rosdep already initialized" rosdep update echo " Sourcing ROS Noetic in ~/.bashrc if not already there..." grep -qxF 'source /opt/ros/noetic/setup.bash' ~/.bashrc || echo 'source /opt/ros/noetic/setup.bash' >> ~/.bashrc source ~/.bashrc echo " Setup complete!" echo -e "\n Use this in your terminal when working:\n" echo "conda activate rosudemy" echo "source /opt/ros/noetic/setup.bash" echo "cd yolov5 &&`

python detect.py --source 0 # (for webcam test)" What this sets up: YOLOv5 via ultralytics pip module + full repo Ready to clone and run future versions like YOLOv11 System-wide ROS tools (Noetic + rosdep) Conda Python 3.8 for model compatibility Easy terminal re-entry setup Let me know if you also want: Automatic YOLOv11 fork from GitHub if available Jupyter Notebook setup in rosudemy cv\_bridge installation between OpenCV and ROS You said: 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" Skip end-of-life distro "foxy" Skip end-of-life distro "galactic" Skip end-of-life distro "groovy" Add distro "humble" Skip end-of-life distro "hydro" Skip end-of-life distro "indigo" Skip end-of-life distro "iron" Skip end-of-life distro "jade" Add distro "jazzy" Add distro "kilted" Skip end-of-life distro "kinetic" Skip end-of-life distro "lunar" Skip end-of-life distro "melodic" Skip end-of-life distro "noetic" Add distro "rolling" which to add noetic install how to check ros properly with desktop full packages installed ChatGPT said: You're asking two things:

1. You ran rosdep update and saw: Skip end-of-life distro "noetic" This is normal — ROS considers Noetic end-of-life upstream (May 2025), so rosdep doesn't actively pull dependency updates for it anymore, but it still works if already installed. If your system already has Noetic installed, it's fine to continue using it.
2. How to check if ROS Noetic Desktop Full is installed correctly? Run the following checks one by one:
  - A. Check installation directory: ls /opt/ros/noetic You should see folders like bin, lib, share, setup.bash, etc.
  - B. Check if core ROS command works: roscore You should see roscore running with logs like: ... started core service [/rosout]
  - C. Check if some key packages exist (e.g., rviz, rqt, etc.) rosrun rviz rviz rosrun rqt\_gui rqt\_gui If you see GUI windows opening, desktop packages are installed.
  - D. Check with dpkg if full package is installed: dpkg -l | grep ros-noetic-desktop-full If installed, you'll see output like: ii ros-noetic-desktop-full 1.15.11-1focal all ROS Desktop Full install
  - E. Confirm with rosversion: rosversion -d Should return: noetic
  - F. Run a test package: Try this: sudo apt install ros-noetic-turtlesim rosrun turtlesim turtlesim\_node You should see a turtle simulation window.

If ros-noetic-desktop-full was not installed: Run: sudo apt update sudo apt install ros-noetic-desktop-full Let me know if you want a command to list all installed ROS Noetic packages, or verify catkin/roslaunch/rospy etc. You said: osrun rqt\_gui rqt\_gui Traceback (most recent call last): File "/opt/ros/noetic/lib/rqt\_gui/rqt\_gui", line 6, in <module> import rospkg ModuleNotFoundError: No module named 'rospkg' "ModuleNotFoundError: No module named 'rospkg' (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ dpkg -l | grep ros-noetic-desktop-full ii ros-noetic-desktop-full 1.5.0-1focal.20250521.014741 amd64 A metapackage to aggregate several packages. (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ ChatGPT said: Great — you've confirmed that: ros-noetic-desktop-full is installed (via dpkg) But you're seeing an error: ModuleNotFoundError: No module named 'rospkg' This means that when running rosrun rqt\_gui rqt\_gui, it's using your Conda Python, which doesn't have the ROS Python packages. Why This Happens You're inside the rosudemy Conda environment: (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ So when you run a script, it's using Conda's Python (e.g., /home/.../miniconda3/envs/rosudemy/bin/python) — which doesn't have rospkg, rospy, etc.



Solution: Use System Python for ROS commands To properly run rosrn, rqt, roscore, etc., you should exit Conda, or avoid letting Conda override PYTHONPATH. Option 1: Deactivate Conda before running ROS conda deactivate rosrn rqt\_gui rqt\_gui This will use the system Python from /opt/ros/noetic, where rospkg and other ROS libraries are properly installed. Option 2: Add system Python to rosudemy (not recommended, but possible) If you absolutely want to run ROS code inside Conda (not just interact with it), you'll need to install ROS Python packages into Conda (but mixing can break things): pip install rospkg catkin\_pkg roscpp roscpp\_pywrap roscpp\_pywrap\_generator wstool But even with this, GUI tools like rqt or rviz are compiled C++ binaries, and still rely on /opt/ros/noetic. Quick Test: To confirm you're using system Python, run: which python Expected output when not in Conda: /usr/bin/python3 Expected output in Conda: /home/youruser/miniconda3/envs/rosudemy/bin/python Best Practice Recap: Task Environment How to run roscore, rosrn, rqt\_gui System shell conda deactivate, then run YOLO, ML models rosudemy Conda env conda activate rosudemy Use both System + Conda combo Source ROS after activating Conda conda activate rosudemy source /opt/ros/noetic/setup.bash python3 # use rospkg, etc. inside Conda Would you like a helper script that automates switching between ROS and Conda contexts cleanly? You said: now how to check everthing workig properly ChatGPT said: Great! Now let's fully verify that your ROS Noetic + Conda (rosudemy) hybrid environment is working as expected. Full Checklist to Confirm Setup Part A: ROS Core and GUI tools 1. ROS Environment Check source /opt/ros/noetic/setup.bash printenv | grep ROS You should see variables like ROS\_VERSION, ROS\_DISTRO, etc. 2. Start roscore roscore You should see: ... started core service [/rosout] 3. Test rqt\_gui Open another terminal and run: source /opt/ros/noetic/setup.bash rosrn rqt\_gui rqt\_gui It should open the GUI. Part B: rosudemy Conda Env Check 4. Activate and Use Conda + ROS Python conda activate rosudemy source /opt/ros/noetic/setup.bash python3 Then inside Python: import rospkg print(rospkg.\_\_file\_\_) # Should show path from /opt/ros/noetic If this works, ROS Python is accessible inside Conda! Part C: YOLO + OpenCV Check (Inside Conda) 5. Check OpenCV conda activate rosudemy python3 Then: import cv2 print(cv2.\_\_version\_\_) 6. Test YOLOv5 from ultralytics yolo task=detect mode=predict model=yolov5s.pt source=0 This should activate your webcam and run live detection. Optional: Test turtlesim sudo apt install ros-noetic-turtlesim roscore # In a new terminal source /opt/ros/noetic/setup.bash rosrn turtlesim turtlesim\_node You should see a turtle window! Everything is Working If: Test Expected Outcome roscore Starts successfully rqt\_gui GUI opens import rospkg in Conda Works without error cv2.\_\_version\_\_ in Conda Prints OpenCV version yolo detect from ultralytics Webcam window with detection boxes turtlesim\_node Shows animated turtle in GUI Would you like me to generate a check\_rosudemy.sh script that runs all these checks automatically for you? You said: (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ printenv | grep ROS ROS\_VERSION=1 ROS\_PYTHON\_VERSION=3 ROS\_PACKAGE\_PATH=/opt/ros/noetic/share ROSLISP\_PACKAGE\_DIRECTORIES= ROS\_ETC\_DIR=/opt/ros/noetic/etc/ros ROS\_MASTER\_URI=http://localhost:11311 ROS\_ROOT=/opt/ros/noetic/share/ros ROS\_DISTRO=noetic (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ roscore ... logging to /home/sakthees-monk/.ros/log/882518a6-63cf-11f0-9b6f-e93e91f5950d/roslaunch-saktheesmonk-desktop-18938.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://saktheesmonk-desktop:45869/ ros\_comm version 1.17.4 SUMMARY ===== PARAMETERS \* /rostdistro: noetic \* /rosversion: 1.17.4 NODES RLEException: roscore cannot run as another roscore/master is already running. Please kill other roscore/master processes before relaunching. The ROS\_MASTER\_URI is http://saktheesmonk-

```

desktop:11311/ The traceback for the exception was written to the log file (rosudemy) sakthees-
monk@saktheesmonk-desktop:~$ source /opt/ros/noetic/setup.bash (rosudemy) sakthees-
monk@saktheesmonk-desktop:~$ rosrunc rqt_gui rqt_gui Traceback (most recent call last): File
"/opt/ros/noetic/lib/rqt_gui/rqt_gui", line 13, in <module> sys.exit(main.main()) File
"/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/main.py", line 61, in main return super(File
"/opt/ros/noetic/lib/python3/dist-packages/qt_gui/main.py", line 406, in main from
python_qt_binding import QT_BINDING File
"/opt/ros/noetic/lib/python3/dist-packages/python_qt_binding/__init__.py", line 55, in <module>
from .binding_helper import loadUi, QT_BINDING, QT_BINDING_MODULES,
QT_BINDING_VERSION # @UnusedImport File
"/opt/ros/noetic/lib/python3/dist-packages/python_qt_binding/binding_helper.py", line 274, in
<module> _select_qt_binding(File
"/opt/ros/noetic/lib/python3/dist-packages/python_qt_binding/binding_helper.py", line 118, in
_select_qt_binding raise ImportError(ImportError: Could not find Qt binding (looked for: 'pyqt',
'pyside'): ImportError for 'pyqt': No module named 'PyQt5' Traceback (most recent call last): File
"/opt/ros/noetic/lib/python3/dist-packages/python_qt_binding/binding_helper.py", line 108, in
_select_qt_binding QT_BINDING_VERSION = binding_loader(required_modules,
optional_modules) File
"/opt/ros/noetic/lib/python3/dist-packages/python_qt_binding/binding_helper.py", line 153, in
_load_pyqt_named_import('PyQt5.%s' % module_name) File "/opt/ros/noetic/lib/python3/dist-
packages/python_qt_binding/binding_helper.py", line 133, in _named_import module =
builtins.__import__(name) ModuleNotFoundError: No module named 'PyQt5' ImportError for
'pyside': No module named 'PySide2' Traceback (most recent call last): File
"/opt/ros/noetic/lib/python3/dist-packages/python_qt_binding/binding_helper.py", line 108, in
_select_qt_binding QT_BINDING_VERSION = binding_loader(required_modules,
optional_modules) File
"/opt/ros/noetic/lib/python3/dist-packages/python_qt_binding/binding_helper.py", line 185, in
_load_pyside_named_import('PySide2.%s' % module_name) File "/opt/ros/noetic/lib/python3/dist-
packages/python_qt_binding/binding_helper.py", line 133, in _named_import module =
builtins.__import__(name) ModuleNotFoundError: No module named 'PySide2' (rosudemy)
sakthees-monk@saktheesmonk-desktop:~$ ChatGPT said: You're very close to a complete working
ROS + Conda setup! Here's exactly what's happening and how to fix it. Problem Summary
You're running: rosrunc rqt_gui rqt_gui ...but getting this: ImportError: Could not find Qt binding
(looked for: 'pyqt', 'pyside'): No module named 'PyQt5' No module named 'PySide2' This happens
because rqt_gui is a Python GUI tool, and it can't find a Qt Python binding (PyQt5 or PySide2) —
because you're inside a Conda environment, which overrides the default system Python paths.
Solution Option 1: Install PyQt5 and/or PySide2 in Conda Since you're inside the rosudemy
Conda environment, install the Qt bindings there: conda activate rosudemy conda install pyqt -y #
For PyQt5 # OR conda install pyside2 -y # For PySide2 Then retry: rosrunc rqt_gui rqt_gui
Option 2: Use ROS Outside of Conda (Recommended for GUI tools) ROS tools like rqt_gui, rviz,
turtlesim, etc., are built for system Python, not Conda. So for these, deactivate Conda: conda
deactivate source /opt/ros/noetic/setup.bash rosrunc rqt_gui rqt_gui This should now launch without
any missing PyQt5 errors, since ROS Noetic installs the correct versions system-wide. Best
Practice Recap (Final Setup Strategy) Task Where to Run Notes roscore, rosrunc, rqt_gui Outside
Conda (system shell) Avoid Conda interference YOLOv5, ML/CV Inside Conda (rosudemy) Use
conda activate rosudemy Bridge between both source /opt/ros/noetic/setup.bash inside Conda

```

Needed for rospy, etc.      Want to Fully Test? Here's a full system + Conda ROS sanity test: # In terminal 1 (system shell) conda deactivate source /opt/ros/noetic/setup.bash roscore # In terminal 2 conda deactivate source /opt/ros/noetic/setup.bash rosrn rqt\_gui rqt\_gui # In terminal 3 conda activate rosudemy source /opt/ros/noetic/setup.bash python3 -c "import rospy; print('rospy is working in Conda!')" Would you like a script that runs all these checks and confirms your setup automatically? You said: rosrn rqt\_gui rqt\_gui

RosPluginProvider.load(qt\_gui\_cpp/CppPluginProvider) exception raised in  
\_\_builtin\_\_.\_\_import\_\_(qt\_gui\_cpp.cpp\_plugin\_provider, [CppPluginProvider]): Traceback (most recent call last): File "/opt/ros/noetic/lib/python3/dist-packages/rqt\_gui/ros\_plugin\_provider.py", line 79, in load module = \_\_builtin\_\_.\_\_import\_\_( File  
"/opt/ros/noetic/lib/python3/dist-packages/qt\_gui\_cpp/cpp\_plugin\_provider.py", line 33, in  
<module> from .cpp\_binding\_helper import qt\_gui\_cpp File "/opt/ros/noetic/lib/python3/dist-packages/qt\_gui\_cpp/cpp\_binding\_helper.py", line 43, in <module> from . import  
libqt\_gui\_cpp\_sip ValueError: PyCapsule\_GetPointer called with incorrect name  
RecursivePluginProvider.discover() loading plugin "qt\_gui\_cpp/CppPluginProvider" failed:  
Traceback (most recent call last): File  
"/opt/ros/noetic/lib/python3/dist-packages/qt\_gui/recursive\_plugin\_provider.py", line 60, in  
discover instance = self.\_plugin\_provider.load(plugin\_descriptor.plugin\_id(), None) File  
"/opt/ros/noetic/lib/python3/dist-packages/rqt\_gui/ros\_plugin\_provider.py", line 90, in load raise e  
File "/opt/ros/noetic/lib/python3/dist-packages/rqt\_gui/ros\_plugin\_provider.py", line 79, in load  
module = \_\_builtin\_\_.\_\_import\_\_( File  
"/opt/ros/noetic/lib/python3/dist-packages/qt\_gui\_cpp/cpp\_plugin\_provider.py", line 33, in  
<module> from .cpp\_binding\_helper import qt\_gui\_cpp File "/opt/ros/noetic/lib/python3/dist-packages/qt\_gui\_cpp/cpp\_binding\_helper.py", line 43, in <module> from . import  
libqt\_gui\_cpp\_sip ValueError: PyCapsule\_GetPointer called with incorrect name (rosudemy)  
sakthees-monk@saktheesmonk-desktop:~\$ error ChatGPT said: The error you're seeing:  
ValueError: PyCapsule\_GetPointer called with incorrect name ...comes from a mismatch between  
Python versions and compiled C++ Qt bindings (SIP bindings) used by rqt\_gui in ROS Noetic. This  
is very common when you're running ROS Noetic inside a custom Conda environment (like your  
(rosudemy) one), because ROS is built to work with the system's default Python 3.8 (on Ubuntu  
20.04), not Conda's Python interpreter.      Here's how to fix this: Option 1: Run rqt\_gui outside  
conda (recommended for GUI) ROS tools like rqt\_gui, rviz, etc., expect the system's Python and Qt  
setup. # First, deactivate your conda env: conda deactivate # Then run: rosrn rqt\_gui rqt\_gui If it  
works now, the issue was conda conflicting with ROS's Qt/Python bindings — which it almost  
always does with GUI tools. Option 2: If you must use Conda, set environment variables carefully  
(not always stable) You need to make sure that the Conda environment does not override system Qt,  
SIP, and Python bindings used by ROS. Before running rqt\_gui, set the following in your Conda  
environment: export PYTHONPATH=/opt/ros/noetic/lib/python3/dist-packages:\$PYTHONPATH  
export LD\_LIBRARY\_PATH=/opt/ros/noetic/lib:\$LD\_LIBRARY\_PATH export  
QT\_QPA\_PLATFORM\_PLUGIN\_PATH=/opt/ros/noetic/lib/qt5/plugins Then run: rosrn rqt\_gui  
rqt\_gui Still, this may not fix it completely due to deeper SIP mismatch between Conda Python and  
ROS's system SIP modules.      Best Practice Use Conda only for ML/CV tasks (like YOLO  
training, OpenCV, etc.) Use system Python for ROS GUI tools (rqt\_gui, rviz, etc.) You can have  
both worlds by launching ROS core/system tools from a regular terminal and running CV models  
inside the conda env with ROS bindings (if needed). To Check if ROS Itself is Working Fine  
(without GUI) Use these simple checks: # Open one terminal roscore In another terminal: rosrn

turtlesim turtlesim\_node Then: rosrun turtlesim turtle\_teleop\_key If the turtle moves with your keyboard, ROS core works fine! Let me know if you'd like help testing YOLO or ROS-OpenCV nodes next. You said: # First, deactivate your conda env: (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ conda deactivate (base) sakthees-monk@saktheesmonk-desktop:~\$ (base) sakthees-monk@saktheesmonk-desktop:~\$ # Then run: (base) sakthees-monk@saktheesmonk-desktop:~\$ rosrun rqt\_gui rqt\_gui Traceback (most recent call last): File "/opt/ros/noetic/lib/rqt\_gui/rqt\_gui", line 6, in <module> import rospkg ModuleNotFoundError: No module named 'rospkg' (base) sakthees-monk@saktheesmonk-desktop:~\$ ChatGPT said: You are encountering the ModuleNotFoundError: No module named 'rospkg' even outside your Conda environment, which means your global Python environment is missing ROS Python packages—likely because of conflicts introduced by Conda or missing environment sourcing. Let's fix this step by step: Step 1: Source ROS setup properly Ensure your shell sources ROS Noetic on every new terminal. source /opt/ros/noetic/setup.bash You can also add this to your ~/.bashrc: echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc source ~/.bashrc Step 2: Reinstall missing Python packages for ROS You need to install rospkg and other dependencies in the system Python (not Conda). Run: sudo apt install python3-rospkg python3-rospkg-modules python3-catkin-pkg-modules python3-catkin-pkg Step 3: Try running rqt\_gui again Now, try: rosrun rqt\_gui rqt\_gui Optional: Test if ROS is working properly You can test a simple publisher-subscriber communication: roscore In a new terminal: source /opt/ros/noetic/setup.bash rosrun turtlesim turtlesim\_node In another terminal: source /opt/ros/noetic/setup.bash rosrun turtlesim turtle\_teleop\_key If the turtle moves, your ROS installation works. Let me know if rqt\_gui still throws the libqt\_gui\_cpp\_sip error—we can then rebuild that package from source if needed. You said: (base) sakthees-monk@saktheesmonk-desktop:~\$ conda activate rosudemy (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ rosrun rqt\_gui rqt\_gui RosPluginProvider.load(qt\_gui\_cpp/CppPluginProvider) exception raised in \_\_builtin\_\_.\_\_import\_\_(qt\_gui\_cpp.cpp\_plugin\_provider, [CppPluginProvider]): Traceback (most recent call last): File "/opt/ros/noetic/lib/python3/dist-packages/rqt\_gui/ros\_plugin\_provider.py", line 79, in load module = \_\_builtin\_\_.\_\_import\_\_( File "/opt/ros/noetic/lib/python3/dist-packages/qt\_gui\_cpp/cpp\_plugin\_provider.py", line 33, in <module> from .cpp\_binding\_helper import qt\_gui\_cpp File "/opt/ros/noetic/lib/python3/dist-packages/qt\_gui\_cpp/cpp\_binding\_helper.py", line 43, in <module> from . import libqt\_gui\_cpp\_sip ValueError: PyCapsule\_GetPointer called with incorrect name RecursivePluginProvider.discover() loading plugin "qt\_gui\_cpp/CppPluginProvider" failed: Traceback (most recent call last): File "/opt/ros/noetic/lib/python3/dist-packages/qt\_gui/recursive\_plugin\_provider.py", line 60, in discover instance = self.\_plugin\_provider.load(plugin\_descriptor.plugin\_id(), None) File "/opt/ros/noetic/lib/python3/dist-packages/rqt\_gui/ros\_plugin\_provider.py", line 90, in load raise e File "/opt/ros/noetic/lib/python3/dist-packages/rqt\_gui/ros\_plugin\_provider.py", line 79, in load module = \_\_builtin\_\_.\_\_import\_\_( File "/opt/ros/noetic/lib/python3/dist-packages/qt\_gui\_cpp/cpp\_plugin\_provider.py", line 33, in <module> from .cpp\_binding\_helper import qt\_gui\_cpp File "/opt/ros/noetic/lib/python3/dist-packages/qt\_gui\_cpp/cpp\_binding\_helper.py", line 43, in <module> from . import libqt\_gui\_cpp\_sip ValueError: PyCapsule\_GetPointer called with incorrect name (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ ChatGPT said: You're running into a common compatibility issue when using ROS Noetic with Python virtual environments like Conda, especially when launching rqt\_gui. Here's a breakdown of the problem and a step-by-step fix:

Problem Summary ROS Noetic uses Python 3.8 (on Ubuntu 20.04). rqt\_gui uses compiled C++-Python bindings (libqt\_gui\_cpp\_sip) which are very sensitive to Python environment mismatches. When you use conda activate, you're switching to a different Python interpreter, which breaks the bindings.

Error Breakdown ValueError: PyCapsule\_GetPointer called with incorrect name This error indicates that the compiled Python module (libqt\_gui\_cpp\_sip) was built for the system Python, not for the one in your Conda environment.

Recommended Fix Never run GUI-based ROS tools like rqt\_gui, rviz, etc., from inside a Conda environment.

Steps to Reset and Run rqt\_gui Properly: Exit conda: conda deactivate Ensure you're using system Python: which python3 # Output should be: /usr/bin/python3 Source ROS environment: source /opt/ros/noetic/setup.bash Then run: rosrun rqt\_gui rqt\_gui Check Everything Works To verify ROS Noetic is properly installed and working:

1. Check roscore: roscore You should see: started core service [/rosout]
2. In a new terminal: source /opt/ros/noetic/setup.bash rosrun turtlesim turtlesim\_node
3. In another terminal: source /opt/ros/noetic/setup.bash rosrun turtlesim turtle\_teleop\_key → You should be able to control the turtle with arrow keys.

Summary rqt\_gui fails inside Conda because ROS was built with system Python. Run all GUI tools outside Conda. Use Conda only for custom scripts, ML models, or packages that don't rely on ROS GUI binaries. Let me know if you want to set up a ROS+Conda workflow where only custom Python scripts run inside Conda, but the rest use system ROS.

You said: (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ which python3 /home/sakthees-monk/Downloads/yes/envs/rosudemy/bin/python3 (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ # Output should be: /usr/bin/python3 (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ source /opt/ros/noetic/setup.bash (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ rosrun rqt\_gui rqt\_gui RosPluginProvider.load(qt\_gui\_cpp/CppPluginProvider) exception raised in \_\_builtin\_\_.\_\_import\_\_(qt\_gui\_cpp.cpp\_plugin\_provider, [CppPluginProvider]): Traceback (most recent call last): File "/opt/ros/noetic/lib/python3/dist-packages/rqt\_gui/ros\_plugin\_provider.py", line 79, in load module = \_\_builtin\_\_.\_\_import\_\_( File "/opt/ros/noetic/lib/python3/dist-packages/qt\_gui\_cpp/cpp\_plugin\_provider.py", line 33, in <module> from .cpp\_binding\_helper import qt\_gui\_cpp File "/opt/ros/noetic/lib/python3/dist-packages/qt\_gui\_cpp/cpp\_binding\_helper.py", line 43, in <module> from . import libqt\_gui\_cpp\_sip ValueError: PyCapsule\_GetPointer called with incorrect name RecursivePluginProvider.discover() loading plugin "qt\_gui\_cpp/CppPluginProvider" failed: Traceback (most recent call last): File "/opt/ros/noetic/lib/python3/dist-packages/qt\_gui/recursive\_plugin\_provider.py", line 60, in discover instance = self.\_plugin\_provider.load(plugin\_descriptor.plugin\_id(), None) File "/opt/ros/noetic/lib/python3/dist-packages/rqt\_gui/ros\_plugin\_provider.py", line 90, in load raise e File "/opt/ros/noetic/lib/python3/dist-packages/rqt\_gui/ros\_plugin\_provider.py", line 79, in load module = \_\_builtin\_\_.\_\_import\_\_( File "/opt/ros/noetic/lib/python3/dist-packages/qt\_gui\_cpp/cpp\_plugin\_provider.py", line 33, in <module> from .cpp\_binding\_helper import qt\_gui\_cpp File "/opt/ros/noetic/lib/python3/dist-packages/qt\_gui\_cpp/cpp\_binding\_helper.py", line 43, in <module> from . import libqt\_gui\_cpp\_sip ValueError: PyCapsule\_GetPointer called with incorrect name (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ roscore ... logging to /home/sakthees-monk/.ros/log/882518a6-63cf-11f0-9b6f-e93e91f5950d/roslaunch-saktheesmonk-desktop-19650.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://saktheesmonk-desktop:39969/ ros\_comm version 1.17.4 SUMMARY =====

PARAMETERS \* /rostdistro: noetic \* /rosversion: 1.17.4 NODES RLEException: roscore cannot run as another roscore/master is already running. Please kill other roscore/master processes before relaunching. The ROS\_MASTER\_URI is http://saktheesmonk-desktop:11311/ The traceback for the exception was written to the log file give pip of ros dep ChatGPT said: You're very close to a working ROS Noetic setup inside your rosudemy Conda environment. Let's handle your questions and errors one by one: You've done correctly: Created a Conda env rosudemy. ROS is sourced via source /opt/ros/noetic/setup.bash. roscore started properly (though it says another instance is running). Error in rosrqt\_gui rqt\_gui Error: ValueError: PyCapsule\_GetPointer called with incorrect name Cause: This error usually occurs when ROS tools like rqt\_gui are run from a Conda environment with a different Python version or incompatible SIP bindings than what ROS expects. ROS Noetic expects Python 3.8 (the system Python on Ubuntu 20.04), and specifically not a Conda Python or custom SIP install. Your current Python is: which python3 /home/sakthees-monk/Downloads/yes/envs/rosudemy/bin/python3 This is Conda's Python, not system Python (/usr/bin/python3). That is the issue. Fix Options: Option 1: Run ROS tools outside Conda Just use the terminal outside your Conda environment: conda deactivate source /opt/ros/noetic/setup.bash rosrqt\_gui rqt\_gui Why?: Tools like rqt\_gui and rviz must use system Python and native bindings, not Conda. Option 2: Keep using Conda for code, not for GUI tools Use rosudemy environment only for your scripts, training, or YOLO pipelines, not for rqt\_gui, rviz, etc. Installing rosdep using pip in Conda If you're asking how to install rosdep inside Conda: pip install rosdep But it's better to install it system-wide using: sudo apt install python3-rosdep Then initialize once: sudo rosdep init rosdep update Check that everything works: Check ROS setup: source /opt/ros/noetic/setup.bash roscore New terminal: Run a basic talker/listener test: rosrqt\_tutorials talker Another terminal: rosrqt\_tutorials listener YOLO or your training pipeline: Run it from Conda rosudemy env — works best for training. Summary: Task Run in Conda? Run Outside Conda? roscore, rqt\_gui, rviz No Yes Python training (YOLO, etc) Yes No ROS rosdep setup No (system) Yes If you want help verifying your rosdep is working or need to test YOLO inside the ROS node, I can guide you further. Unlock more with Plus ChatGPT Plus gives you higher limits, smarter models, and Sora for video. ChatGPT can make mistakes. Check important info. See Cookie Preferences. give to test ros core installed correctly

ChatGPT said:

From the logs and commands you've shared, your ROS Noetic environment is mostly working, but you've encountered a few key issues. Let's go step by step and fix them:

---

### Summary of Current State

1. You have successfully activated your Conda environment `rosudemy`.
  2. `roscore` is working (though another instance was already running).
  3. Running `rqt_gui` throws an error due to Python/C++ SIP binding issues.
  4. You're asking for the correct **pip install command for rosdep**.
-

## 1. Install rosdep via pip inside your rosudemy environment

Run this command **inside your activated environment**:

```
pip install rosdep
```

If you want to check which rosdep is being used, run:

```
which rosdep
```

It should point to:

```
/home/sakthees-monk/Downloads/yes/envs/rosudemy/bin/rosdep
```

---

## 2. Fixing rqt\_gui Plugin Error (SIP binding issue)

This is a **known compatibility issue** caused by the mismatch between system-installed ROS and Conda's Python interpreter. ROS expects system Python and certain compiled .so bindings to match.

### Solution Options:

#### Option 1: Run rqt\_gui with system Python (Recommended for GUI)

Use a separate terminal **outside Conda**, and simply do:

```
source /opt/ros/noetic/setup.bash
rqt_gui
```

This avoids Conda Python and uses the system's matching environment.

#### Option 2: Fix inside Conda (not recommended, tedious)

You would have to:

- Match exact Python version (e.g. 3.8 for Noetic)
- Ensure python-sip and qt\_gui\_cpp bindings match
- Recompile qt\_gui\_cpp from source inside your Conda

This is complex and error-prone.

---

## 3. Check Everything Is Working Properly

After roscore, you can test a simple ROS node.

### In Terminal 1:

```
roscore
```

**In Terminal 2 (after `source /opt/ros/noetic/setup.bash`):**

```
roslaunch turtlesim turtlesim_node
```

**In Terminal 3:**

```
roslaunch turtlesim turtle_teleop_key
```

If the turtle window pops up and you can control it — ROS is working correctly.

---

## 4. Optional: Set ROS automatically with Conda env

You can edit your environment's activation hook:

```
nano ~/anaconda3/envs/rosudemy/etc/conda/activate.d/env_vars.sh
```

Add:

```
#!/bin/bash
source /opt/ros/noetic/setup.bash
```

Then make it executable:

```
chmod +x ~/anaconda3/envs/rosudemy/etc/conda/activate.d/env_vars.sh
```

Now, every time you do `conda activate rosudemy`, it will also source ROS.

---

Would you like me to generate a `.yaml` file for this environment that includes all necessary packages for ROS + YOLO + Udemy-style development?

**You said:**

```
source /opt/ros/noetic/setup.bash (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ rqt_gui
rqt_gui: command not found
```

**ChatGPT said:**

Thanks for the update.

You're getting:

```
rqt_gui: command not found
```

This means that the `rqt_gui` package is **not installed** on your system or not available in your `PATH`. Since `rqt_gui` is part of the ROS Noetic desktop installation, let's confirm and install it.

---

## Step-by-Step Fix

### 1. First, ensure ROS environment is sourced

In any terminal:



```
source /opt/ros/noetic/setup.bash
```

(Do **not** activate Conda for this step. This is better tested with system Python.)

---

## 2. Install `rqt_gui` and its plugins:

Run this:

```
sudo apt update
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins
```

This installs:

- `rqt_gui`
  - Visualization plugins (`rqt_plot`, `rqt_graph`, etc.)
- 

## 3. Test `rqt_gui`:

Now run:

```
rqt_gui
```

You should see the RQT window with plugin menus.

---

## Optional: Run inside Conda environment

If you **really want to run `rqt_gui` inside your Conda env** (though not recommended), ensure the system `rqt_gui` binary is available inside Conda's PATH:

Add this to your Conda env activation script:

```
export PATH=/opt/ros/noetic/bin:$PATH
```

And again:

```
source /opt/ros/noetic/setup.bash
```

But GUI tools like `rqt_gui` work best **outside Conda**, since they depend on system Python SIP bindings and Qt.

---

Would you like a full test checklist (turtlesim, publisher/subscriber, visualization, etc.) to verify your ROS install completely?

**You said:**

```
(rosudemy) sakthees-monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui: command not found
(rosudemy) sakthees-monk@saktheesmonk-desktop:~$
```

ChatGPT said:

You're in the **rosudemy conda environment**, and you're seeing:

```
rqt_gui: command not found
```

This happens because:

1. The `rqt_gui` executable is installed in the **system ROS environment**, not inside your conda environment.
  2. Your `PATH` inside the conda environment does **not** include `/opt/ros/noetic/bin`.
- 

## Fix in Conda: Add ROS binaries to PATH

To fix it *inside the conda environment*, run this:

```
source /opt/ros/noetic/setup.bash
```

Then try:

```
rqt_gui
```

If it still says "command not found", check:

```
which rqt_gui
```

If no path is shown, then the binary is not found.

---

## Permanent Fix (Optional): Add to `.bashrc` in conda env

If you want to **always have ROS available** when `rosudemy` is activated, edit the `activate.d` script for that env:

```
mkdir -p ~/miniconda3/envs/rosudemy/etc/conda/activate.d
nano ~/miniconda3/envs/rosudemy/etc/conda/activate.d/ros_setup.sh
```

Paste this:

```
#!/bin/bash
source /opt/ros/noetic/setup.bash
```

Save and exit (Ctrl+O, Enter, Ctrl+X).

Now, every time you `conda activate rosudemy`, it will source ROS and make `rqt_gui` available.

---

## Final Check

After reactivating your env:

```
conda deactivate
```

```
conda activate rosudemy
rqt_gui
```

If it works, the RQT window should launch.

---

Let me know if you want a full ROS test checklist (like `turtlesim`, `rqt_graph`, `rostopic echo`, etc.) to verify all tools are working.

**You said:**

```
rosudemy) sakthees-monk@saktheesmonk-desktop:~$ source /opt/ros/noetic/setup.bash (rosudemy)
sakthees-monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui: command not found (rosudemy)
sakthees-monk@saktheesmonk-desktop:~$ ok add permanatlyt o gedit bashrc
```

ChatGPT said:

Great — to **permanently add ROS to your environment**, here's how to add it to your `~/.bashrc` using `gedit`.

---

## Step-by-Step to Add ROS to `.bashrc`

1. Open `.bashrc` with `gedit`:

- `gedit ~/.bashrc`
- Scroll to the **bottom** of the file, and **add this line**:
- `source /opt/ros/noetic/setup.bash`
- Save and close the file (`Ctrl + S`, then close `gedit`).
- Reload `.bashrc` to apply changes immediately:

4. `source ~/.bashrc`

---

## Now test it:

Try launching:

```
rqt_gui
```

If it still shows `command not found`, make sure the ROS Noetic package for RQT is installed:

---

## Install `rqt_gui` and dependencies if missing:

```
sudo apt update
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins
```

Then try again:

```
rqt_gui
```

---

Let me know if you'd like to test ROS with something visual like `turtlesim`!

**You said:**

```
Unpacking ros-noetic-rqt (0.5.5-1focal.20250519.233427) ... Setting up ros-noetic-rqt (0.5.5-1focal.20250519.233427) ... (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ rqt_gui
rqt_gui: command not found (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ source
/opt/ros/noetic/setup.bash (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui:
command not found (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ gedit ~/.bashrc
(rosudemy) sakthees-monk@saktheesmonk-desktop:~$ source ~/.bashrc sakthees-
monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui: command not found sakthees-
monk@saktheesmonk-desktop:~$ conda activate (base) sakthees-monk@saktheesmonk-desktop:~$
conda activate rosudemy (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui:
command not found (rosudemy) sakthees-monk@saktheesmonk-desktop:~$
```

ChatGPT said:

You're very close — the issue is that **ROS tools like `rqt_gui` are not available inside your Conda environment (rosudemy) by default**, because Conda **isolates** the environment and does not automatically include system-wide binaries like ROS tools.

---

## **Fix: Make ROS tools (like `rqt_gui`) available in your Conda environment**

### **Step 1: Add ROS environment to your Conda environment manually**

Run this command **inside your rosudemy conda environment**:

```
echo "source /opt/ros/noetic/setup.bash" >>
~/miniconda3/envs/rosudemy/etc/conda/activate.d/env_vars.sh
```

Replace `miniconda3` with `anaconda3` if you use Anaconda.

Also make the deactivate script to clean up after:

```
mkdir -p ~/miniconda3/envs/rosudemy/etc/conda/deactivate.d
echo "unset ROS_PACKAGE_PATH" >>
~/miniconda3/envs/rosudemy/etc/conda/deactivate.d/env_vars.sh
```

---

### **Step 2: Reactivate your conda env**

```
conda deactivate
conda activate rosudemy
```

---

### **Now try again:**

```
rqt_gui
```

It should launch the GUI now.

---

## If you still get "command not found"

It likely means the ROS tools aren't in your system PATH. Try this manually:

```
export PATH=$PATH:/opt/ros/noetic/bin
```

Then:

```
rqt_gui
```

---

Let me know if you'd like to test ROS with something like:

```
roscore
roslaunch turtlesim turtlesim_node
```

**You said:**

```
akthees-monk@saktheesmonk-desktop:~/conda$ pwd /home/sakthees-monk/.conda sakthees-monk@saktheesmonk-desktop:~/conda$ "(rosudemy) sakthees-monk@saktheesmonk-desktop:~$ source /opt/ros/noetic/setup.bash (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui: command not found (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ gedit ~/.bashrc (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ source ~/.bashrc sakthees-monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui: command not found sakthees-monk@saktheesmonk-desktop:~$ conda activate (base) sakthees-monk@saktheesmonk-desktop:~$ conda activate rosudemy (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui: command not found (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ echo "source /opt/ros/noetic/setup.bash" >> ~/miniconda3/envs/rosudemy/etc/conda/activate.d/env_vars.sh bash: /home/sakthees-monk/miniconda3/envs/rosudemy/etc/conda/activate.d/env_vars.sh: No such file or directory (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ "
```

ChatGPT said:

Thanks for the details. Here's exactly what's happening and how to **fix** it:

---

## Problem:

You installed ROS Noetic system-wide (outside of conda), but when you activate your Conda environment (rosudemy), the `rqt_gui` command is still **not found**.

This is likely because:

1. ROS Noetic's executables (like `rqt_gui`) are **not in the Conda environment's PATH**.
  2. Even though you sourced `/opt/ros/noetic/setup.bash`, conda may override PATH or not persist it correctly across sessions.
-

## Fix — Step-by-Step:

### Step 1: Manually Check if `rqt_gui` exists

Run this command to see where `rqt_gui` is located:

```
which rqt_gui
```

If it returns something like `/opt/ros/noetic/bin/rqt_gui`, then the tool is installed properly system-wide.

---

### Step 2: Add ROS setup to your conda env permanently

We'll **create** the required conda hook folders and script.

Run:

```
mkdir -p ~/miniconda3/envs/rosudemy/etc/conda/activate.d
mkdir -p ~/miniconda3/envs/rosudemy/etc/conda/deactivate.d
```

Then create the **activation script**:

```
gedit ~/miniconda3/envs/rosudemy/etc/conda/activate.d/ros.sh
```

Paste this inside:

```
#!/bin/bash
source /opt/ros/noetic/setup.bash
```

Then also create the **deactivation script**:

```
gedit ~/miniconda3/envs/rosudemy/etc/conda/deactivate.d/ros.sh
```

Paste this to remove ROS from PATH when deactivating:

```
#!/bin/bash
export PATH=$(echo "$PATH" | tr ':' '\n' | grep -v "/opt/ros/noetic/bin" | paste
-sd ':' -)
```

Make both scripts executable:

```
chmod +x ~/miniconda3/envs/rosudemy/etc/conda/activate.d/ros.sh
chmod +x ~/miniconda3/envs/rosudemy/etc/conda/deactivate.d/ros.sh
```

---

### Step 3: Reactivate the environment

```
conda deactivate
conda activate rosudemy
rqt_gui
```

If everything worked, `rqt_gui` should now open.

---

## Extra Check:

If `rqt_gui` still doesn't work:

```
ls /opt/ros/noetic/bin | grep rqt_gui
```

This ensures the executable is present.

---

Let me know if you'd like to also **link ROS tools into Conda's PATH manually** or build ROS inside a Conda environment fully.

**You said:**

rosudemy check path

ChatGPT said:

To **check if your ROS Noetic environment (rosudemy) has the correct path to `rqt_gui`**, follow these steps:

---

### Step 1: Check if `rqt_gui` is installed

Run this command outside the conda environment:

```
which rqt_gui
```

If installed correctly, you should see:

```
/opt/ros/noetic/bin/rqt_gui
```

If not, install it using:

```
sudo apt install ros-noetic-rqt
```

---

### Step 2: Check if `rqt_gui` is in PATH inside conda env

Activate your environment:

```
conda activate rosudemy
```

Then check:

```
echo $PATH | grep ros
```

You should see something like:

```
/opt/ros/noetic/bin
```

If not, the ROS path is missing from your conda env.

---

### Step 3: Fix conda activation hook

Manually create the required file to source ROS setup every time:

```
mkdir -p ~/miniconda3/envs/rosudemy/etc/conda/activate.d
gedit ~/miniconda3/envs/rosudemy/etc/conda/activate.d/ros_env.sh
```

Add this line inside the file:

```
source /opt/ros/noetic/setup.bash
```

Save and exit.

Then **deactivate and reactivate**:

```
conda deactivate
conda activate rosudemy
```

Now test:

```
which rqt_gui
rqt_gui
```

---

### Optional: Debug with env

You can run:

```
env | grep ROS
```

You should see ROS environment variables like:

```
ROS_ROOT=/opt/ros/noetic/share/ros
ROS_PACKAGE_PATH=/opt/ros/noetic/share
```

---

Let me know what output you get from these steps, especially `which rqt_gui` and `echo $PATH | grep ros`.

**You said:**

```
(rosudemy) sakthees-monk@saktheesmonk-desktop:~$ sudo apt install ros-noetic-rqt Reading
package lists... Done Building dependency tree Reading state information... Done ros-noetic-rqt is
already the newest version (0.5.5-1focal.20250519.233427). The following packages were
automatically installed and are no longer required: chromium-codecs-ffmpeg-extra gstreamer1.0-
vaapi lib32gcc-s1 libc6-i386 libgstreamer-plugins-bad1.0-0 linux-headers-5.15.0-67-generic linux-
hwe-5.15-headers-5.15.0-67 linux-image-5.15.0-67-generic linux-modules-5.15.0-67-generic linux-
modules-extra-5.15.0-67-generic Use 'sudo apt autoremove' to remove them. 0 upgraded, 0 newly
installed, 0 to remove and 1 not upgraded. (rosudemy) sakthees-monk@saktheesmonk-desktop:~$
conda activate rosudemy (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ echo $PATH |
grep ros /opt/ros/noetic/bin:/home/sakthees-monk/Downloads/yes/envs/rosudemy/bin:/home/
sakthees-monk/Downloads/yes/condabin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/
usr/games:/usr/local/games:/snap/bin:/usr/local/android-studio/bin:/usr/local/android-studio/bin
```



(rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ /opt/ros/noetic/bin bash: /opt/ros/noetic/bin:  
Is a directory (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ find the ppath and do

ChatGPT said:

Great! From your output, it's clear that:

- `ros-noetic-rqt` is installed.
  - `/opt/ros/noetic/bin` is in your **\$PATH**.
  - You're in the correct conda environment (rosudemy).
- 

Now let's **verify and run rqt\_gui** properly.

---

### Step 1: Confirm it's in the path

Run this command:

```
which rqt_gui
```

Expected output:

```
/opt/ros/noetic/bin/rqt_gui
```

If that shows up, run:

```
rqt_gui
```

---

### If `rqt_gui` does not launch:

Try launching with full path:

```
/opt/ros/noetic/bin/rqt_gui
```

If this still **doesn't work**, check for missing display or plugin issues:

```
echo $DISPLAY
```

You should get something like:

```
:0
```

If it's empty, run:

```
export DISPLAY=:0
```

Then try again:

```
rqt_gui
```

---

## Optional Debugging

Check Python errors from rqt:

```
rqt_gui --debug
```

---

## Permanent Fix for Activation

Ensure this file exists:

```
nano ~/miniconda3/envs/rosudemy/etc/conda/activate.d/ros_env.sh
```

And contains:

```
source /opt/ros/noetic/setup.bash
export DISPLAY=:0
```

Save and Ctrl+O, Ctrl+X. Then:

```
conda deactivate
conda activate rosudemy
rqt_gui
```

---

Let me know:

1. What you get from `which rqt_gui`
2. Whether `rqt_gui` launches or gives error logs.

**You said:**

The following packages were automatically installed and are no longer required: chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi lib32gcc-s1 libc6-i386 libgstreamer-plugins-bad1.0-0 linux-headers-5.15.0-67-generic linux-hwe-5.15-headers-5.15.0-67 linux-image-5.15.0-67-generic linux-modules-5.15.0-67-generic linux-modules-extra-5.15.0-67-generic Use 'sudo apt autoremove' to remove them. 0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded. (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ conda activate rosudemy (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ echo \$PATH | grep ros /opt/ros/noetic/bin:/home/sakthees-monk/Downloads/yes/envs/rosudemy/bin:/home/sakthees-monk/Downloads/yes/condabin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/usr/local/android-studio/bin:/usr/local/android-studio/bin (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ /opt/ros/noetic/bin bash: /opt/ros/noetic/bin: Is a directory (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ which rqt\_gui (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ /opt/ros/noetic/bin/rqt\_gui bash: /opt/ros/noetic/bin/rqt\_gui: No such file or directory (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ /opt/ros/noetic/bin/rqt\_gui bash: /opt/ros/noetic/bin/rqt\_gui: No such file or directory (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$

**ChatGPT said:**

Thanks! Based on your output:

- `/opt/ros/noetic/bin` is in your `$PATH`.

- But `rqt_gui` **does not exist**, even though `ros-noetic-rqt` is installed.

This can happen if the RQT GUI dependencies are **not fully installed**, or if there's a broken/missing plugin.

---

## Let's fix it step by step:

---

### Step 1: Install full RQT suite (especially `rqt_gui`, `rqt_common_plugins`, etc.)

Run this:

```
sudo apt update
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins
```

These install the full GUI and all necessary backends (like `rqt_gui`, `rqt_graph`, `rqt_console`, etc.)

---

### Step 2: Recheck executable

After installation:

```
which rqt_gui
```

Expected output:

```
/opt/ros/noetic/bin/rqt_gui
```

Now try launching:

```
rqt_gui
```

---

### Step 3: If it still fails

Try locating it manually:

```
find /opt/ros/noetic/ -name rqt_gui
```

If that still shows nothing, check if Python entrypoints exist:

```
roslaunch rqt_gui rqt_gui
```

or

```
python3 -m rqt_gui
```

Make sure to run this **inside your rosudemy conda env**.

---

## Additional Fix: Source and Python path

Add this to your conda env activation script:

```
mkdir -p ~/miniconda3/envs/rosudemy/etc/conda/activate.d
nano ~/miniconda3/envs/rosudemy/etc/conda/activate.d/ros_env.sh
```

Add:

```
#!/bin/bash
source /opt/ros/noetic/setup.bash
export DISPLAY=:0
```

Then run:

```
conda deactivate
conda activate rosudemy
rqt_gui
```

---

Let me know what you get after installing:

```
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins
```

I'll guide you from there.

**You said:**

```
ove and 1 not upgraded. (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ which rqt_gui
(rosudemy) sakthees-monk@saktheesmonk-desktop:~$ /opt/ros/noetic/bin/rqt_gui bash:
/opt/ros/noetic/bin/rqt_gui: No such file or directory (rosudemy) sakthees-monk@saktheesmonk-
desktop:~$
```

**ChatGPT said:**

The issue is that the `rqt_gui` command is **not found** on your system because it hasn't been installed yet.

---

## To fix this:

Install the `rqt_gui` package for ROS Noetic:

```
sudo apt update
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins
```

This will install:

- `rqt_gui`
  - Common useful plugins
- 

## After installation, verify:

```
which rqt_gui
```

You should see:

```
/opt/ros/noetic/bin/rqt_gui
```

Then you can run:

```
rqt_gui
```

---

### **Optional: Add ROS to .bashrc permanently**

If not already done, add this to the end of your ~/.bashrc:

```
source /opt/ros/noetic/setup.bash
```

Then reload it:

```
source ~/.bashrc
```

Let me know once you've installed the packages — we'll check further if needed.

**You said:**

now to checek ll gfive

ChatGPT said:

Great! Now let's verify that your ROS Noetic setup and environment (rosudemy) are **working properly**. Here's a complete checklist and commands to run:

---

## **1. Check ROS Environment**

Run these commands **inside your activated conda environment (rosudemy)**:

```
echo $ROS_DISTRO
```

Expected output:

```
noetic
```

```
which roscore
```

Expected output:

```
/opt/ros/noetic/bin/roscore
```

---

## **2. Start roscore**

```
roscore
```

Expected:

```
... logging to /home/youruser/.ros/log/...
started core service [/rosout]
```

If it runs, your ROS core is working.

Leave this terminal running and **open a new terminal** (or tab) for the next steps.

---

### 3. Check `rqt_gui`

In the new terminal (still inside `rosudemy` environment):

```
rqt_gui
```

This should open the RQT graphical interface.

If it fails, paste the error message here.

---

### 4. Check ROS Nodes

Test if nodes work by running a simple publisher/subscriber:

In **Terminal 1**:

```
roscore
```

In **Terminal 2**:

```
roslaunch turtlesim turtlesim_node
```

In **Terminal 3**:

```
roslaunch turtlesim turtle_teleop_key
```

Use arrow keys to move the turtle. If it works, your ROS setup is fully functional.

---

### 5. Optional: Test Environment Path

Check Python paths and ROS packages visibility:

```
echo $PATH
echo $PYTHONPATH
```

---

If all of the above work, then your `rosudemy` environment and ROS Noetic are set up correctly!  
Let me know if anything fails in these steps, and I'll help debug it.

**You said:**

```
sakthees-monk@saktheesmonk-desktop:~$ conda activate (base) sakthees-monk@saktheesmonk-
sakthees-monk@saktheesmonk-desktop:~$ conda activate rosudemy (rosudemy) sakthees-monk@saktheesmonk-
rqt_gui rqt_gui: command not found (rosudemy) sakthees-monk@saktheesmonk-desktop:~$
```

ChatGPT said:

Thanks — this confirms the issue:

```
rqt_gui: command not found
even after activating rosudemy and sourcing /opt/ros/noetic/setup.bash.
```

---

**Let's fix it step-by-step:**

---

## 1. Install rqt\_gui and dependencies

Run this in your terminal (outside conda base path; it installs system-wide):

```
sudo apt update
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins
```

This installs:

- rqt\_gui
  - rqt\_graph, rqt\_plot, rqt\_console, etc.
- 

## 2. Re-source ROS and test

After installation:

```
source /opt/ros/noetic/setup.bash
rqt_gui
```

It should now open the RQT GUI.

---

## Optional: Add source to .bashrc

So you don't have to run `source /opt/ros/noetic/setup.bash` every time, add it to your `.bashrc`:

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

---

Let me know if it opens now or if you see any new errors — I'll help further if needed.

**You said:**

W: Target DEP-11 (main/dep11/Components-all.yml) is configured multiple times in /etc/apt/sources.list.d/ros-latest.list:1 and /etc/apt/sources.list.d/ros1-latest.list:1 W: Target DEP-11-icons-small (main/dep11/icons-48x48.tar) is configured multiple times in /etc/apt/sources.list.d/ros-latest.list:1 and /etc/apt/sources.list.d/ros1-latest.list:1 W: Target DEP-11-icons (main/dep11/icons-

64x64.tar) is configured multiple times in /etc/apt/sources.list.d/ros-latest.list:1 and /etc/apt/sources.list.d/ros1-latest.list:1 W: Target DEP-11-icons-hidpi (main/dep11/icons-64x64@2.tar) is configured multiple times in /etc/apt/sources.list.d/ros-latest.list:1 and /etc/apt/sources.list.d/ros1-latest.list:1 W: Target CNF (main/cnf/Commands-amd64) is configured multiple times in /etc/apt/sources.list.d/ros-latest.list:1 and /etc/apt/sources.list.d/ros1-latest.list:1 W: Target CNF (main/cnf/Commands-all) is configured multiple times in /etc/apt/sources.list.d/ros-latest.list:1 and /etc/apt/sources.list.d/ros1-latest.list:1 (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ source /opt/ros/noetic/setup.bash (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$ rqt\_gui rqt\_gui: command not found (rosudemy) sakthees-monk@saktheesmonk-desktop:~\$

ChatGPT said:

Thanks for the update. Let's carefully solve both issues:

---

## 1. [Fix] ROS sources duplicated — harmless but clean it

You're getting warnings because ROS was added **twice** to APT sources:

- `ros-latest.list`
- `ros1-latest.list`

### Solution:

Run:

```
sudo rm /etc/apt/sources.list.d/ros1-latest.list
sudo apt update
```

This removes the redundant list and clears warnings.

---

## 2. [Fix] `rqt_gui`: command not found

You're still getting this **because `ros-noetic-rqt` and its plugins aren't installed yet.**

### Run this to install all RQT-related tools:

```
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins
```

This will install:

- `rqt_gui`
- `rqt_plot`, `rqt_graph`, `rqt_console`, etc.

After that:

```
source /opt/ros/noetic/setup.bash
rqt_gui
```

It should now launch the GUI.



---

## Bonus: Add ROS to your shell permanently

To avoid manually running `source /opt/ros/noetic/setup.bash` every time, add it to your `.bashrc`:

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

---

Let me know if RQT opens now or if you get a new error.

**You said:**

```
thees-monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui: command not found (rosudemy)
sakthees-monk@saktheesmonk-desktop:~$ echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc (rosudemy)
sakthees-monk@saktheesmonk-desktop:~$ source ~/.bashrc sakthees-monk@saktheesmonk-
desktop:~$ save within rosudmy coda envuironment so only not working
```

ChatGPT said:

Got it. You're inside a **virtual environment** called `rosudemy`, which is why `rqt_gui` is not found — the ROS environment (`/opt/ros/noetic/setup.bash`) is not being sourced **within that virtualenv**.

Let's fix that properly:

---

## Step-by-Step Fix for `rqt_gui` inside `virtualenv` (`rosudemy`)

### 1. Activate your `virtualenv` and manually source ROS

```
source ~/path/to/rosudemy/bin/activate # Or however you activate it
source /opt/ros/noetic/setup.bash
```

Then try:

```
rqt_gui
```

If that works, you're good. But for convenience, we can **auto-source ROS inside the virtualenv**.

---

### 2. Make ROS auto-source inside `virtualenv`

Edit the `activate` script of your `virtualenv`:

```
nano ~/path/to/rosudemy/bin/activate
```

**Scroll to the bottom** and add this line:

```
source /opt/ros/noetic/setup.bash
```

Then save and exit (Ctrl+O, Enter, Ctrl+X).

Now, every time you do:

```
source ~/path/to/rosudemy/bin/activate
```

It will automatically also source the ROS environment.

---

### 3. Try again

```
source ~/path/to/rosudemy/bin/activate
rqt_gui
```

---

## If it still says "command not found"

Then you're **missing the installation** of RQT in your system (not related to Python env).

Install it:

```
sudo apt update
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins
```

---

Let me know if you want help modifying the `activate` script or locating your virtualenv path.

**You said:**

```
osudemy) sakthees-monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui: command not found
(rosudemy) sakthees-monk@saktheesmonk-desktop:~$ echo "source /opt/ros/noetic/setup.bash" >>
~/bashrc (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ source ~/.bashrc sakthees-
monk@saktheesmonk-desktop:~$ conda activate (base) sakthees-monk@saktheesmonk-desktop:~$
conda activate rosudemy (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ source
~/path/to/rosudemy/bin/activate # Or however you activate it bash:
/home/sakthees-monk/path/to/rosudemy/bin/activate: No such file or directory (rosudemy)
sakthees-monk@saktheesmonk-desktop:~$ source /opt/ros/noetic/setup.bash (rosudemy) sakthees-
monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui: command not found (rosudemy) sakthees-
monk@saktheesmonk-desktop:~$
```

**ChatGPT said:**

Thanks for the detailed trace. Let's solve this step by step:

---

### Problem Summary:

- You're in a `conda` environment named `rosudemy`.
  - You sourced `/opt/ros/noetic/setup.bash` — good.
  - Still, `rqt_gui` gives: `command not found`.
-

## 1. Check if `rqt_gui` is installed

Run this:

```
apt list --installed | grep rqt
```

If it doesn't list anything like `ros-noetic-rqt`, then you need to install it.

---

## 2. Install RQT packages (if missing)

Run this (outside the virtual environment, or with `sudo`):

```
sudo apt update
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins
```

This will install:

- `rqt_gui`
  - all the core RQT plugins (like `rqt_plot`, `rqt_graph`, etc.)
- 

## 3. Try running `rqt_gui` again

After installation:

```
source /opt/ros/noetic/setup.bash
rqt_gui
```

If it works here, it will work inside the `rosudemy` conda environment too **as long as** you source `/opt/ros/noetic/setup.bash` **after** activating `rosudemy`.

---

## Optional: Make ROS auto-load with `conda activate rosudemy`

To avoid running `source /opt/ros/noetic/setup.bash` every time, do this:

```
nano ~/miniconda3/envs/rosudemy/etc/conda/activate.d/env_vars.sh
```

Add this line:

```
source /opt/ros/noetic/setup.bash
```

If the folder doesn't exist, create it:

```
mkdir -p ~/miniconda3/envs/rosudemy/etc/conda/activate.d
nano ~/miniconda3/envs/rosudemy/etc/conda/activate.d/env_vars.sh
```

---

# Final Check

```
conda activate rosudemy
rqt_gui
```

Should now work properly.

---

Let me know what `apt list --installed | grep rqt` shows — I'll guide further based on that.

## You said:

```
(rosudemy) sakthees-monk@saktheesmonk-desktop:~$ apt list --installed | grep rqt
WARNING: apt does not have a stable CLI interface. Use with caution in scripts.
ros-noetic-rqt-action/focal,now 0.4.11-1focal.20250520.001424 amd64 [installed,automatic]
ros-noetic-rqt-bag-plugins/focal,now 0.5.3-1focal.20250520.001317 amd64 [installed,automatic]
ros-noetic-rqt-bag/focal,now 0.5.3-1focal.20250519.235942 amd64 [installed,automatic]
ros-noetic-rqt-common-plugins/focal,now 0.4.11-1focal.20250521.005128 amd64 [installed]
ros-noetic-rqt-console/focal,now 0.4.14-1focal.20250520.001205 amd64 [installed,automatic]
ros-noetic-rqt-dep/focal,now 0.4.14-1focal.20250520.000033 amd64 [installed,automatic]
ros-noetic-rqt-graph/focal,now 0.4.16-1focal.20250519.235907 amd64 [installed,automatic]
ros-noetic-rqt-gui-cpp/focal,now 0.5.5-1focal.20250519.232315 amd64 [installed,automatic]
ros-noetic-rqt-gui-py/focal,now 0.5.5-1focal.20250519.233238 amd64 [installed,automatic]
ros-noetic-rqt-gui/focal,now 0.5.5-1focal.20250519.232427 amd64 [installed,automatic]
ros-noetic-rqt-image-view/focal,now 0.4.19-1focal.20250520.001352 amd64 [installed,automatic]
ros-noetic-rqt-launch/focal,now 0.4.10-1focal.20250520.001300 amd64 [installed,automatic]
ros-noetic-rqt-logger-level/focal,now 0.4.13-1focal.20250519.235914 amd64 [installed,automatic]
ros-noetic-rqt-moveit/focal,now 0.5.13-1focal.20250520.001222 amd64 [installed,automatic]
ros-noetic-rqt-msg/focal,now 0.4.12-1focal.20250520.001309 amd64 [installed,automatic]
ros-noetic-rqt-nav-view/focal,now 0.5.8-1focal.20250520.003700 amd64 [installed,automatic]
ros-noetic-rqt-plot/focal,now 0.4.16-1focal.20250520.001122 amd64 [installed,automatic]
ros-noetic-rqt-pose-view/focal,now 0.5.13-1focal.20250520.004134 amd64 [installed,automatic]
ros-noetic-rqt-publisher/focal,now 0.4.12-1focal.20250520.001359 amd64 [installed,automatic]
ros-noetic-rqt-py-common/focal,now 0.5.5-1focal.20250520.000903 amd64 [installed,automatic]
ros-noetic-rqt-py-console/focal,now 0.4.12-1focal.20250519.233341 amd64 [installed,automatic]
ros-noetic-rqt-reconfigure/focal,now 0.5.7-1focal.20250521.003225 amd64 [installed,automatic]
ros-noetic-rqt-robot-dashboard/focal,now 0.5.8-1focal.20250520.003750 amd64 [installed,automatic]
ros-noetic-rqt-robot-monitor/focal,now 0.5.15-1focal.20250520.001222 amd64 [installed,automatic]
ros-noetic-rqt-robot-plugins/focal,now 0.5.8-1focal.20250520.013008 amd64 [installed,automatic]
ros-noetic-rqt-robot-steering/focal,now 0.5.14-1focal.20250519.235722 amd64 [installed,automatic]
ros-noetic-rqt-runtime-monitor/focal,now 0.5.10-1focal.20250519.233325 amd64
[installed,automatic]
ros-noetic-rqt-rviz/focal,now 0.7.2-1focal.20250520.012319 amd64
[installed,automatic]
ros-noetic-rqt-service-caller/focal,now 0.4.12-1focal.20250520.001357 amd64
[installed,automatic]
ros-noetic-rqt-shell/focal,now 0.4.13-1focal.20250519.233725 amd64
[installed,automatic]
ros-noetic-rqt-srv/focal,now 0.4.11-1focal.20250520.001432 amd64
[installed,automatic]
ros-noetic-rqt-tf-tree/focal,now 0.6.5-1focal.20250520.001233 amd64
[installed,automatic]
ros-noetic-rqt-top/focal,now 0.4.11-1focal.20250519.233400 amd64
```

```
[installed,automatic] ros-noetic-rqt-topic/focal,now 0.4.15-1focal.20250519.235732 amd64
[installed,automatic] ros-noetic-rqt-web/focal,now 0.4.11-1focal.20250519.233405 amd64
[installed,automatic] ros-noetic-rqt/focal,now 0.5.5-1focal.20250519.233427 amd64 [installed]
(rosudemty) sakthees-monk@saktheesmonk-desktop:~$
```

ChatGPT said:

Perfect — you've confirmed that **rqt\_gui** and all required RQT packages are installed .

---

Yet `rqt_gui` still gives `command not found`, which means **your shell isn't seeing the ROS binary path properly inside the rosudemty conda environment, even after `source /opt/ros/noetic/setup.bash`**.

Let's fix that step by step:

---

## Step 1: Verify if `rqt_gui` binary exists

Run:

```
ls /opt/ros/noetic/bin/rqt_gui
```

If this outputs the path (and doesn't say "No such file or directory"), then the binary exists — great. If it doesn't exist, something went wrong in the ROS installation. But from your `apt list` output, it's likely fine.

---

## Step 2: Manually source ROS inside Conda env (one-time test)

Run the following **after activating your conda environment**:

```
conda activate rosudemty
source /opt/ros/noetic/setup.bash
echo $PATH | tr ':' '\n' | grep ros
which rqt_gui
```

You should see `/opt/ros/noetic/bin` in the `PATH` and `which rqt_gui` should output:  
`/opt/ros/noetic/bin/rqt_gui`

Then try:

```
rqt_gui
```

If this works, the issue is **just that ROS is not auto-sourced when conda activates**.

---

## Step 3: Make ROS source automatically when conda activates

Create this file (if it doesn't exist):

```
mkdir -p ~/miniconda3/envs/rosudemy/etc/conda/activate.d
nano ~/miniconda3/envs/rosudemy/etc/conda/activate.d/env_vars.sh
```

Paste into it:

```
#!/bin/bash
source /opt/ros/noetic/setup.bash
```

Make it executable:

```
chmod +x ~/miniconda3/envs/rosudemy/etc/conda/activate.d/env_vars.sh
```

Now test:

```
conda deactivate
conda activate rosudemy
which rqt_gui
rqt_gui
```

---

Let me know what `ls /opt/ros/noetic/bin/rqt_gui` returns, and whether `which rqt_gui` works after sourcing!

**You said:**

```
ros-noetic-rqt/focal,now 0.5.5-1focal.20250519.233427 amd64 [installed] (rosudemy) sakthees-monk@saktheesmonk-desktop:~$ ls /opt/ros/noetic/bin/rqt_gui ls: cannot access
'/opt/ros/noetic/bin/rqt_gui': No such file or directory (rosudemy) sakthees-monk@saktheesmonk-desktop:~$
```

**ChatGPT said:**

Thanks — this clears it up.

You have the **ros-noetic-rqt** package installed, but the `rqt_gui` binary is **missing**, meaning:

**Something went wrong with your ros-noetic-rqt package installation.**

Even though `apt` says it's installed, the file `/opt/ros/noetic/bin/rqt_gui` is not present, which it **definitely should be**.

---

### Fix it: Reinstall ros-noetic-rqt and dependencies

Let's force reinstall everything related to RQT:

```
sudo apt update
sudo apt install --reinstall ros-noetic-rqt ros-noetic-rqt-common-plugins
```

This will restore any missing binaries like `rqt_gui`.

Then check again:

```
ls /opt/ros/noetic/bin/rqt_gui
```

and:

```
rqt_gui
```

---

### Optional: Also install all GUI plugins

To be safe, you can install the full `desktop-full` stack (if not already):

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

This includes all GUI tools, `rqt`, `rviz`, and more.

---

### Final Check After Reinstall

Once reinstalled:

```
source /opt/ros/noetic/setup.bash
rqt_gui
```

Let me know what happens after the reinstall.

**You said:**

```
eed to get 6,316 B of archives. After this operation, 0 B of additional disk space will be used. Get:1
http://packages.ros.org/ros/ubuntu focal/main amd64 ros-noetic-rqt amd64 0.5.5-
1focal.20250519.233427 [2,628 B] Get:2 http://packages.ros.org/ros/ubuntu focal/main amd64 ros-
noetic-rqt-common-plugins amd64 0.4.11-1focal.20250521.005128 [3,688 B] Fetched 6,316 B in 1s
(5,701 B/s) (Reading database ... 307890 files and directories currently installed.) Preparing to
unpack .../ros-noetic-rqt_0.5.5-1focal.20250519.233427_amd64.deb ... Unpacking ros-noetic-rqt
(0.5.5-1focal.20250519.233427) over (0.5.5-1focal.20250 519.233427) ... Preparing to unpack
.../ros-noetic-rqt-common-plugins_0.4.11-1focal.20250521.005 128_amd64.deb ... Unpacking ros-
noetic-rqt-common-plugins (0.4.11-1focal.20250521.005128) over (0. 4.11-
1focal.20250521.005128) ... Setting up ros-noetic-rqt-common-plugins (0.4.11-
1focal.20250521.005128) ... Setting up ros-noetic-rqt (0.5.5-1focal.20250519.233427) ...
(rosudemy) sakthees-monk@saktheesmonk-desktop:~$ ls /opt/ros/noetic/bin/rqt_gui ls: cannot
access '/opt/ros/noetic/bin/rqt_gui': No such file or directory (rosudemy) sakthees-
monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui: command not found (rosudemy) sakthees-
monk@saktheesmonk-desktop:~$
```

ChatGPT said:

Thanks — now this is clear:

Even **after reinstalling**, `rqt_gui` is still missing. That's **not normal**, and it suggests:

## ROS is broken due to a partial or minimal install

You only installed `ros-noetic-rqt`, which is **just a launcher interface**, but it **doesn't include the main `rqt_gui` executable** unless its core dependencies are installed.

---

### Solution: Install `rqt_gui` directly with all necessary plugins

Try this command to **install the actual GUI front-end and all its plugins**:

```
sudo apt install ros-noetic-rqt-gui ros-noetic-rqt-gui-py
```

Then check again:

```
ls /opt/ros/noetic/bin/rqt_gui
```

And launch it:

```
rqt_gui
```

---

### Optional (if still broken)

Install all the main RQT tools and plugins:

```
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins ros-noetic-rqt-gui
ros-noetic-rqt-gui-py ros-noetic-rqt-image-view ros-noetic-rqt-robot-monitor
```

---

Let me know what `ls /opt/ros/noetic/bin/rqt_gui` shows after that.

**You said:**

give to check

ChatGPT said:

Sure — here's **a full checklist and command set** to get `rqt_gui` working on your ROS Noetic (Ubuntu 20.04) system:

---

## Step-by-Step Fix for Missing `rqt_gui`

### 1. Install `rqt_gui` and core RQT tools

```
sudo apt update
sudo apt install ros-noetic-rqt-gui ros-noetic-rqt-gui-py
```

### 2. Check if `rqt_gui` is now present

```
ls /opt/ros/noetic/bin/rqt_gui
```

- If this shows the file path, you're good to go.



### 3. Try running `rqt_gui`

```
source /opt/ros/noetic/setup.bash
rqt_gui
```

If it opens, you're done

---

### Optional: Install All RQT Common Plugins (recommended)

```
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins
```

This includes useful tools like:

- `rqt_graph`
  - `rqt_plot`
  - `rqt_console`
  - `rqt_image_view`
- 

### Final Test Commands

```
which rqt_gui
roscd rqt_gui
roslaunch rqt_gui rqt_gui
```

---

Let me know what output you get for:

```
ls /opt/ros/noetic/bin/rqt_gui
```

And whether `rqt_gui` opens successfully after these.

**You said:**

```
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded. (rosudemey) sakthees-
monk@saktheesmonk-desktop:~$ ls /opt/ros/noetic/bin/rqt_gui ls: cannot access
'/opt/ros/noetic/bin/rqt_gui': No such file or directory (rosudemey) sakthees-monk@saktheesmonk-
desktop:~$
```

**ChatGPT said:**

Thanks. That confirms the issue clearly: **`rqt_gui` is not actually installed properly** even though it shows as installed via APT.

Let's **fix it properly step-by-step**:

---

### Step 1: Manually reinstall `rqt_gui`

Run this:

```
sudo apt install --reinstall ros-noetic-rqt-gui
```

Then check again:

```
ls /opt/ros/noetic/bin/rqt_gui
```

---

## Step 2: Install full RQT suite (in case some parts are missing)

Run this just to be sure:

```
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins
```

---

## Step 3: Check where rqt\_gui actually is

If after reinstall you **still don't find it in /opt/ros/noetic/bin**, try this:

```
dpkg -L ros-noetic-rqt-gui | grep rqt_gui
```

This will show the actual file path installed by the package.

---

## Step 4: Run using ROS

```
source /opt/ros/noetic/setup.bash
roslaunch rqt_gui rqt_gui
```

Let me know the output of:

```
dpkg -L ros-noetic-rqt-gui | grep rqt_gui
```

and whether `roslaunch rqt_gui rqt_gui` works.

**You said:**

```
rosudemy) sakthees-monk@saktheesmonk-desktop:~$ roslaunch rqt_gui rqt_gui
RosPluginProvider.load(qt_gui_cpp/CppPluginProvider) exception raised in
__builtin__.__import__(qt_gui_cpp.cpp_plugin_provider, [CppPluginProvider]): Traceback (most
recent call last): File "/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/ros_plugin_provider.py",
line 79, in load module = __builtin__.__import__(File
"/opt/ros/noetic/lib/python3/dist-packages/qt_gui_cpp/cpp_plugin_provider.py", line 33, in
<module> from .cpp_binding_helper import qt_gui_cpp File "/opt/ros/noetic/lib/python3/dist-
packages/qt_gui_cpp/cpp_binding_helper.py", line 43, in <module> from . import
libqt_gui_cpp_sip ValueError: PyCapsule_GetPointer called with incorrect name
RecursivePluginProvider.discover() loading plugin "qt_gui_cpp/CppPluginProvider" failed:
Traceback (most recent call last): File
"/opt/ros/noetic/lib/python3/dist-packages/qt_gui/recursive_plugin_provider.py", line 60, in
discover instance = self._plugin_provider.load(plugin_descriptor.plugin_id(), None) File
"/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/ros_plugin_provider.py", line 90, in load raise e
File "/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/ros_plugin_provider.py", line 79, in load
module = __builtin__.__import__(File
"/opt/ros/noetic/lib/python3/dist-packages/qt_gui_cpp/cpp_plugin_provider.py", line 33, in
```

```
<module> from .cpp_binding_helper import qt_gui_cpp File "/opt/ros/noetic/lib/python3/dist-
packages/qt_gui_cpp/cpp_binding_helper.py", line 43, in <module> from . import
libqt_gui_cpp_sip ValueError: PyCapsule_GetPointer called with incorrect name (rosudemy)
sakthees-monk@saktheesmonk-desktop:~$ dpkg -L ros-noetic-rqt-gui | grep rqt_gui
/opt/ros/noetic/lib/pkgconfig/rqt_gui.pc /opt/ros/noetic/lib/python3/dist-packages/rqt_gui
/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/__init__.py /opt/ros/noetic/lib/python3/dist-
packages/rqt_gui/__pycache__
/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/__pycache__/__init__.cpython-38.pyc
/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/__pycache__/main.cpython-38.pyc
/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/__pycache__/ros_plugin_provider.cpython-38.pyc
/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/__pycache__/rospkg_plugin_provider.cpython-
38.pyc /opt/ros/noetic/lib/python3/dist-packages/rqt_gui/main.py /opt/ros/noetic/lib/python3/dist-
packages/rqt_gui/ros_plugin_provider.py
/opt/ros/noetic/lib/python3/dist-packages/rqt_gui/rospkg_plugin_provider.py
/opt/ros/noetic/lib/python3/dist-packages/rqt_gui-0.5.5.egg-info /opt/ros/noetic/lib/python3/dist-
packages/rqt_gui-0.5.5.egg-info/PKG-INFO /opt/ros/noetic/lib/python3/dist-packages/rqt_gui-
0.5.5.egg-info/dependency_links.txt /opt/ros/noetic/lib/python3/dist-packages/rqt_gui-0.5.5.egg-
info/top_level.txt /opt/ros/noetic/lib/rqt_gui /opt/ros/noetic/lib/rqt_gui/rqt_gui
/opt/ros/noetic/share/rqt_gui /opt/ros/noetic/share/rqt_gui/cmake
/opt/ros/noetic/share/rqt_gui/cmake/rqt_guiConfig-version.cmake
/opt/ros/noetic/share/rqt_gui/cmake/rqt_guiConfig.cmake /opt/ros/noetic/share/rqt_gui/package.xml
/opt/ros/noetic/share/rqt_gui/resource /opt/ros/noetic/share/rqt_gui/resource/rqt.png (rosudemy)
sakthees-monk@saktheesmonk-desktop:~$
```

ChatGPT said:

Thank you — you've narrowed it down perfectly.

### Summary of Problem:

You're getting this error:

ValueError: PyCapsule\_GetPointer called with incorrect name

This is a **known issue** that occurs when **rqt** or **qt\_gui\_cpp** is being run inside a **Conda environment**, and there is a **mismatch between Python versions or SIP bindings**.

---

### Root Cause:

ROS Noetic is tightly coupled to **system Python 3.8** and specific system-wide SIP bindings. Conda's Python interpreter and libraries (like **sip**) conflict with them.

---

### Working Solution:

**You should run `rqt_gui` *outside* the Conda environment.**

---

## Steps to run correctly:

### 1. Deactivate Conda environment:

- `conda deactivate`

### • Source ROS:

- `source /opt/ros/noetic/setup.bash`

### • Run `rqt_gui`:

3. `rqt_gui`

This should launch the GUI without the Python capsule error.

---

## Why this works:

ROS Noetic uses `/usr/bin/python3` and expects system-wide Python modules like `sip` to behave in a certain way. Conda replaces or masks these, causing runtime failures like `PyCapsule_GetPointer` mismatch.

---

## Optional: If you must run Python scripts in Conda (e.g., ML code), do it without launching GUI tools like `rqt` inside Conda.

Let me know if you'd like to set up a workflow where ROS and Conda co-exist cleanly!

### You said:

```
akthees-monk@saktheesmonk-desktop:~$ source /opt/ros/noetic/setup.bash sakthees-monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui: command not found sakthees-monk@saktheesmonk-desktop:~$ install without conda
```

### ChatGPT said:

Perfect — since you want to **install `rqt_gui` without using Conda**, here's how to do it correctly for **ROS Noetic on Ubuntu 20.04**:

---

## Step-by-step: Install `rqt_gui` (System-Wide)

### 1. Ensure your system ROS environment is sourced (you already did):

- `source /opt/ros/noetic/setup.bash`

### • Install all RQT core tools:

- `sudo apt update`

```
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins
```

- Confirm installation:

- `which rqt_gui`

```
Expected output: /opt/ros/noetic/bin/rqt_gui
```

- **Now run:**

```
4. rqt_gui
```

---

### **If rqt\_gui still says "command not found"**

Double-check the folder:

```
ls /opt/ros/noetic/bin | grep rqt
```

You should see `rqt_gui` listed.

Also try:

```
roslaunch rqt_gui rqt_gui
```

If it still doesn't work, please send the output of:

```
apt list --installed | grep rqt
```

This will confirm if the packages are installed correctly.

---

Let me know once you've tried the install — I'll help you troubleshoot if anything's missing.

**You said:**

```
sakthees-monk@saktheesmonk-desktop:~$ ls /opt/ros/noetic/bin | grep rqt rqt rqt_bag rqt_console
rqt_dep rqt_graph rqt_image_view rqt_logger_level rqt_plot rqt_shell rqt_topic sakthees-mon
```

ChatGPT said:

Thanks! Based on this output:

```
ls /opt/ros/noetic/bin | grep rqt
rqt
rqt_bag
rqt_console
rqt_dep
rqt_graph
rqt_image_view
rqt_logger_level
rqt_plot
rqt_shell
rqt_topic
```

This means **rqt\_gui** itself is missing, but many individual RQT plugins are installed.

---

### **Fix: Install the core rqt\_gui package directly**

Run this:

```
sudo apt install ros-noetic-rqt-gui
```

Then verify again:

```
ls /opt/ros/noetic/bin | grep rqt_gui
```

You should now see:

```
rqt_gui
```

---

### Then run:

```
rqt_gui
```

Let me know the result. If it launches a GUI window, it's working!

**You said:**

```
t_topic sakthees-monk@saktheesmonk-desktop:~$ sudo apt install ros-noetic-rqt-gui Reading
package lists... Done Building dependency tree Reading state information... Done ros-noetic-rqt-gui
is already the newest version (0.5.5-1focal.20250519.232427). The following packages were
automatically installed and are no longer required: chromium-codecs-ffmpeg-extra gstreamer1.0-
vaapi lib32gcc-s1 libc6-i386 libgstreamer-plugins-bad1.0-0 linux-headers-5.15.0-67-generic linux-
hwe-5.15-headers-5.15.0-67 linux-image-5.15.0-67-generic linux-modules-5.15.0-67-generic linux-
modules-extra-5.15.0-67-generic Use 'sudo apt autoremove' to remove them. 0 upgraded, 0 newly
installed, 0 to remove and 1 not upgraded. sakthees-monk@saktheesmonk-desktop:~$ ls
/opt/ros/noetic/bin | grep rqt_gui sakthees-monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui:
command not found sakthees-monk@saktheesmonk-desktop:~$ its ubuntu
```

**ChatGPT said:**

Thanks for the update. You're on **Ubuntu** and **ROS Noetic**, and even though `ros-noetic-rqt-gui` is *already installed*, the binary `rqt_gui` is **still missing** from `/opt/ros/noetic/bin`.

This usually indicates one of the following:

---

### Step-by-step Fix:

#### 1. Make sure ROS paths are correctly sourced

Before testing anything, run:

```
source /opt/ros/noetic/setup.bash
```

Then try:

```
which rqt_gui
```

If that still says `rqt_gui: command not found`, then continue below.

---

#### 2. Try installing full rqt metapackage

Sometimes, the individual `rqt_gui` binary is included under a broader metapackage. Run:

```
sudo apt install ros-noetic-rqt
```

Then:

```
source /opt/ros/noetic/setup.bash
which rqt_gui
```

---

### 3. If still not found: Check Python environment interference

If you installed ROS **outside Conda**, but you're still using a Conda environment ((rosudemy) prompt indicates this), Python-related scripts may not be visible.

To check without Conda interference:

```
conda deactivate
```

Then try:

```
source /opt/ros/noetic/setup.bash
rqt_gui
```

---

### 4. Optional: Force reinstall just in case it's broken

```
sudo apt install --reinstall ros-noetic-rqt-gui
```

---

## Summary of Commands

```
conda deactivate
source /opt/ros/noetic/setup.bash
sudo apt install ros-noetic-rqt
which rqt_gui
rqt_gui
```

Let me know what happens after this.

**You said:**

```
se 'sudo apt autoremove' to remove them. 0 upgraded, 0 newly installed, 0 to remove and 1 not
upgraded. sakthees-monk@saktheesmonk-desktop:~$ ls /opt/ros/noetic/bin | grep rqt_gui sakthees-
monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui: command not found sakthees-
monk@saktheesmonk-desktop:~$ source /opt/ros/noetic/setup.bash sakthees-monk@saktheesmonk-
desktop:~$ which rqt_gui sakthees-monk@saktheesmonk-desktop:~$ sudo apt install ros-noetic-rqt
Reading package lists... Done Building dependency tree Reading state information... Done ros-
noetic-rqt is already the newest version (0.5.5-1focal.20250519.233427). The following packages
were automatically installed and are no longer required: chromium-codecs-ffmpeg-extra
gstreamer1.0-vaapi lib32gcc-s1 libc6-i386 libgstreamer-plugins-bad1.0-0 linux-headers-5.15.0-67-
generic linux-hwe-5.15-headers-5.15.0-67 linux-image-5.15.0-67-generic linux-modules-5.15.0-67-
generic linux-modules-extra-5.15.0-67-generic Use 'sudo apt autoremove' to remove them. 0
upgraded, 0 newly installed, 0 to remove and 1 not upgraded. sakthees-monk@saktheesmonk-
desktop:~$ source /opt/ros/noetic/setup.bash sakthees-monk@saktheesmonk-desktop:~$ which
rqt_gui sakthees-monk@saktheesmonk-desktop:~$
```

ChatGPT said:

You're trying to run `rqt_gui`, but even after installing `ros-noetic-rqt`, the command is still not found and `which rqt_gui` returns nothing. Let's troubleshoot step-by-step.

---

## 1. Check if `rqt_gui` binary exists

You already tried:

```
ls /opt/ros/noetic/bin | grep rqt_gui
```

It returned nothing — this means the binary is not present in the expected location.

---

## 2. Make sure all required `rqt` packages are installed

Install **these three key packages**:

```
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins ros-noetic-rqt-gui
```

---

## 3. Update environment and try again

After installing, run:

```
source /opt/ros/noetic/setup.bash
```

Then try:

```
which rqt_gui
```

If it returns something like `/opt/ros/noetic/bin/rqt_gui`, you're good to go.

Then run:

```
rqt_gui
```

---

## 4. If it still fails, check your Python environment

Sometimes `rqt_gui` fails if ROS is not using the correct Python version (Noetic uses Python 3).

Run:

```
python3 -m rqt_gui
```

Or:

```
roslaunch rqt_gui rqt_gui
```

---



## 5. Check PATH manually

Try adding this to your shell config (~/.bashrc):

```
source /opt/ros/noetic/setup.bash
export PATH=$PATH:/opt/ros/noetic/bin
```

Then:

```
source ~/.bashrc
which rqt_gui
```

---

Let me know the output of:

```
roslaunch rqt_gui rqt_gui
```

If it still doesn't work, I can walk you through checking if the `rqt_gui` Python module exists and is executable.

**You said:**

```
linux-hwe-5.15-headers-5.15.0-67 linux-image-5.15.0-67-generic linux-modules-5.15.0-67-generic
linux-modules-extra-5.15.0-67-generic Use 'sudo apt autoremove' to remove them. 0 upgraded, 0
newly installed, 0 to remove and 1 not upgraded. sakthees-monk@saktheesmonk-desktop:~$ source
/opt/ros/noetic/setup.bash sakthees-monk@saktheesmonk-desktop:~$ which rqt_gui sakthees-
monk@saktheesmonk-desktop:~$ rqt_gui rqt_gui: command not found sakthees-
monk@saktheesmonk-desktop:~$ python3 -m rqt_gui /usr/bin/python3: No module named
rqt_gui.__main__; 'rqt_gui' is a package and cannot be directly executed sakthees-
monk@saktheesmonk-desktop:~$
```

**ChatGPT said:**

It looks like `rqt_gui` is **not installed** correctly or **not available in your system's path**, even though `ros-noetic-rqt` is installed. Here's how you can **fix this step-by-step**:

---

### Step 1: Make Sure All RQT Plugins Are Installed

You should install the **complete RQT suite** to ensure all plugins (including `rqt_gui`) are included.

Run this:

```
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins
```

---

### Step 2: Check Again for `rqt_gui` Binary

After installation, run:

```
source /opt/ros/noetic/setup.bash
which rqt_gui
```

It should return something like:

```
/opt/ros/noetic/bin/rqt_gui
```

Then run:

```
rqt_gui
```

---

### Step 3: Fix Python Path Issues (If Any)

Sometimes ROS uses its own environment. Try activating the ROS Python environment explicitly:

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

Then test again:

```
which rqt_gui
rqt_gui
```

---

### Optional Debug Step: Check If Binary Exists

Try this:

```
ls /opt/ros/noetic/lib/rqt_gui
```

Or:

```
find /opt/ros/noetic -name rqt_gui
```

If the binary isn't there, installing `ros-noetic-rqt` and `rqt-common-plugins` again should fix it.

---

Let me know the output of these commands if you're still stuck, and I'll guide you further.

**You said:**

so ros gui and some only works with native python good

ChatGPT said:

Yes, exactly — you've got it right. Here's the clarification:

---

### ROS (Noetic) GUI tools like `rqt_gui` work best with the *native system Python* (not Conda)

**Why?**

1. **ROS Noetic** is built to work with **Python 3.8** (Ubuntu 20.04's default), and it **installs its tools in system paths** (like `/opt/ros/noetic/...`).
2. Tools like `rqt_gui`, `rqt_graph`, and plugins rely on:
  - `rospkg`, `qt_gui`, `catkin_pkg`, etc.

- ROS-specific environment variables like `ROS_PACKAGE_PATH` and correct `PYTHONPATH`.

### 3. Conda isolates environments, which:

- Hides or overrides `/opt/ros/noetic/lib/python3/dist-packages`
- Breaks CMake builds or prevents GUIs like `rqt_gui` from being found.

---

## Best Practice for GUI & Tooling (ROS)

| Use Case                                                               | Recommended Setup                                                                                                                    |
|------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| Run <code>rqt_gui</code> ,<br><code>rqt_bag</code> , <code>rviz</code> | Use system shell with <code>source /opt/ros/noetic/setup.bash</code>                                                                 |
| Writing ROS Python nodes                                               | Use system Python ( <code>/usr/bin/python3</code> )                                                                                  |
| Building with <code>catkin_make</code>                                 | Use terminal without Conda                                                                                                           |
| Using Conda for ML/CV                                                  | You can still use Conda for <i>non-ROS</i> tasks like TensorFlow, OpenCV, PyTorch etc. — just <b>keep it separate</b> from ROS tools |

---

## Bonus Tip: Switch Easily with an Alias

You can define an alias in `~/ .bashrc`:

```
alias rosenv='source /opt/ros/noetic/setup.bash'
```

Now just run:

```
rosenv
rqt_gui
```

---

Let me know if you want a hybrid setup — e.g., using Conda for ML but still talking to ROS. It's possible with workarounds like `PYTHONPATH` bridging or `catkin build` customizations.

### You said:

ok for arduino serial communication and raspberry pi and slam lidar and all cheap to costly peripherals install serial; radio lora ,zigbee,bluetooth ,wifi all communication in native python

ChatGPT said:

Yes — for **Arduino, Raspberry Pi, SLAM, LiDAR, LoRa, ZigBee, Bluetooth, WiFi**, and similar **hardware + communication modules, using native Python** (system Python or Python3 installed via apt) is:

---

## The Best & Most Stable Option

---

# Why Native Python Is Best for Hardware Communication:

| Reason                                       | Explanation                                                                                                                                                                                                                                                                                                    |
|----------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>1. Kernel &amp; Drivers Compatibility</b> | Libraries like <code>pyserial</code> , <code>RPi.GPIO</code> , <code>smbus</code> , <code>spidev</code> , <code>bluetooth</code> , and <code>socket</code> are tightly coupled with Linux device files (like <code>/dev/ttyUSB0</code> , <code>/dev/i2c-1</code> , etc.) and work smoothly with native Python. |
| <b>2. System-wide Udev Rules</b>             | USB/Serial rules (e.g., for <code>/dev/ttyUSB0</code> ) are often mapped in <code>/etc/udev/</code> which native Python can access without Conda's path isolation.                                                                                                                                             |
| <b>3. ROS/SLAM tools</b>                     | Tools like <b>RTAB-Map</b> , <b>Hector SLAM</b> , <b>Cartographer</b> , and <b>serial SLAM modules</b> expect system libraries and often won't detect hardware correctly from inside Conda.                                                                                                                    |
| <b>4. Compilation of C/C++ Extensions</b>    | Many device libraries (e.g., for LIDAR or IMUs) have C/C++ backend modules that compile with <code>g++</code> and <code>python3-dev</code> . These align with native Python.                                                                                                                                   |
| <b>5. GUI Tools &amp; Monitoring</b>         | Tools like <code>minicom</code> , <code>screen</code> , <code>roserial_python</code> , <code>arduino-cli</code> , <code>rqt_plot</code> , and even custom serial dashboards prefer native environments.                                                                                                        |

## Common Libraries You'll Use (Best on Native Python)

| Peripheral                                | Python Library                                                     | Notes                                            |
|-------------------------------------------|--------------------------------------------------------------------|--------------------------------------------------|
| <b>Arduino Serial</b>                     | <code>pyserial</code>                                              | Reliable on native                               |
| <b>LoRa (e.g. SX127x)</b>                 | <code>pySX127x</code> or <code>serial</code> + manual              | Needs access to <code>/dev/ttyUSB*</code>        |
| <b>Zigbee</b>                             | <code>xbee</code> or <code>pyserial</code>                         | Native GPIO + serial often needed                |
| <b>Bluetooth (HC-05 / BLE)</b>            | <code>bluetooth</code> , <code>pybluez</code> , <code>bleak</code> | Native required for DBus, BlueZ                  |
| <b>WiFi Modules (ESP, etc.)</b>           | <code>socket</code> , <code>requests</code>                        | No issue, but low-level config better in native  |
| <b>Raspberry Pi GPIO / I2C / SPI</b>      | <code>RPi.GPIO</code> , <code>gpiozero</code> , <code>smbus</code> | Conda isolates device tree paths                 |
| <b>SLAM with LIDAR (RPLIDAR, YDLIDAR)</b> | <code>rplidar</code> , <code>ydlidar_ros</code> , ROS nodelets     | Highly dependent on native ROS & hardware access |
| <b>Serial Cameras, GPS</b>                | <code>pyserial</code> , <code>gpsd</code> , <code>cv2</code>       | Camera nodes need V4L2 and <code>/dev/</code>    |
| <b>IMU / Sensors (MPU6050, BMP280)</b>    | <code>smbus</code> , <code>i2c-tools</code>                        | Native bus access                                |

### Conda is Great for:

- Machine Learning (TensorFlow, PyTorch)
- Jupyter
- OpenCV + AI

- Simulation (Gazebo plugins in isolated ML tasks)

But **hardware control & serial peripherals are NOT reliable in Conda unless heavily bridged** (which is advanced and fragile).

---

## Best Practice Summary

| Task                        | Use                                                                                                          |
|-----------------------------|--------------------------------------------------------------------------------------------------------------|
| Arduino, LoRa, ZigBee, SLAM | <b>Native Python (apt install, system python)</b>                                                            |
| ROS + SLAM + LIDAR          | <b>Native Python + ROS setup.bash</b>                                                                        |
| TensorFlow, OpenCV          | Conda (for ML only)                                                                                          |
| ROS + ML combo              | Use native Python for ROS core, and let ROS nodes call Conda apps via socket/pipe if needed (advanced setup) |

---

If you want a **device communication checklist or test code for each peripheral**, I can prepare one for:

- LoRa
- Bluetooth
- SLAM with RPLIDAR
- Arduino + ROS
- Zigbee (XBee)
- Camera streaming

Let me know!

**You said:**

give all above to install gmap viwer turtle sim visual blocvk programming in rios and all setup to install

**ChatGPT said:**

Here's a full setup guide to install **ROS Noetic** with:

- `turtlesim`
  - `gmapping`
  - `rqt`, `rviz`, `rqt_graph`, etc.
  - visual block programming (like Blockly for ROS)
  - All dependencies and tools for serial, LoRa, Zigbee, Bluetooth, WiFi communication
  - On **native Python 3 (Python 3.8+)**
-

## STEP 1: Basic ROS Desktop-Full Install

```
sudo apt update
sudo apt install ros-noetic-desktop-full
```

---

## STEP 2: ROS Environment Setup

Add to ~/.bashrc:

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

---

## STEP 3: Install rosdep

```
sudo apt install python3-rosdep
sudo rosdep init
rosdep update
```

---

## STEP 4: Essential ROS Packages

```
sudo apt install ros-noetic-turtlesim \
ros-noetic-slam-gmapping \
ros-noetic-rqt \
ros-noetic-rqt-common-plugins \
ros-noetic-rviz \
ros-noetic-rqt-graph \
ros-noetic-rqt-robot-plugins \
ros-noetic-navigation \
ros-noetic-map-server
```

---

## STEP 5: TurtleBot / Lidar / GMapping Compatible Simulation Setup

```
sudo apt install ros-noetic-turtlebot3 \
ros-noetic-turtlebot3-simulations \
ros-noetic-turtlebot3-msgs
```

Add to bashrc:

```
echo "export TURTLEBOT3_MODEL=burger" >> ~/.bashrc
source ~/.bashrc
```

---

## STEP 6: Install Visual Programming for ROS (Blockly)

### Option A: [Blockly for ROS](#)

```
sudo apt install git npm
git clone https://github.com/JdeRobot/blockly
cd blockly
npm install
npm start
```

You can control ROS nodes from web-based visual blocks.

---

## STEP 7: Python Packages for Peripheral Communication

```
Serial (Arduino, USB)
pip3 install pyserial

Bluetooth
sudo apt install bluetooth bluez
pip3 install pybluez

WiFi sockets
pip3 install socket

Zigbee (XBee)
pip3 install digi-xbee

LoRa (via UART)
pip3 install pyserial

MQTT for WiFi/Bluetooth/LoRa communication
pip3 install paho-mqtt

WebSocket
pip3 install websockets

I2C, SPI, GPIO for Raspberry Pi
sudo apt install python3-smbus python3-rpi.gpio i2c-tools
```

---

## STEP 8: Install Arduino IDE (for flashing sensors)

```
sudo apt install arduino
```

Use `/dev/ttyUSB0` or `/dev/ttyACM0` with `pyserial`.

---

## STEP 9: Example Run for Verification

```
roscore # In one terminal

In another terminal
roslaunch turtlesim turtlesim_node

In another terminal
roslaunch rqt_graph rqt_graph
```

You should see GUI windows popping up.

---

Would you like a **ready-made ROS workspace** zipped with all these installed? Or shall I give you a shell script (`.sh`) that installs everything in one go?

**You said:**

node-sorted-object node-spdx-correct node-spdx-exceptions node-spdx-expression-parse node-spdx-license-ids node-sshpk node-ssri node-stream-each node-stream-iterate node-stream-shift node-strict-uri-encode node-string-decoder node-string-width node-strip-ansi node-strip-eof node-strip-json-comments node-supports-color node-tar node-term-size node-text-table node-through node-through2 node-timed-out node-tough-cookie node-tunnel-agent node-tweetnacl node-typedarray node-typedarray-to-buffer node-uid-number node-unique-filename node-unique-string node-unpipe node-uri-js node-url-parse-lax node-url-to-options node-util-deprecate node-uuid node-validate-npm-package-license node-validate-npm-package-name node-verror node-wcwidth.js node-which node-which-module node-wide-align node-widest-line node-wrap-ansi node-wrappy node-write-file-atomic node-xdg-basedir node-xtend node-y18n node-yallist node-yargs node-yargs-parser nodejs nodejs-doc npm python-pkg-resources python2 python2-minimal python2.7 python2.7-minimal 0 upgraded, 299 newly installed, 0 to remove and 1 not upgraded. Need to get 20.4 MB of archives. After this operation, 115 MB of additional disk space will be used. Do you want to continue? [Y/n] y Abort. sakthees-monk@saktheesmonk-desktop:~\$

**ChatGPT said:**

To fully set up **ROS Noetic** with all key tools like **Turtlesim**, **SLAM tools**, **GMapping**, **LoRa**, **Zigbee**, **Bluetooth**, **Wi-Fi communication**, and **visual block programming tools**, here's a comprehensive native Python & system-wide installation guide for Ubuntu 20.04 (recommended for ROS Noetic):

---

## 1. Basic ROS Noetic Setup

```
sudo apt update
sudo apt upgrade -y

ROS Noetic full desktop version
sudo apt install ros-noetic-desktop-full -y

Environment setup
echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc
source ~/.bashrc

ROS Dependencies
sudo apt install python3-rosdep python3-rosinstall python3-rosinstall-generator
python3-wstool build-essential -y

Initialize rosdep
sudo rosdep init
rosdep update
```

---

## 2. Turtlesim + GMapping + SLAM

```
Turtlesim (demo)
sudo apt install ros-noetic-turtlesim -y

SLAM tools
sudo apt install ros-noetic-slam-gmapping ros-noetic-hector-slam -y
```



```
Gmapping RViz and teleop keyboard
sudo apt install ros-noetic-teleop-twist-keyboard ros-noetic-rviz -y
```

---

### 3. Visualization Tools

```
RQT GUI tools
sudo apt install ros-noetic-rqt ros-noetic-rqt-common-plugins ros-noetic-rqt-robot-plugins -y

RViz 3D visualization
sudo apt install ros-noetic-rviz -y
```

---

## 4. Serial, LoRa, Zigbee, Bluetooth, WiFi Communication

### General Communication Libraries (Native Python)

```
Serial (for Arduino, GPS, etc.)
sudo apt install python3-serial -y

Bluetooth
sudo apt install bluetooth bluez bluez-tools rfkill -y
sudo apt install python3-bluez -y

LoRa (via USB serial usually)
Use serial or custom SPI driver depending on hardware

Zigbee (XBee modules)
sudo apt install python3-xbee -y

WiFi (socket level comms)
sudo apt install python3-socket -y
```

For hardware like **LoRa E32**, **nRF24L01**, **HC-05**, **ESP-01**, use `pyserial`, `pybluez`, `spidev`, and `socket` programming depending on interface.

---

## 5. Arduino Integration

```
Install Arduino IDE
sudo apt install arduino arduino-core -y

Install roserial for ROS-Arduino communication
sudo apt install ros-noetic-roserial ros-noetic-roserial-arduino ros-noetic-roserial-python -y
```

---

## 6. Visual Programming (Blockly / Node-RED)

### Option A: Blockly for ROS

```
Blockly integration (unofficial tools)
cd ~
git clone https://github.com/gramaziokohler/ros_blockly.git
cd ros_blockly
chmod +x ros_blockly.sh
./ros_blockly.sh
```

### Option B: Node-RED ROS Bridge

```
Install Node.js and npm
sudo apt install nodejs npm -y

Install Node-RED
sudo npm install -g --unsafe-perm node-red

Run Node-RED
node-red

ROS Node-RED bridge
cd ~/.node-red
npm install node-red-contrib-ros
```

Visit <http://localhost:1880> to open visual editor.

---

## 7. TurtleBot & Navigation Stack (Optional Advanced)

```
sudo apt install ros-noetic-turtlebot3* ros-noetic-navigation -y
```

---

## 8. Raspberry Pi and GPIO Support (optional)

```
Raspberry Pi GPIO library (native testing on Pi)
sudo apt install python3-rpi.gpio -y
```

---

## 9. Testing

### Launch turtlesim:

```
roscore
In another terminal
roslaunch turtlesim turtlesim_node
roslaunch turtlesim turtle_teleop_key
```

### Launch RViz and Gmapping:

```
roslaunch turtlebot3_slam turtlebot3_slam.launch
```

---

If anything fails or shows `Abort .`, rerun the `apt install` with `sudo` and confirm with `y`.

Would you like me to give you a `.sh` script to install **all the above at once**?

**You said:**

```
Blockly integration (unofficial tools) cd ~ git clone
https://github.com/gramaziokohler/ros_blockly.git cd ros_blockly chmod +x ros_blockly.sh
./ros_blockly.sh without user name clone
```

ChatGPT said:

You're installing **ROS Blockly**, a visual block programming interface for ROS. The `ros_blockly.sh` script automates setup. Here's how to do it **without needing a username**, assuming you're logged in as your normal user (`sakthees-monk`) and your `~` (home) directory is writable:

### Complete Steps to Clone and Run `ros_blockly`:

```
cd ~
git clone https://github.com/gramaziokohler/ros_blockly.git
cd ros_blockly
chmod +x ros_blockly.sh
./ros_blockly.sh
```

---

#### What this does:

- `git clone` brings the repo from GitHub.
- `chmod +x` makes the shell script executable.
- `./ros_blockly.sh` runs the install script, which:
  - Sets up ROS environment.
  - Installs required packages (`roslaunch`, `roscpp`, etc.).
  - Runs a Blockly web interface that connects to ROS.

---

#### If `ros_blockly.sh` asks for ROS workspace:

If needed, create and build a ROS workspace:

```
mkdir -p ~/ros_blockly_ws/src
cd ~/ros_blockly_ws/src
catkin_init_workspace
cd ..
catkin_make
source devel/setup.bash
```

---

## After setup:

Run the web interface (from inside the `ros_blockly` directory):

```
roslaunch ros_blockly ros_blockly.launch
```

Then open your browser and go to:

`http://localhost:9090`

---

Let me know if you also want to add support for:

- `turtlesim`
- `gmapping` / SLAM
- LoRa/ZigBee/Serial/Bluetooth integration in Python
- Raspberry Pi + Arduino ROS serial setup

### You said:

```
git clone https://github.com/gramaziokohler/ros_blockly.git Cloning into 'ros_blockly'... Username
for 'https://github.com': Password for 'https://cd%20ros_blockly%0Dchmod%20%2Bx
%20ros_blockly.sh%0D.%2Fros_blockly.sh%0D@github.com': [2]+ Stopped git clone
https://github.com/gramaziokohler/ros_blockly.git sakthees-monk@saktheesmonk-desktop:~$ cd ~
sakthees-monk@saktheesmonk-desktop:~$ git clone
https://github.com/gramaziokohler/ros_blockly.git fatal: destination path 'ros_blockly' already exists
and is not an empty directory. sakthees-monk@saktheesmonk-desktop:~$ cd ros_blockly sakthees-
monk@saktheesmonk-desktop:~/ros_blockly$ chmod +x ros_blockly.sh chmod: cannot access
'ros_blockly.sh': No such file or directory
sakthees-monk@saktheesmonk-desktop:~/ros_blockly$./ros_blockly.sh bash: ./ros_blockly.sh: No
such file or directory sakthees-monk@saktheesmonk-desktop:~/ros_blockly$
```

### ChatGPT said:

It looks like the clone operation was **interrupted** or **partially failed**, and as a result, the `ros_blockly.sh` file is **missing** in the `ros_blockly` directory.

Let's fix this step-by-step:

---

## Step 1: Delete the incomplete folder

Since the clone failed, remove the corrupted/incomplete repo:

```
rm -rf ~/ros_blockly
```

---

## Step 2: Re-clone the repository (no username or password needed)

You must clone using the **HTTPS read-only link** or switch to SSH if you're using GitHub authentication. Use this **correct read-only link**:

```
git clone https://github.com/gramaziokohler/ros_blockly.git
```

**Do not use a link with a user or `cd@. . .` in it.**

---

### Step 3: Run the script

```
cd ros_blockly
chmod +x ros_blockly.sh
./ros_blockly.sh
```

---

### Bonus Tip – Use SSH for GitHub (to avoid password prompts)

If you plan to work with GitHub often:

1. Generate SSH key:
  - `ssh-keygen -t rsa -b 4096 -C "your_email@example.com"`
  - Add it to GitHub:  
Copy your key:
  - `cat ~/.ssh/id_rsa.pub`

Then go to GitHub → Settings → SSH and GPG Keys → Add new.

- Clone using:
3. `git clone git@github.com:gramaziokohler/ros_blockly.git`
- 

Once it runs, I'll help you launch it and connect it to other ROS packages like `turtlesim`, `SLAM`, or sensors.

**You said:**

```
git clone https://github.com/gramaziokohler/ros_blockly.git fatal: could not create work tree dir
'ros_blockly': No such file or directory sakthees-monk@saktheesmonk-desktop:~/ros_blockly$
```

ChatGPT said:

The error:

```
fatal: could not create work tree dir 'ros_blockly': No such file or directory
```

means your current directory (`~/ros_blockly`) **already exists**, and **you're trying to clone into a directory with the same name**, causing a conflict.

---

### Fix It Step-by-Step

#### Step 1: Delete the broken or partially cloned folder

You must first remove the existing `ros_blockly` directory:

```
rm -rf ~/ros_blockly
```

### Step 2: Clone again cleanly

Now clone the repo again from your **home directory**:

```
cd ~
git clone https://github.com/gramaziokohler/ros_blockly.git
```

### Step 3: Run the install script

```
cd ros_blockly
chmod +x ros_blockly.sh
./ros_blockly.sh
```

---

If this works, and you want to proceed with:

- `turtlesim`
- `gmapping`
- Blockly Visual Programming
- ROS GUI + RViz

Let me know and I'll guide you through the full setup, including launching blocks and ROS bridge connections.

**You said:**

```
git clone https://github.com/gramaziokohler/ros_blockly.git Cloning into 'ros_blockly'... Username
for 'https://github.com':
```

**ChatGPT said:**

You're seeing this prompt:

```
Username for 'https://github.com':
```

because you're trying to clone the repository using **HTTPS**, and GitHub **requires authentication** for HTTPS cloning if the repository has been restricted (private or deprecated setup). But this repository is public, so this shouldn't happen normally.

---

## Solution Options

### Option 1: Use GitHub's Anonymous Read-Only HTTPS (no login needed)

Just use the correct URL without login. But GitHub has moved many things to redirect or require tokens in some older setups.

Run these commands exactly:

```
cd ~
rm -rf ros_blockly # remove any broken directory
git clone https://github.com/gramaziokohler/ros_blockly.git
```

If this **still asks for username**, go with SSH clone instead.

---

### Option 2: Use SSH Clone (Best for developers)

#### 1. Generate SSH key (if not already):

```
ssh-keygen -t ed25519 -C "your_email@example.com"
```

Press Enter for all options (no passphrase needed unless preferred).

#### 2. Add SSH key to GitHub:

```
cat ~/.ssh/id_ed25519.pub
```

Copy the output and [add it here](#) → **New SSH Key**.

#### 3. Clone with SSH instead of HTTPS:

```
git clone git@github.com:gramaziokohler/ros_blockly.git
```

---

Let me know if you want to **skip authentication and download a ZIP file** instead — I can give the direct command or link.

**You said:**

```
git clone https://github.com/gramaziokohler/ros_blockly.git Cloning into 'ros_blockly'... Username for 'https://github.com':
```

**ChatGPT said:**

You're being asked for a GitHub **username and password** because of one of the following issues:

---

### Why This Happens

1. **You previously cloned partially into `ros_blockly`**, and the folder exists but is **corrupted** or **locked**.
  2. The clone is using **HTTPS**, and GitHub now requires **token-based authentication**, not just username/password.
  3. The GitHub repository is public, but your Git session may be confused due to cache or partial data.
- 

### Recommended Fix (Clean Start, No Username Needed)

#### 1. Delete any broken folder first:

```
• cd ~
rm -rf ros_blockly
```

#### • Clone again (as anonymous, no username required):

```
2. git clone https://github.com/gramaziokohler/ros_blockly.git
```

If this still asks for username, **stop using HTTPS** and follow the SSH method below.

---

### **Option: Use SSH Instead (No username prompt ever)**

#### **1. Check if SSH key exists:**

```
ls ~/.ssh/id_ed25519.pub
```

If it doesn't exist, generate one:

- `ssh-keygen -t ed25519 -C "your_email@example.com"`
- **Copy the public key:**
- `cat ~/.ssh/id_ed25519.pub`

Copy it and add it to GitHub here: <https://github.com/settings/keys> → **New SSH key**.

- **Clone using SSH:**

```
3. git clone git@github.com:gramaziokohler/ros_blockly.git
```

---

### **Alternative: Download ZIP Instead (no Git)**

If you're not familiar with Git or want a quick setup:

#### **1. Download this ZIP file:**

[https://github.com/gramaziokohler/ros\\_blockly/archive/refs/heads/master.zip](https://github.com/gramaziokohler/ros_blockly/archive/refs/heads/master.zip)

#### **2. Extract it:**

```
2. unzip master.zip
 mv ros_blockly-master ros_blockly
```

---

Let me know what method you prefer and I'll walk you through it step-by-step.

#### **You said:**

```
install apache sakthees-monk@saktheesmonk-desktop:~/robot_blockly/scripts$ sudo apt-get
install apache2 Reading package lists... Done Building dependency tree Reading state information...
Done The following packages were automatically installed and are no longer required: chromium-
codecs-ffmpeg-extra gstreamer1.0-vaapi lib32gcc-s1 libc6-i386 libgstreamer-plugins-bad1.0-0
linux-headers-5.15.0-67-generic linux-hwe-5.15-headers-5.15.0-67 linux-image-5.15.0-67-generic
linux-modules-5.15.0-67-generic linux-modules-extra-5.15.0-67-generic Use 'sudo apt autoremove'
to remove them. The following additional packages will be installed: apache2-bin apache2-data
apache2-utils libaprutil1-dbd-sqlite3 libaprutil1-ldap Suggested packages: apache2-doc apache2-
suexec-pristine | apache2-suexec-custom The following NEW packages will be installed: apache2
apache2-bin apache2-data apache2-utils libaprutil1-dbd-sqlite3 libaprutil1-ldap 0 upgraded, 6 newly
installed, 0 to remove and 0 not upgraded. Need to get 1,546 kB of archives. After this operation,
6,967 kB of additional disk space will be used. Do you want to continue? [Y/n] y Abort.cd
```



```
<catkin_ws_root>/src git clone https://github.com/erlerobot/robot_blockly cd
robot_blockly/frontend git clone https://github.com/erlerobot/blockly git clone
https://github.com/erlerobot/ace-builds cd ../scripts # install apache sudo apt-get install apache2 #
make sure that Apache is running properly in your robot and then ./deploy.sh
```

ChatGPT said:

You're on the right track to set up the `robot_blockly` project. Let's walk through everything cleanly step-by-step to fix and complete your installation:

---

## Step-by-Step Guide to Install and Deploy `robot_blockly` with Apache

### 1. Navigate to your `catkin_ws/src` folder

If not already created:

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

### 2. Clone the main repository

```
git clone https://github.com/erlerobot/robot_blockly.git
```

### 3. Navigate into the frontend directory and clone dependencies

```
cd robot_blockly/frontend
git clone https://github.com/erlerobot/blockly
git clone https://github.com/erlerobot/ace-builds
```

These are important frontend libraries needed by the Blockly interface.

### 4. Install Apache

Back in the terminal:

```
sudo apt-get update
sudo apt-get install apache2
```

Once installed, confirm Apache is running:

```
sudo systemctl status apache2
```

You should see something like: `active (running)`

### 5. Back to `scripts` folder to deploy frontend

```
cd ~/catkin_ws/src/robot_blockly/scripts
chmod +x deploy.sh
./deploy.sh
```

This script will copy the frontend files to the Apache server folder (usually `/var/www/html`), making them accessible via a browser.

---

## Access in Browser

Once deployed successfully, you can open your browser and go to:

`http://localhost`

If everything went well, you should see the Blockly UI.

---

## Troubleshooting Tips

- If `./deploy.sh` fails:
    - Open the script and ensure it's copying files to `/var/www/html`
    - You may need to run it as root:
  - `sudo ./deploy.sh`
  - If Apache is not showing Blockly UI:
  - Manually copy the frontend files:
  - `sudo cp -r ../frontend/* /var/www/html/`
  - If port 80 is blocked:
  - Try opening it:
    - `sudo ufw allow 'Apache Full'`
- 

Let me know if you hit any errors during the deploy or want to connect this to a ROS topic!

**You said:**

```
sakthees-monk@saktheesmonk-desktop:~/robot_blockly/scripts$
```

**ChatGPT said:**

Great! You're now inside the `robot_blockly/scripts` directory and ready for the next step.

To proceed with the **Blockly ROS integration**, run:

```
./deploy.sh
```

This script will:

- Copy Blockly frontend files to the Apache web server directory (`/var/www/html`).
  - Set up everything to view and run Blockly-based visual programming via browser.
- 

## What to do after `deploy.sh`:

### 1. Check in browser:

Open your browser and visit:

- `http://localhost`

You should see the Blockly-based robot programming interface.

- **Run the backend ROS services:**

Open a new terminal and run:

```
roscore
```

Then in another terminal:

```
2. rosrun robot_blockly server.py
```

This enables communication between the Blockly interface and ROS nodes.

---

### Tips:

- If you get a **permission error**, run:

```
• chmod +x deploy.sh
sudo ./deploy.sh
```

- Make sure your Apache server is running:

- `sudo systemctl status apache2`

Let me know if you want to add **TurtleSim**, **SLAM**, or **GMAP viewer** integration next.

**You said:**

```
chromium-codecs-ffmpeg-extra gstreamer1.0-vaapi lib32gcc-s1 libc6-i386 libgstreamer-plugins-
bad1.0-0 linux-headers-5.15.0-67-generic linux-hwe-5.15.0-headers-5.15.0-67 linux-image-5.15.0-
67-generic linux-modules-5.15.0-67-generic linux-modules-extra-5.15.0-67-generic Use 'sudo apt
autoremove' to remove them. The following additional packages will be installed: apache2-bin
apache2-data apache2-utils libaprutil1-dbd-sqlite3 libaprutil1-ldap Suggested packages: apache2-
doc apache2-suexec-pristine | apache2-suexec-custom The following NEW packages will be
installed: apache2 apache2-bin apache2-data apache2-utils libaprutil1-dbd-sqlite3 libaprutil1-ldap 0
upgraded, 6 newly installed, 0 to remove and 0 not upgraded. Need to get 1,546 kB of archives.
After this operation, 6,967 kB of additional disk space will be used. Do you want to continue? [Y/n]
y Abort. sakthees-monk@saktheesmonk-desktop:~/robot_blockly/scripts$./deploy.sh cp: target
'/var/www/html/' is not a directory sakthees-monk@saktheesmonk-desktop:~/robot_blockly/scripts$
```

ChatGPT said:

You're almost done — just a few fixes needed now to complete the Blockly + ROS setup.

---

### Problem:

`./deploy.sh` failed with:

```
cp: target '/var/www/html/' is not a directory
```

### Root Cause:

Apache is **not yet installed**, so the web root `/var/www/html/` **does not exist**.

---

## Fix Step-by-Step:

### 1. Install Apache Server (again — properly this time):

Your previous install attempt was **aborted**. Let's install it now:

```
sudo apt update
sudo apt install apache2 -y
```

### 2. Create the `/var/www/html/` directory (if still missing):

After installing, this should be created automatically. But just to be sure:

```
sudo mkdir -p /var/www/html
```

---

### 3. Rerun the deploy script:

Now that Apache is ready and the directory exists:

```
cd ~/robot_blockly/scripts
chmod +x deploy.sh
sudo ./deploy.sh
```

---

### 4. Test in Browser:

Open this link in Chromium or Firefox:

`http://localhost`

You should now see the Blockly ROS interface.

---

## Optional: Start ROS Connection

### Terminal 1:

```
roscore
```

### Terminal 2:

```
roslaunch robot_blockly server.py
```

---

Let me know when you're ready to add:

- TurtleSim control via Blockly
- GMAPPING (SLAM) visualizer
- AI or sensor nodes via Blockly

All of these can be added one by one!

**You said:**

http://localhost You should now see the Blockly ROS interface. not showing

ChatGPT said:

If `http://localhost` is not showing the Blockly ROS interface, let's troubleshoot step by step. This is a common issue when setting up `ros_blockly`. Below is a checklist to identify and fix the problem:

---

## Step-by-Step Fix for `http://localhost` not loading Blockly ROS:

### 1. Check if Apache is running

Run:

```
sudo systemctl status apache2
```

You should see something like:

```
active (running)
```

#### **If it's not running:**

Start it manually:

```
sudo systemctl start apache2
```

And enable it at boot:

```
sudo systemctl enable apache2
```

---

### 2. Check if Blockly files were copied to `/var/www/html/`

Run:

```
ls /var/www/html/
```

You should see files like:

```
index.html
blockly/
scripts/
```

### **If nothing is there:**

Try this again:

```
cd ~/robot_blockly/scripts
sudo ./deploy.sh
```

This script should copy the Blockly files to `/var/www/html`.

---

## **3. Check what port Apache is serving**

Apache should serve on port **80** by default.

Run:

```
sudo netstat -tulnp | grep apache2
```

or (if `netstat` not available):

```
sudo lsof -i -P -n | grep LISTEN | grep apache
```

You should see a line like:

```
apache2 LISTEN 0.0.0.0:80
```

### **If not listening on port 80:**

Check Apache configuration:

```
cat /etc/apache2/ports.conf
```

Make sure it says:

```
Listen 80
```

If you changed the port, access it like:

```
http://localhost:8080
```

Or whatever port is set.

---

## **4. Access Apache default page to confirm webserver is working**

Try:

```
http://localhost
```

You should see at least **Apache2 Ubuntu Default Page**.

### **If even that doesn't work:**

Your Apache install might be broken. Reinstall:

```
sudo apt remove apache2 -y
sudo apt install apache2 -y
```

---

## 5. Permissions Check

Make sure Apache has read access:

```
sudo chown -R www-data:www-data /var/www/html/
sudo chmod -R 755 /var/www/html/
```

---

## 6. Check Browser Console

If page loads but Blockly doesn't show, open **developer tools** (F12) → go to **Console tab** → see if any JS/CORS errors are showing. Tell me what you find.

---

## If Still Not Working, Please Share:

Run and send the output of:

1. `sudo systemctl status apache2`
2. `ls /var/www/html`
3. `curl http://localhost`
4. `cat /var/log/apache2/error.log | tail -n 20`

With that info, I can give you exact next commands.

Would you like to continue adding TurtleSim blocks after this works?

**You said:**

```
./deploy.sh sakthees-monk@saktheesmonk-desktop:~/robot_blockly/scripts$ sudo systemctl status
apache2 ● apache2.service - The Apache HTTP Server Loaded: loaded
(/lib/systemd/system/apache2.service; enabled; vendor preset: enabled) Active: active (running)
since Fri 2025-07-18 18:33:37 IST; 1min 11s ago Docs: https://httpd.apache.org/docs/2.4/ Main
PID: 46338 (apache2) Tasks: 55 (limit: 9322) Memory: 5.0M CGroup:
/system.slice/apache2.service └─46338 /usr/sbin/apache2 -k start └─46339 /usr/sbin/apache2 -k
start └─46340 /usr/sbin/apache2 -k start Jul 18 18:33:37 saktheesmonk-desktop systemd[1]:
Starting The Apache HTTP Server... Jul 18 18:33:37 saktheesmonk-desktop apachectl[46318]:
AH00558: apache2: Could not reliably de> Jul 18 18:33:37 saktheesmonk-desktop systemd[1]:
Started The Apache HTTP Server. lines 1-15/15 (END)
```

ChatGPT said:

It looks like Apache is installed and running properly, but your Blockly for ROS (

**You said:**

where is link

ChatGPT said:

If Apache is running **correctly**, then the Blockly ROS interface should be available at:

<http://localhost>

But since it's **not showing**, let's debug step by step.

---

### **1. Check if `/var/www/html/index.html` exists**

Run:

```
ls /var/www/html
```

Look for files like `index.html`, `blockly`, or `ros_blockly`. If nothing is there or the folder is empty, then the deployment didn't copy files correctly.

---

### **2. Re-run the deployment script with sudo:**

```
cd ~/robot_blockly/scripts
sudo ./deploy.sh
```

This ensures the script has permission to copy files to `/var/www/html/`.

---

### **3. Then restart Apache:**

```
sudo systemctl restart apache2
```

---

### **4. Visit in browser:**

Open:

`http://localhost`

Or try:

`http://127.0.0.1`

If there's a subdirectory like `ros_blockly`, try:

`http://localhost/ros_blockly`

---

### **Optional: Check what's in `/var/www/html`**

```
tree /var/www/html
```

Or just:

```
ls -l /var/www/html
```

If there's no content, the deployment didn't copy files properly.

---



Let me know the output of `ls /var/www/html`, and I'll guide you further if needed.

**You said:**

```
sudo apt install python3-rpi.gpio -y Reading package lists... Done Building dependency tree
Reading state information... Done E: Unable to locate package python3-rpi.gpio E: Couldn't find
any package by glob 'python3-rpi.gpio' sakthees-monk@saktheesmonk-desktop:~$
```

ChatGPT said:

You're seeing this error:

```
E: Unable to locate package python3-rpi.gpio
```

because you're on a **regular PC/Ubuntu system**, not a Raspberry Pi OS. The `python3-rpi.gpio` package is **only available in Raspberry Pi OS repos**.

---

### **Alternative: Use a GPIO library that works on all systems or simulate GPIO**

Since you're **not on a Raspberry Pi**, you can't (and don't need to) use `RPi.GPIO`.

---

### **If you're just testing `ros_blockly` or Blockly + ROS on Ubuntu:**

You can **safely skip** installing `python3-rpi.gpio` unless you plan to run this on a real Raspberry Pi board.

---

### **What to do:**

1. **Don't install `python3-rpi.gpio` — skip it.**
2. Make sure Apache is serving your Blockly interface correctly by re-running:

```
cd ~/robot_blockly/scripts
sudo ./deploy.sh
sudo systemctl restart apache2
xdg-open http://localhost
```

3. Use `ros_blockly` on your PC without worrying about Raspberry Pi GPIO.
- 

### **Want to simulate GPIO for testing?**

You could use `gpiozero` with a mock pin factory (for testing):

```
sudo apt install python3-gpiozero
```

Then, in Python:

```
from gpiozero import LED
from gpiozero.pins.mock import MockFactory

Use mock pin factory for simulation
from gpiozero import Device
```

```
Device.pin_factory = MockFactory()

led = LED(17)
led.on()
```

---

Let me know if you're planning to run this on a Raspberry Pi board, and I'll guide you with the correct setup. Otherwise, you're good to move ahead without `python3-rpi.gpio`.

**You said:**

for jet son and dji drones industrial 4.0 and minig robots and pipe inspection and mid sea submarines also give all peripherals and install and agv and defence based install both gui and python based

**ChatGPT said:**

You're asking for a **comprehensive setup** for **Jetson Nano** (and similar edge devices) used in:

- **Industrial 4.0** applications
- **AGVs (Autonomous Guided Vehicles)**
- **DJI drones**
- **Mining robots**
- **Pipe inspection**
- **Mid-sea submarines**
- **Defense robotics**

You want both **Python-based and GUI-based control + monitoring**, and need:

- Hardware/peripherals list
  - Complete install stack (ROS, vision, GPIO, ML, SLAM)
  - Real-time GUI + script interface
  - Offline capability, defense-grade reliability
- 

## PART 1: HARDWARE SETUP

| System          | Suggested Hardware/Peripherals                                   |
|-----------------|------------------------------------------------------------------|
| Jetson Nano     | 4GB B01 model, 128GB UHS-I SD, heatsink/fan                      |
| Motor Drivers   | Cytron MD10C, L298N, or Roboclaw                                 |
| Sensors         | LiDAR (RPLIDAR A1/A2), Ultrasonic, IMU (MPU-9250), GPS (NEO-M8N) |
| Cameras         | Raspberry Pi Camera v2 or Intel RealSense D435i                  |
| GPIO/I2C        | Industrial-grade I/O boards (e.g., Sequent Microsystems)         |
| Power Supply    | 5V 4A min or LiPo 11.1V with 5V BEC                              |
| Chassis/Frame   | DFRobot Rover, waterproof casing for submarine/drones            |
| Comms           | 4G/5G, WiFi, LoRa, or Xbee Pro for long-range                    |
| DJI Integration | DJI SDK for onboard flight controller integration                |

---

# PART 2: SOFTWARE INSTALLATION STACK

Target: Jetson Nano running **Ubuntu 20.04 / 18.04 (JetPack 4.x)**

## 1. Update System + JetPack

```
sudo apt update && sudo apt upgrade -y
```

## 2. Install ROS (Melodic/Noetic on Jetson Nano)

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu bionic main" >
/etc/apt/sources.list.d/ros-latest.list'
sudo apt install curl
curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo
apt-key add -
sudo apt update
sudo apt install ros-melodic-desktop-full -y
echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc
source ~/.bashrc
sudo apt install python-rosdep python-rosinstall python-rosinstall-generator
python-wstool build-essential
sudo rosdep init
rosdep update
```

---

## 3. Install Required ROS Packages

```
sudo apt install ros-melodic-navigation ros-melodic-slam-gmapping ros-melodic-
move-base ros-melodic-joy ros-melodic-tf ros-melodic-rviz ros-melodic-robot-
state-publisher ros-melodic-map-server
```

---

## 4. Install Perception + Vision Stack

```
sudo apt install ros-melodic-cv-bridge ros-melodic-image-transport python3-
opencv
sudo apt install ros-melodic-vision-opencv
sudo apt install v4l-utils
```

For RealSense camera:

```
sudo apt install librealsense2-dkms librealsense2-utils librealsense2-dev
```

---

## 5. Install GPIO + I2C + UART

```
sudo apt install python3-pip i2c-tools
sudo pip3 install Jetson.GPIO smbus pyserial
sudo groupadd -f gpio
sudo usermod -a -G gpio $USER
```

---

## 6. Install AGV + SLAM + Obstacle Avoidance

```
cd ~/catkin_ws/src
git clone https://github.com/ros-planning/navigation.git
git clone https://github.com/tu-darmstadt-ros-pkg/hector_slam.git
cd ~/catkin_ws
catkin_make
```

---

## 7. Install GUI Tools

```
sudo apt install ros-melodic-rqt ros-melodic-rqt-common-plugins ros-melodic-rviz
```

You can also use:

- **Web-based GUI:** rosbridge, roswww, roslibjs
  - **Blockly for ROS:** Visual programming interface
  - **Jupyter Lab:** GUI + script blending
- 

## 8. Install Drone/DJI SDK

```
DJI Onboard SDK
git clone https://github.com/dji-sdk/Onboard-SDK.git
cd Onboard-SDK
mkdir build && cd build
cmake ..
make -j4
sudo make install
```

Also install:

```
sudo apt install ros-melodic-mavros ros-melodic-mavros-extras
```

Setup PX4/DJI communication through /dev/ttyUSB0 or UART.

---

# PART 3: PYTHON + GUI BASED CONTROL

## A. Python APIs for Sensors + Actuators:

- `Jetson.GPIO` – GPIO I/O
  - `smbus` – I2C sensors
  - `pyserial` – UART/GPS/ESC
  - `cv2`, `pyrealsense2` – camera control
  - `rospy` – ROS Python control
-

## B. GUI-Based Interfaces:

| Tool                          | Purpose                         |
|-------------------------------|---------------------------------|
| rviz                          | 3D robot + sensor visualization |
| rqt_graph                     | Node + topic debugging          |
| rqt_robot_steering            | Manual robot control            |
| Blockly for ROS               | Drag-and-drop robot coding      |
| WebGUI with Flask + SocketIO  | Live control on web/mobile      |
| Jupyter + OpenCV + Matplotlib | Robot logs + vision control     |

---

## PART 4: SPECIAL CASES

### Underwater / Pipe / Submarine Bots

- Waterproof servo motors (Blue Robotics)
  - Use **Ethernet cameras**, IMU
  - ROS Nodes:
    - `bluerov_ros_playground`
    - `rovio` (Visual-Inertial Odometry)
    - `robot_localization`
  - Use ROSbag for offline debugging
  - Use `nav_msgs/Odometry`, `sensor_msgs/Imu`
- 

### Mining / Industrial Robots

- Use **LiDAR + Camera + Dust-Proof enclosure**
  - Remote GUI via VNC or WebRTC
  - Real-time alerts with MQTT or ZeroMQ
- 

## DEFENSE-GRADE SECURITY

- Use **firewalled VPN** like Tailscale or WireGuard
  - Harden OS with ufw, AppArmor, read-only SD
  - Enable **ROS Secure Mode (ROS2)** if possible
  - Use CAN bus for robustness (e.g., `ros_canopen`)
-

# OPTIONAL INSTALLS

| Feature               | Command                                                               |
|-----------------------|-----------------------------------------------------------------------|
| Docker for ROS        | <code>sudo apt install docker.io + use ros:melodic image</code>       |
| TensorFlow/PyTorch    | JetPack pre-installed (check with <code>nvcc --version</code> )       |
| YOLO Object Detection | Use <code>ultralytics/yolov5</code> or <code>ros-deep-learning</code> |

---

Would you like a **bash script to automate** this full install on Jetson Nano, or a **PDF manual** for field deployment in AGVs, drones, and submarines?

Let me know which part you want first: full GUI interface, Python control panel, ROS2 upgrade, or mission planner.

**You said:**

from humanoid to gaudar pod and allkinds of robot

**ChatGPT said:**

To create a **complete robotics software + hardware ecosystem** (GUI + Python CLI + ROS-based) that can support:

- **Humanoids, quadrupeds, AGVs**
- **Jetson Nano/Orin, DJI drones, Defence robots, Industry 4.0 robots**
- **Mining robots, Pipe inspection, Submarines, and Mid-sea robotics**

You need a **modular, scalable setup** covering simulation, control, sensors, communication, and UIs.

---

## Master Robotics Stack Setup Guide (Ubuntu 20.04/22.04)

### 1. Essential System Setup

```
sudo apt update && sudo apt upgrade -y
sudo apt install -y build-essential git curl wget cmake unzip
```

---

### 2. Install ROS (Recommended: ROS Noetic or ROS 2 Humble)

**For ROS Noetic (Ubuntu 20.04, Python 3)**

```
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 install curl
curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo
apt-key add -

sudo apt update
sudo apt install ros-noetic-desktop-full -y
echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

```
sudo apt install -y python3-rosdep python3-rosinstall python3-rosinstall-
generator python3-wstool build-essential
sudo rosdep init
rosdep update
```

---

### **3. Install Jetson SDK + CSI Cameras + Deep Learning**

#### **On Jetson Nano / Orin:**

```
sudo apt install nvidia-jetpack
sudo apt install python3-opencv libopencv-dev
```

Use [jetson-inference](#) for AI + vision.

---

### **4. Robot Drivers + Libraries**

```
GPIO, I2C, SPI for Jetson/RPi
sudo apt install python3-pip i2c-tools python3-smbus
pip3 install Jetson.GPIO adafruit-blinka RPi.GPIO pigpio

CAN Bus / Serial
sudo apt install can-utils python3-can python3-serial

Motor & IMU Libraries
pip3 install adafruit-circuitpython-motor adafruit-circuitpython-lsm6ds lsm303-
accel
```

---

### **5. Robot Programming (Python GUI + CLI)**

```
pip3 install pyserial opencv-python PyQt5 matplotlib numpy scipy pillow
```

For GUI:

- Tkinter
  - PyQt5
  - Kivy
  - PyGame (for gamepad interface)
- 

### **6. Install Simulation + RViz + Gazebo**

```
sudo apt install ros-noetic-gazebo-ros-pkgs ros-noetic-rviz ros-noetic-joint-
state-publisher-gui
```

For quadrupeds/humanoids:

```
sudo apt install ros-noetic-moveit ros-noetic-control-toolbox
```

---

## 7. AI + Edge Compute Packages

```
ML / DL
pip3 install torch torchvision torchaudio
pip3 install tensorflow keras scikit-learn
```

Install NVIDIA DeepStream for Jetson:

```
sudo apt install deepstream-6.2
```

---

## 8. Peripherals for All Robots

- **Sensors:** LiDAR (RPLidar, Leddar), IMUs (MPU6050, BNO055), ultrasonic, IR
  - **Cameras:** CSI (Jetson), USB, Realsense D435
  - **Actuators:** Dynamixel (XL430), MG996R, BLDCs
  - **Comms:** WiFi, 4G/5G dongle, LoRa, RF, CAN
  - **Mobility:** Track wheels, underwater motors, quadruped legs (servo-based or brushless)
  - **Power:** Li-ion + smart BMS
- 

## 9. Robot Frameworks + GUIs

- **Blockly + Web GUI:** [robot\\_blockly](#)
  - **Mission Planning GUI:** QGroundControl for drones, Rviz+MoveIt for ground bots
  - **Full framework:**
    - [OpenManipulator](#)
    - [Husky](#)
    - [Spot\\_micro](#)
- 

## 10. Web Interface Control

```
sudo apt install apache2
cd ~/robot_blockly/scripts
./deploy.sh
xdg-open http://localhost
```

---

## 11. Test with Simulators & Examples

```
roslaunch turtlebot3_gazebo turtlebot3_empty_world.launch
roslaunch spot_micro_motion_cmd spot_micro_launch.launch
```

---



## 12. Install Industrial/Defense Specific Tools

- **ROS-Industrial:** `ros-noetic-industrial-core`
  - **Defence drone platform:** ArduPilot + MAVROS
  - **Underwater:** BlueROV2 (QGroundControl + ROS)
  - **AGVs:** Nav2 Stack (ROS 2), Cartographer
- 

## You Now Have:

- Python & GUI interfaces
  - Web dashboard with Blockly
  - Full ROS control + sensors + actuation
  - Jetson Nano/Orin + drones + quadrupeds supported
  - AGVs, submarines, defence bots, pipe inspection, industrial robots
- 

If you want, I can give you a **GitHub folder structure + `install_all.sh`** script for automation. Would you like that?

**You said:**

android apple defence electronics communication etc borwell rescue and every thing give

**ChatGPT said:**

You're exploring a huge range—from Android & Apple devices, industrial electronics, communication, rescue systems, to defense robotics, drones, submarines, and IoT-enabled robots. Here's a complete overview of platforms and integration frameworks across these domains:

---

## 1. Mobile Device Control (Android & iOS) for Industrial & Rescue Robots

- **Android Team Awareness Kit (ATAK):** Widely used by militaries and emergency responders for real-time situational awareness and communication. [Wikipedia+4UCI Social Sciences+4ResearchGate+4Wikipedia](#)
- **ROS-Mobile App:** An Android application that lets you control and monitor ROS robots directly from your phone via ROS bridge (rosbridge), offering robot telemetry, topic monitoring, and command publishing. [arXiv+1Wikipedia+1](#)
- **Industrial Robot Tablet Interface:** Example: ASystr robots programmed using an Android tablet over Bluetooth, offering point teaching, supervision, and emergency controls. [systemantics.com](#)

**Use Case:** Build Android/iOS apps (e.g. with React Native + OPC-UA or rosbridge) to teleoperate quadrupeds, drones, or AGVs remotely from your phone/tablet.

---

## 2. Unmanned Ground Vehicles (UGVs) & Rescue Robots

- **Taurob Tracker:** A rugged, ATEX-rated UGV used by fire departments and military for smoke environment reconnaissance and rescue tasks.  
[arXiv+8Medium+8www.slideshare.net+8Wikipedia+1www.slideshare.net+1](#)
  - **UGV Interoperability Profile (IOP):** Defense standard (managed by US DoD) for open architecture and modular control of unmanned ground vehicles. [Wikipedia](#)
  - **Karo Rescue Robot:** Demonstrated GUI-based control (ROS + RQt/plugin) for rescue missions including thermal imaging, LIDAR+IMU-based mapping, joystick control, and autonomy layers. [SpringerOpen](#)
- 

## 3. Drones, Submarines & Multi-Modal Robots

- **ArduPilot:** Open-source autopilot used across UAVs, fixed-wing, VTOLs, submarines, surface vehicles. Widely adopted—including by military and research communities.  
[Wikipedia](#)
  - **Rescue Quadruped Robots:** Prototypes equipped with cameras and sensors, controlled via smartphone apps (e.g., Blynk) for detecting survivors in disaster zones.  
[ResearchGate+3www.slideshare.net+3SCIRP+3](#)
- 

## 4. Communications: Industrial, Defense & Rescue

- **Wi-Fi, Bluetooth, Satellite & Mesh Networks:** Hybrid communication strategies are used in rescue operations to maintain connectivity in harsh terrains. Mesh networks ensure self-healing and coverage in damaged infrastructure environments. [COMPUTER SCIENCE CAFÉ](#)
- 

## 5. UIs & Integration Frameworks

- **Hector UI (2025):** Recent open-source UI built specifically for rescue and inspection robots, adaptable to wheeled, tracked, and walking robots. Offers remote telemetry, control and flexible operator visualization interfaces. [arXiv](#)
-

# Integration Matrix for Full-Stack Robotics

| Domain              | Mobile Control                                                                       | Core Framework            | UI / Interface                | Comms                      | Hardware Platforms            |
|---------------------|--------------------------------------------------------------------------------------|---------------------------|-------------------------------|----------------------------|-------------------------------|
| Industrial Robots   | Android/iOS<br>Tablets (React Native + OPC-UA / rosbridge)<br><a href="#">Medium</a> | ROS or custom             | Mobile GUI + web dashboard    | Wi-Fi / Bluetooth / OPC-UA | 6-axis manipulators, AGVs     |
| Rescue UGV / Robot  | ROS-Mobile, ATAK                                                                     | ROS (Noetic or 2.x)       | Hector UI, RQt, RViz          | Mesh, Satellite, Wi-Fi     | Taurob Tracker, Karo quad/UGV |
| Drones / Submarines | Tablet or phone app + mavros / ArduPilot                                             | ArduPilot, ROS            | QGroundControl, ROS GUI       | Radio, Satellite, CAN      | DJI / PX4 drones, BlueROV2    |
| Detection Robots    | Blynk smartphone app                                                                 | ROS + Camera Pipeline     | Custom tablet UI + video feed | Wi-Fi                      | Quadruped rescue bots         |
| Android-Based Bots  | Android control via IOIO boards                                                      | Microcontroller + Android | Mobile app UI                 | Wi-Fi, Bluetooth           | Lego/ Educational robots      |

## Getting Started: Example Setup for a Rescue Robot with Android Control

### 1. Base Robot Stack:

- Install ROS (Noetic or ROS 2 Jazzy Jalisco – latest as of May 2024) [UCI Social SciencesWikipedia](#).
- Add Hector UI or custom RQt plugins for telemetry & video.

### 2. Mobile Integration:

- Use **ROS-Mobile** (Android) to subscribe/publish to ROS topics via rosbridge.
- Alternatively, develop a hybrid app with React Native + NodeOPCUA for controlling hardware (as used in industrial robots). [arXivMedium](#)

### 3. Communication Stack:

- Set up mesh networking (e.g. BATMAN-adv, Mesh Wi-Fi) for multi-robot coordination.
- Use satellite link (e.g., Iridium/Mavlink backup) for remote operation beyond local Wi-Fi range.

### 4. Simulation & Field Tests:

- Use Gazebo + RViz for verifying movement, SLAM, control.
- Deploy physical robot (Taurob Tracker or custom quadruped) with SLAM and remote control.

---

Would you like me to provide:

- A **template Android/iOS app** codebase for ROS integration?
- A **bash installer script** for ROS + Hector UI + ROS-Mobile + SLAM?
- A **Modular Github project structure** that covers defense robotics, drones, rescue bots?

Let me know which direction you want to go next—I can generate scripts, prototypes, or guidance tailored for humanoid, quadruped, drone, or underwater robots.

**You said:**

give installation for all

**ChatGPT said:**

Creating a **complete setup** for **all types of robots**, including **humanoids, quadrupeds, AGVs, drones (DJI), submarines, pipe inspection, mining robots**, along with **Android/Apple communication, industrial 4.0, borewell rescue, defense electronics**, and covering **GUI and Python-based programming tools**, is a massive ecosystem. However, here's a **modular guide** you can follow **step-by-step**, depending on your interest.

---

## System Preparation (Ubuntu 20.04 or 22.04 Recommended)

```
sudo apt update && sudo apt upgrade -y
sudo apt install git curl wget build-essential python3-pip python3-dev python3-venv -y
```

---

### 1. ROS Noetic (for Ubuntu 20.04)

```
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 install curl -y
curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo
apt-key add -
sudo apt update
sudo apt install ros-noetic-desktop-full -y
echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc
source ~/.bashrc
sudo apt install python3-rosdep -y
sudo rosdep init
rosdep update
```

---

### 2. GUI Tools for Robot Visualization

```
sudo apt install ros-noetic-rviz ros-noetic-rqt ros-noetic-rqt-common-plugins
ros-noetic-rqt-robot-plugins -y
```

---

### 3. Turtlesim and Basic Simulations

```
sudo apt install ros-noetic-turtlesim -y
```

---

### 4. Blockly (ROS + Visual Programming)

```
cd ~
git clone https://github.com/gramaziokohler/ros_blockly.git
cd ros_blockly
chmod +x ros_blockly.sh
./ros_blockly.sh
```

Open Blockly in browser:

<http://localhost:8080>

---

### 5. Drone (DJI SDK + MAVROS + PX4/ArduPilot)

```
sudo apt install ros-noetic-mavros ros-noetic-mavros-extras -y
wget https://raw.githubusercontent.com/mavlink/mavros/master/mavros/scripts/
install_geographiclib_datasets.sh
chmod +x install_geographiclib_datasets.sh
sudo ./install_geographiclib_datasets.sh
```

For DJI:

- Install DJI OSDK: <https://github.com/dji-sdk/Onboard-SDK>
  - ROS package: <https://github.com/dji-sdk/Onboard-SDK-ROS>
- 

### 6. AGVs / AMRs / Defense Robots

Install packages:

```
sudo apt install ros-noetic-navigation ros-noetic-gmapping ros-noetic-slam-
gmapping ros-noetic-amcl ros-noetic-move-base -y
```

---

### 7. Humanoids / Quadrupeds

```
sudo apt install ros-noetic-dynamixel-sdk ros-noetic-dynamixel-workbench* -y
```

Use with:

- OpenQuadruped: <https://github.com/Robotics-Club-IIT-BHU/OpenQuadruped>
- ROS humanoids: <https://github.com/ros-naoqi>

---

## 8. Borewell Rescue Robots

Use:

- SnakeBot control kits
- Teleoperated control via Joystick/ROS

```
sudo apt install ros-noetic-joy ros-noetic-teleop-twist-keyboard -y
```

---

## 9. Pipe Inspection / Industrial Robots

Use:

- UR5/UR10: [https://github.com/UniversalRobots/Universal\\_Robots\\_ROS\\_Driver](https://github.com/UniversalRobots/Universal_Robots_ROS_Driver)
  - Pipebots: Use crawler bots + Lidar + camera stream
- 

## 10. Electronics / Communication

- Install Serial:

```
sudo apt install ros-noetic-serial -y
```

- Install CAN Communication:

```
sudo apt install ros-noetic-canopen* -y
```

---

## 11. Python Robot Frameworks

```
pip3 install opencv-python pyserial numpy scipy matplotlib imutils
```

---

## 12. Android & iOS ROS Communication

**Android:**

- Use **ROS Android Sensor Driver**:  
[https://github.com/rosjava/android\\_core](https://github.com/rosjava/android_core)
- Or use WebSockets (rosbridge\_suite):

```
sudo apt install ros-noetic-rosbridge-server -y
```

## iOS:

Use **roslibjs** in WebView or flutter-dart bridge.

---

## 13. Mid Sea / Submarine ROS

- Use:
  - **BlueROV2**: <https://www.bluerobotics.com>
  - **ArduSub + QGroundControl**
  - **Underwater GPS + SONAR packages**

```
sudo apt install ros-noetic-sonar-image-view ros-noetic-sonar-drivers -y
```

---

## 14. Industrial 4.0 Setup

- MQTT Integration:

```
sudo apt install mosquitto mosquitto-clients -y
pip3 install paho-mqtt
```

- OPC-UA:

```
pip3 install opcua
```

---

## 15. AI, ML, Vision, YOLO etc.

```
pip3 install torch torchvision ultralytics scikit-learn pandas
sudo apt install ros-noetic-cv-bridge ros-noetic-image-transport -y
```

---

## 16. Setup All Workspace

```
mkdir -p ~/catkin_ws/src
cd ~/catkin_ws
catkin_make
echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

---

## Optional: Defense/Encrypted Comms & Remote Ops

- Use **ROS Secure DDS**, VPNs (Wireguard), and encrypted ROS 2 nodes.
  - Enable SSH, MQTT TLS, or WebRTC as needed.
-

Would you like me to generate a **PDF setup guide** for offline use or make a **bash script** to automate all of this?

**You said:**

```
sudo apt update && sudo apt upgrade -y [sudo] password for sakthees-monk: Sorry, try again.
[sudo] password for sakthees-monk: Hit:2 http://security.ubuntu.com/ubuntu focal-security
InRelease Hit:3 http://in.archive.ubuntu.com/ubuntu focal InRelease Hit:4
http://in.archive.ubuntu.com/ubuntu focal-updates InRelease Hit:5
http://in.archive.ubuntu.com/ubuntu focal-backports InRelease Hit:6
http://in.archive.ubuntu.com/ubuntu focal-proposed InRelease Get:1
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64 InRelease [1,581
B] Err:1 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64 InRelease
The following signatures couldn't be verified because the public key is not available: NO_PUBKEY
A4B469963BF863CC Reading package lists... Done W: GPG error:
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64 InRelease: The
following signatures couldn't be verified because the public key is not available: NO_PUBKEY
A4B469963BF863CC E: The repository
'http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64 InRelease' is not
signed. N: Updating from such a repository can't be done securely, and is therefore disabled by
default. N: See apt-secure(8) manpage for repository creation and user configuration details.
(yoloven) (base) sakthees-monk@saktheesmonk-desktop:~$
```

**ChatGPT said:**

You're seeing this error because the NVIDIA CUDA repository you're using is for **Ubuntu 16.04**, but you're on **Ubuntu 20.04 (focal)**, and it's missing the GPG key required to verify its authenticity.

---

## Here's how to fix it:

### Option 1: (RECOMMENDED) Remove the outdated CUDA repo if you're not using it:

```
sudo rm /etc/apt/sources.list.d/cuda*
sudo apt update
```

This removes the misconfigured repository entirely.

---

### Option 2: Fix the GPG Key if you still need CUDA from that repository:

```
sudo apt-key adv --fetch-keys
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/
x86_64/7fa2af80.pub
sudo apt update
```

But **this is not ideal** for Ubuntu 20.04 users, since you're pulling packages from a repository meant for Ubuntu 16.04. This might cause compatibility issues.

---



### Option 3: Add the correct CUDA repo for Ubuntu 20.04

1. Remove the old repo:

```
sudo rm /etc/apt/sources.list.d/cuda*
```

2. Add the correct repo for Ubuntu 20.04:

```
wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/x86_64/cuda-ubuntu2004.pin
sudo mv cuda-ubuntu2004.pin /etc/apt/preferences.d/cuda-repository-pin-600
```

```
sudo apt-key adv --fetch-keys
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/x86_64/7fa2af80.pub
```

```
sudo add-apt-repository "deb
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/x86_64/ "
```

```
sudo apt update
```

Then you can run:

```
sudo apt upgrade -y
```

---

Let me know if you want to use the CUDA repo or just remove it.

The last ROS 1 release Noetic will go end of life on [May 31st](#) with that the ROS Wiki (this website) will also be EOL and transition to being an archive. **Maintainers:** Please migrate any wiki content into your package's README.md file. If you need more help on migrating code please see this [migration guide](#). Or watch Shane's [Lightning Talk from ROSCon 2024](#).



Documentation

Browse Software

New

- [blockly](#)

### ROS 2 Documentation

The ROS Wiki is for ROS 1. Are you using ROS 2 ([Humble](#), [Iron](#), or [Rolling](#))?

[Check out the ROS 2 Project Documentation](#)

Package specific documentation can be found on [index.ros.org](#)

# Wiki

- [Distributions](#)
- [ROS/Installation](#)
- [ROS/Tutorials](#)
- [RecentChanges](#)
- [blockly](#)

# Page

- Immutable Page
- [Info](#)
- [Attachments](#)
- 
- 

# User

- [Login](#)

# Blockly

[See blockly on index.ros.org for more info including anything ROS 2 related.](#)

[Documentation Status](#)

Contents

1. [Blockly](#)
  1. [Overview](#)
  2. [Background](#)
  3. [Nodes](#)
    1. [blockly\\_server](#)
      1. [Subscribed Topics](#)
4. [Usage](#)
  1. [Frontend](#)
  2. [Backend](#)
5. [Resources](#)
6. [More information about robot\\_blockly](#)

## Overview

This package provides web-based visualization and block programming tools for robots and drones based in blockly ([github.com/google/blockly](https://github.com/google/blockly)).

Builder



Blocks

Python code

Graph

Logic

Loops

Math

Lists

Variables

Functions

Erle-Spider

Erle-Copter

Erle-HexaCopter

Erle-Rover

repeat 10 times

do

Stand up/down

Go backwards 3 seconds

# Background

For several weeks we (Erle Robotics) taught different groups (going from the high-school level to the PhD one) how to make use of our robots. While most of the people quickly understand the different mechanical parts of a robot, understanding the underlying software is something that took quite a bit of effort.

When making our training sessions, one of the first things we try to introduce is the concept of the Robot Operating System (ROS). While there're many resources on this topic, it's a fact that learning ROS takes some effort even at the PhD level so we started prototyping ideas to make this process as simple as possible.

We wanted to reach high schools students so we realized that we had to remove the assumption of "coding skills" from the equation. This made us look into systems like Scratch for robot programming. Taking inspiration on some previous work we started prototyping. The output of these prototypes has become robot\_blockly. A multiplatform web-based tool for programming robots and drones that use ROS.

## Nodes

### blockly\_server

Backend with the corresponding websockets to launches the ROS code coming from the frontend.

### Subscribed Topics

blockly (invalid message type for MsgLink(msg/type))

- Communication with the robot\_blockly server. To be used in a later stage.

## Usage

### Frontend

```
cd <catkin_ws_root>/src
git clone https://github.com/erlerobot/robot_blockly
cd robot_blockly/frontend
git clone https://github.com/erlerobot/blockly
git clone https://github.com/erlerobot/ace-builds
cd ../scripts
```

```
install apache
sudo apt-get install apache2
make sure that Apache is running properly in your robot and then
./deploy.sh
```

### Backend

```
Install dependencies
sudo pip3 install rosdep rosinstall_generator wstool rosinstall
sudo pip3 install autobahn
cd <catkin_ws_root>
catkin_make_isolated --pkg blockly --install
source install_isolated/setup.bash
roslaunch blockly blockly_backend.py
```

```
now go to http://erle-brain-2.local/
and start playing!
```

## Resources

- Code: [https://github.com/erlerobot/robot\\_blockly](https://github.com/erlerobot/robot_blockly)
- Support: <http://forum.erlerobotics.com/>
- More docs: <http://erlerobotics.com/docs/ROS/Blockly/Intro.html>

## More information about robot\_blockly

More information is available at [Erle Robotics docs](#).

Wiki: blockly (last edited 2015-12-07 12:51:31 by [VictorMayoral](#))

Except where otherwise noted, the ROS wiki is licensed under the [Creative Commons Attribution 3.0](#)

---