



5. USB Camera and Cable Testing

The USB camera will be tested with the help of a python code. This is because we will access the camera using python scripts. You could ofcourse verify if the camera works fine by simply connecting it to the laptop/PC and running cheese, but, we in this section we shall test the USB camera using a python script. We will also test the extension cable to ensure that it also works as expected.

In this test, we will be using OpenCV extended python package. Run the following command in the terminal to check or install the mentioned library.

```
pip3 install opencv-contrib-python
```



After you've installed openCV, perform the following steps:

Step 1: Open the `camera_test.py` file present inside the **Task_3_resources** folder.

Step 2: Connect the USB camera to the extension cable and the extension cable to the computer's USB port.

Step 3: Now run the python script and check if you get the live camera feed as output.

Note: You can change the value of `camera_id` parameter in the code to get video feed of a different camera (eg: your laptop's webcam).

Testing the camera through the ROS package.

After successfully testing the camera using a Python script, we are now poised to extend our evaluation by testing the camera with the ROS2 package. This advancement marks a significant step forward, laying the foundation for subsequent tasks.

Some insights on usb_cam package:

This package acts as a bridge, facilitating communication between the ROS framework and the libv4l2 library's kernel API. This integration simplifies the process of working with standard USB web cameras in ROS2-based robotic applications, offering configurability to suit a variety of use cases.

libv4l2 library : This library is known for implementing a standardized driver for USB web cameras, making it easier to interact with these devices in a consistent and configurable manner.

Step 1 : Installation of usb_cam package :

Open the terminal and copy the below commands:

```
sudo apt-get install v4l-utils
```



```
sudo apt-get install ros-humble-usb-cam
```

Step 2 : Setup :

Connect the USB camera to your system and execute the following commands to view the list of video devices:

```
v4l2-ctl --list-devices
```



```
ertslab@ertslab-ThinkCentre-E73z:~$ v4l2-ctl --list-devices
USB 2.0 Camera: RGB_USB (usb-0000:00:14.0-1):
/dev/video2
/dev/video3
/dev/media1

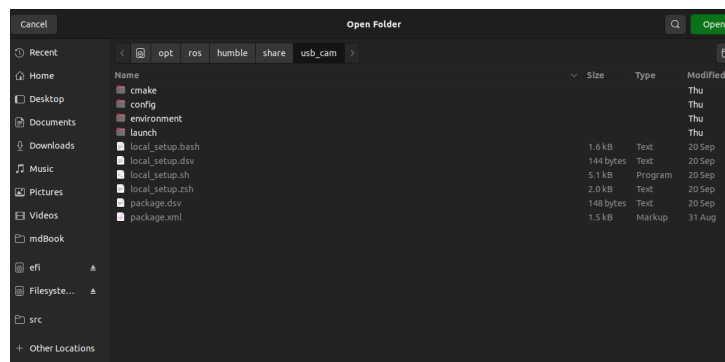
Integrated Camera: Integrated C (usb-0000:00:1a.0-1.3):
/dev/video0
/dev/video1
/dev/media0
```

In the image above, you can observe that two devices are currently connected to the system:

- USB 2.0 Camera: RGB_USB
- Integrated Camera

The **USB 2.0 Camera: RGB_USB** is the USB camera we are looking in to.

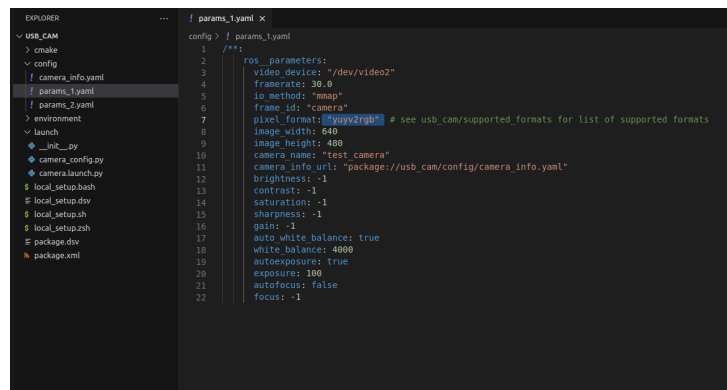
Now, open Visual Studio Code (VSCode) (or any other IDE of your choice, we have used VSCode in this guide), click on "Open Folder," and navigate to "Other Locations," available on the left-hand side at the bottom of the tab. From there, navigate to the "Computer" directory. Search for the usb_cam package. Refer to the image below to easily locate this package.



Now, navigate to the "config" directory and open "params_1.yaml". Inside "params_1.yaml", change the respective video device parameter to "/dev/video2".

Please note that you have to check the list of devices and adjust it accordingly. The path can change when you remove the camera device from the system, so please periodically review the list of devices.

Also, change the pixel format to "yuyv2rgb". You can refer to the image below for clarification.

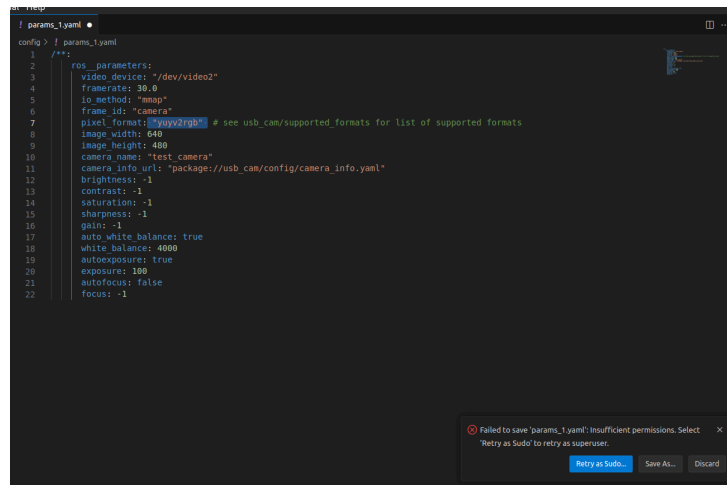


```

1 config > / params_1.yaml
2 /**:
3   ros_parameters:
4     video_device: "/dev/video2"
5     framerate: 30.0
6     io_method: "mmap"
7     frame_id: "camera"
8     pixel_format: "yuyv2rgb" # see usb_cam/supported_formats for list of supported formats
9     image_width: 640
10    image_height: 480
11    camera_name: "test_camera"
12    camera_info_url: "package://usb_cam/config/camera_info.yaml"
13    brightness: -1
14    contrast: -1
15    saturation: -1
16    sharpness: -1
17    gain: -1
18    auto_white_balance: true
19    white_balance: 4000
20    autoexposure: true
21    exposure: 100
22    autofocus: false
23    focus: -1

```

Now, press **Ctrl+S** to save the "params_1.yaml." A popup will appear at the bottom right-hand side. Press "Retry as sudo" and enter the password of your system to save this file.



```

1 config > / params_1.yaml
2 /**:
3   ros_parameters:
4     video_device: "/dev/video2"
5     framerate: 30.0
6     io_method: "mmap"
7     frame_id: "camera"
8     pixel_format: "yuyv2rgb" # see usb_cam/supported_formats for list of supported formats
9     image_width: 640
10    image_height: 480
11    camera_name: "test_camera"
12    camera_info_url: "package://usb_cam/config/camera_info.yaml"
13    brightness: -1
14    contrast: -1
15    saturation: -1
16    sharpness: -1
17    gain: -1
18    auto_white_balance: true
19    white_balance: 4000
20    autoexposure: true
21    exposure: 100
22    autofocus: false
23    focus: -1

```

After saving the file you can now run the below commands to start the camera node

```
ros2 launch usb_cam camera.launch.py
```

Note: If you encounter any errors after executing this command, you may need to change "video2" in "/dev/video2" to another number. To do this, you can list all video devices.

After running the above node successfully open new tab and run

```
ros2 topic list
```

You will be able to see the `/camera1/image_raw` topic in the terminal.

Now, run the `ros2_camera_test.py` file located inside the `Task_3_resources` folder. This script essentially subscribes to the topic `/camera1/image_raw` and displays the video output.