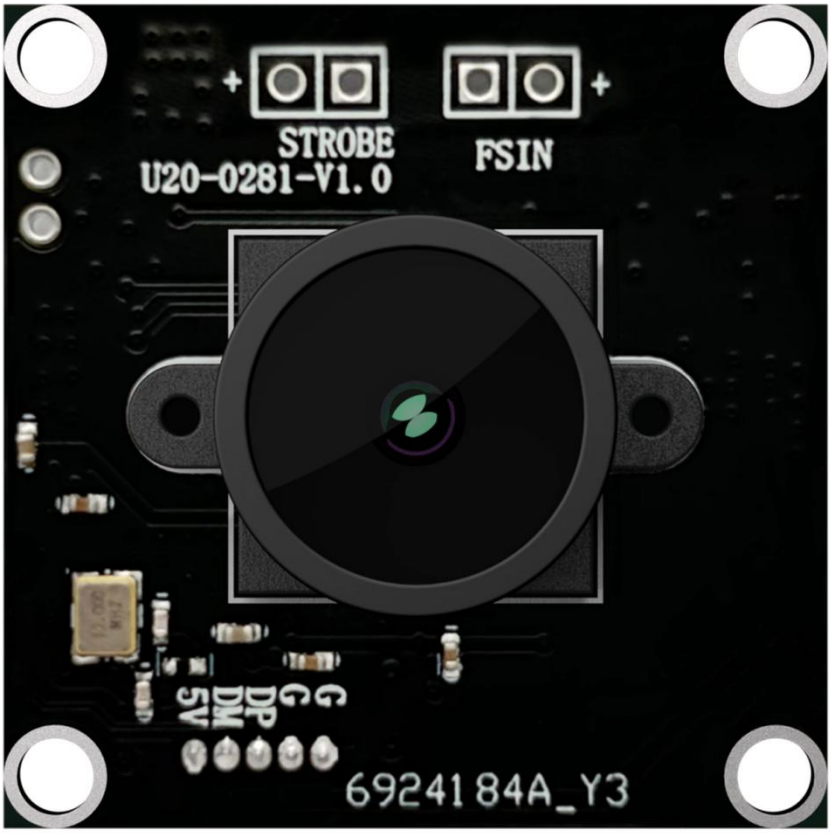
U20CAM-9281M



Normally We will update our development Manual here

Date	Revision	Change Details
2023/10/17	v1.0	First Released
2025/09/09	V1.1	Add software part chapter4
2025/10/31		Add script for raspberry pi

1 General

1.1 Description

U20CAM-9281M is InnoMaker UVC Series Module with 1MP 1/4" monochrome global shutter OV9281 image sensor,low distortion USB 2.0 camera.Feature with external hardware trigger and stobe function.Shoot high-speed moving objects in crisp sharp images. Avoid the rolling artifacts to get a much more accurate complete picture than the rolling shutter cameras. Reserved external trigger ports, support trigger via external signal.)

1.2 Features

- U20CAM-9281M is a 1M global shutter UVC camera module by mono sensor ov9281;
- Compatible with USB2.0,USB3.0 plug and play for Windows,Linux,Mac Os devices;
- Support hardware external trigger mode and live streaming mode;
- Easily wire external trigger pins and strobe pins by 2.0mm pin headers;
- Featured with wide angle fixed M12 LEN FOV Up to 148 degree;

1.3 Specification

Model Name	U20CAM-9281M
Dimension	32mmx32mm
Sensor	Monochrome global shutter OV9281
Pixel Size	3μm*3μm
Resolution	1MP 1280(H)x800(V) Frame rate MJPG Max 120fps,Default 30fps
Output Format	MJPG/YUY2
Len	FOV148°（H） M12 18mm Len Seat No IR filter, sensitive to IR
Input Voltage	Power:5V Current:86mA 0.42W
Shutter Mode	Global Shutter
Image Color	Monochrome
USB Interface	Vendor：1.25mm-5P ZZ-MS，Shouhan
Auto Parameters	White Balance（Manual Option）,Exposure（Manual Option）
Controllable Parameters	Brightness,Contrast,Hue,Saturation,Sharpness,Gamma,White Balance,Backlight Comp,Gain,Exposure,PowerLine Frequency,Low Light Compensation
Support OS	Windows, Linux, Mac Os with UVC Drivers Devices
Cable Length	1M

External Trigger	Support. Use UVC **Parameters** "Focus"
Working Conditions	Operating Temp: -20°C-70°C, Humidity:80-85%
MJPG Output Resolution	• 1280x800 120fps,30fps,15pfs,10fps • 1280x720 120fps,60fps,30fps,20fps,15pfs,10fps • 800x600 120fps,60fps,30fps,20fps,15pfs,10fps • 640x360 120fps,60fps,30fps,20fps,15pfs,10fps • 640x400 120fps,60fps,30fps,20fps,15pfs,10fps • 640x480 120fps,60fps,30fps,20fps,15pfs,10fps • 320x240 120fps,60fps,30fps,20fps,15pfs,10fps • 320x200 120fps,60fps,30fps
YUY2 Output Resolution	• 1280x80 10fps • 1280x720 10fps • 800x600 10fps • 640x400 30fps,20fps,15pfs,10fps • 640x480 30fps,20fps,15pfs,10fps • 320x240 60fps,30fps,20fps,15pfs,10fps • 320x200 60fps,30fps,20fps,15pfs,10fps

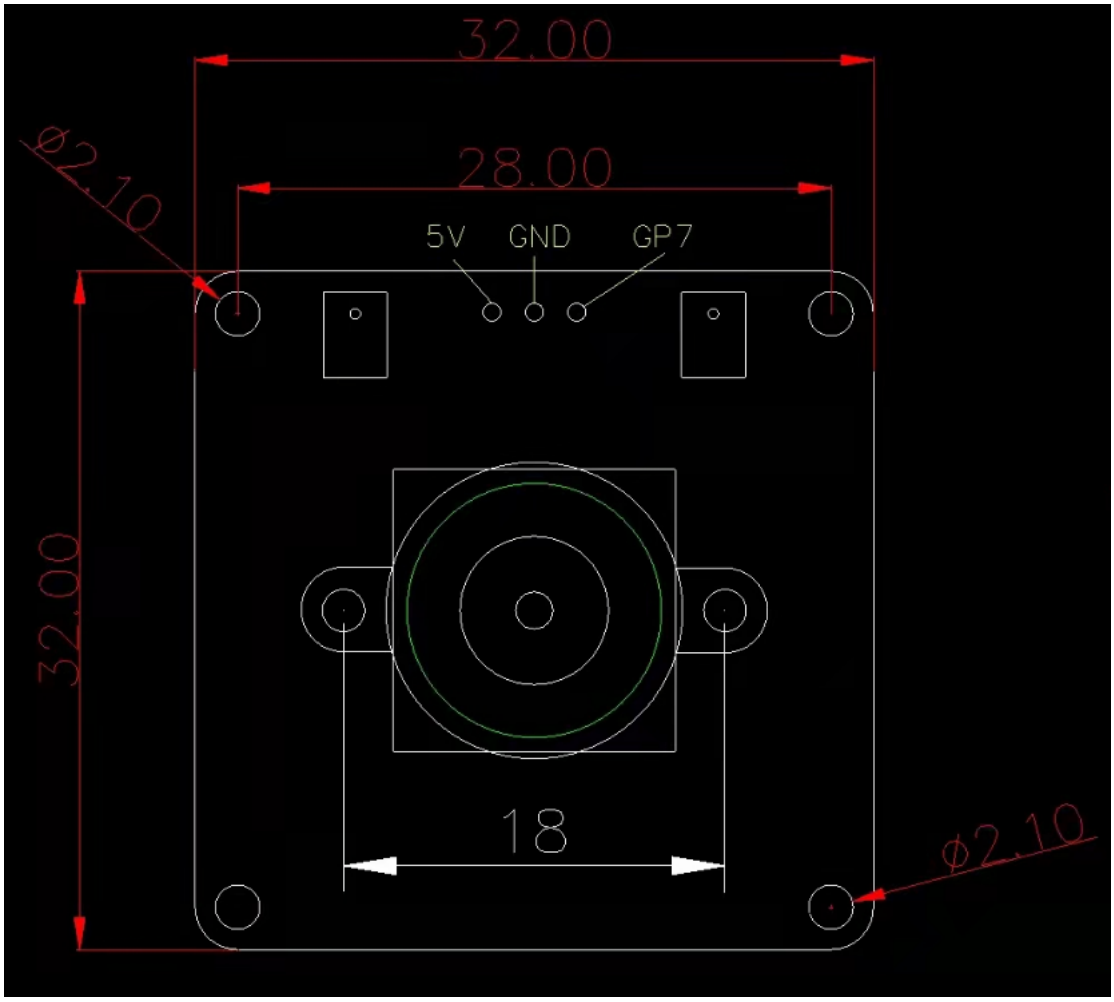
1.4 Resolution Frame Rate

Output Resolution And Frame Rate			
Output Format	**Resolution**	**Frame rate (FPS)**	**Maximum**
MJPG	1280x800	10,15,30,120	1280x800@120fps
	1280x720(720p)	10,15,20,30,60,120	
	800x600	10,15,20,30,60,120	
	640x360(360p)	10,15,20,30,60,120	
	640x400	10,15,20,30,60,120	
	640x480	10,15,20,30,60,120	
	320x240	10,15,20,30,60,120	
	320x200	120,60,30	
YUY2	1280x800	10	1280x800@10fps
	1280x720(720p)	10	
	800x600	10	
	640x400	10,15,20,30	
	640x480	10,15,20,30	
	320x240	10,15,20,30,60	

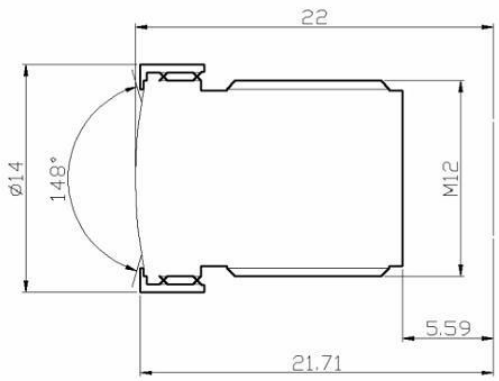
	320x200	10,15,20,30,60	

2 Hardware

2.1 Module Size

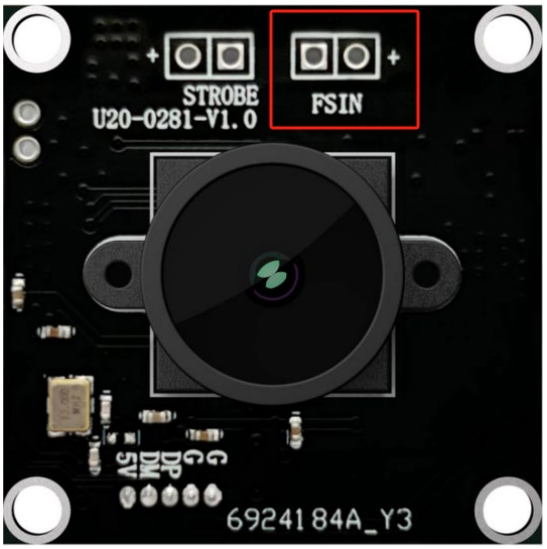


2.2 Camera LEN



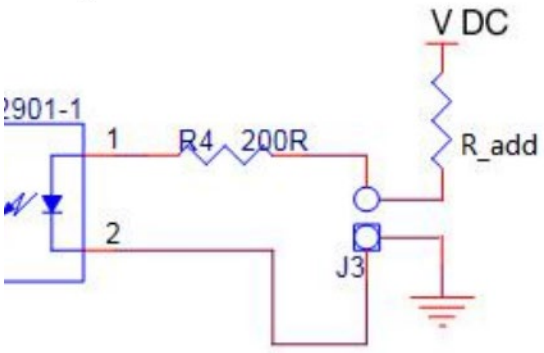
- Interface: M12
- Field of view Fov(D) = 148Degree
- Focal Length 2.8 mm
- Focal Distance Adjustable
- TV DISTORTION <-17%
- F(N) /Aperture 2.2
- Len Seat: 18mm

2.3 External Trigger Signal



PINS	Description
FSIN +	3.3v-5v External Trigger Input
FSIN -	External Ground GND

Reference Circuit



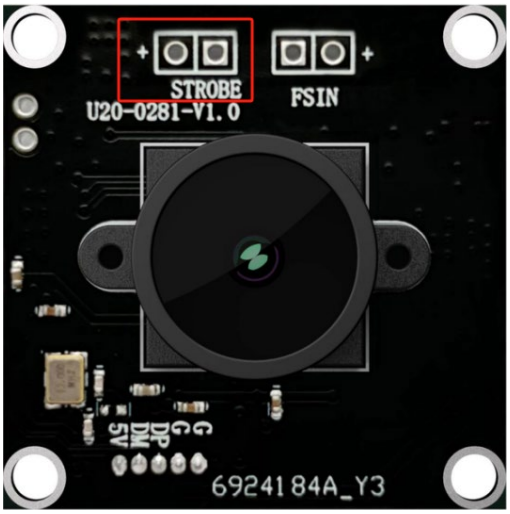
For example, VDC = 12V, Vf = 1.25V

The calculations done here are based on 12VDC. Please do follow these calculations for other voltages like 24VDC.

Let's take the current through IR LED I_f = 20mA.
Voltage drop across the IR LED = 1.25V
The value of Resistor $R_1 = (V_{cc}-V_f)/I_f = (12 – 1.25)/0.02 = 537.5\ \Omega$
Wattage of resistor $R_1 > I_f^2 * R_1 = 0.02^2*537.5 = 0.215W$
Wattage of the resistor R_1 selected should be greater than 0.215W.

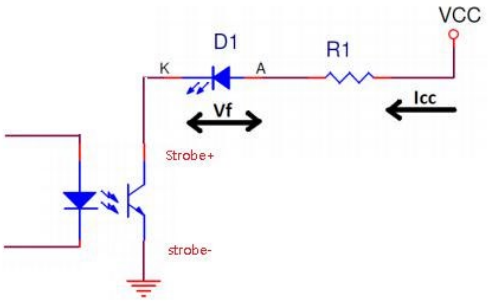
And there is a resistor on board(R4 = 200Ω), So the R_add = R1 - R4 = 537.5 - 200 = 337.5Ω

2.4 STROBE Signal



PINS	Description
STROBE +	Sensor STROB+
STROBE -	

Reference Circuit



On-board TLP281 optocoupler isolation, Notice the max collector current is 50mA.

Output Specifications

S. No	Parameter	Test Condition	Value			Unit
			Min	Typ	Max	
1	Driver Voltage (VCC)			12	24	V
2	Drive current (Icc)			10	50	mA
3	Collector Emitter Breakdown Voltage				80	V
4	Collector Emitter Saturation Voltage	Icc = 1 mA		0.1	0.2	V
5	Power Dissipation				150	mW

Collector-Emitter Saturation Voltage	$V_{CE(sat)}$	I_F = 10mA, I_C = 1mA		0.1	0.2	V

So If the current required to drive the Flash LED is no more than 50mA

The value of series resistor: R1 = (VCC- Vf - VCE) / If

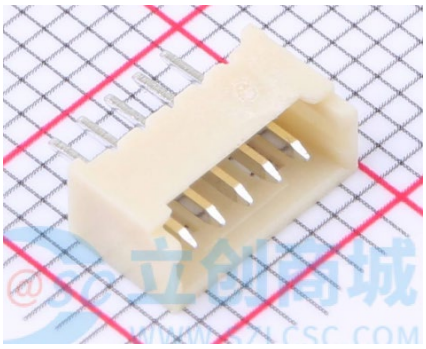
VCC: system Voltage

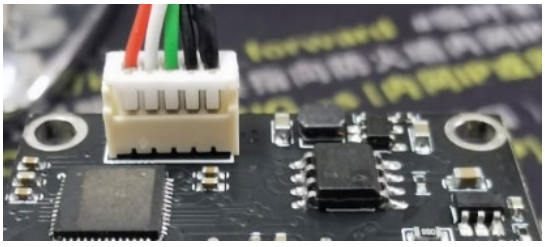
Vf: Forward voltage of Flash LED for current Icc

VCE: Collection Emitter voltage, typical:0.1V

If the current required to drive the flash exceeds 50mA, then it is required to drive it with the help of LED driver circuit, and LED driver circuit can be controlled by using the strobe output pin.

2.5 USB Connector

	Vendor： SHOU HAN(首韩) Name:1.25mm-5P ZZ-MS

1	5V	5V Power
2	DM	USB 2.0 Data-
3	DP	USB 2.0 Data+
4	GND	Ground
5	GND	Ground

3 External Trigger Model

The external trigger mode is to accepts the external input signal to trigger the image output. When the rising edge of the external trigger signal coming, it can output an image. Therefore, it is very suitable for capturing high-speed moving objects. In addition, the sensor enables the sleep state will greatly reduce the power consumption.

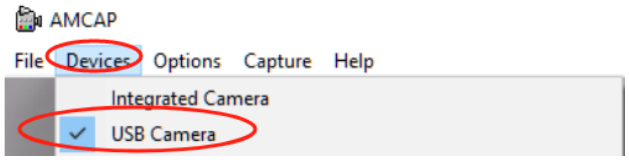
3.1 Enable Trigger Model

We use UVC Auto Focus and Auto exposure to control external trigger of ov9281 uvc camera module.

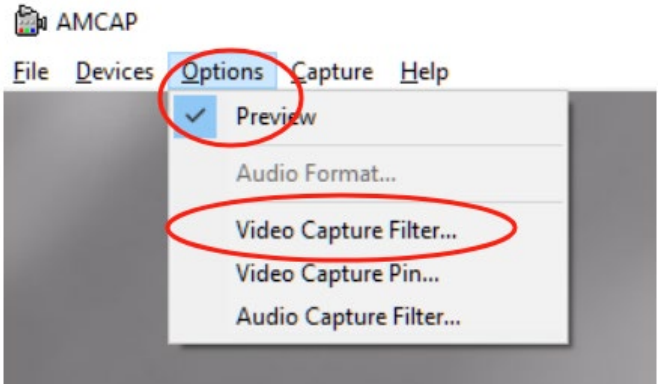
3.1.1 Check Auto focus and Uncheck Auto Exposure

We test it on AMCAP.EXE On windows, which we will provide it on github.
We set UVC Parameters **"Focus"** as the trigger Model Enable options.

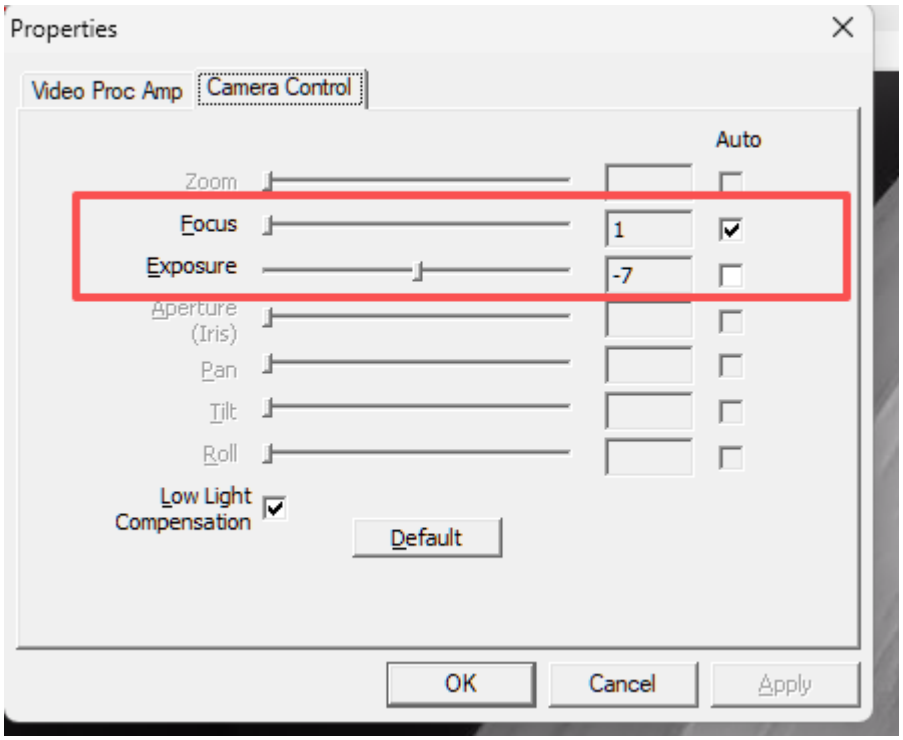
Open AMCAP.EXE, Choose **"USB Camera" from** "Devices"



From **"Options"** Choose **"Video Capture Filter"**



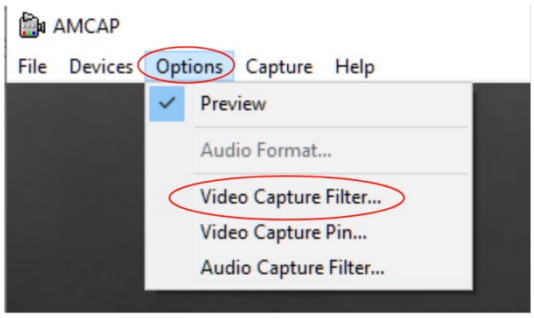
From **"Camera Control"** Find **"Focus"**, check Auto Focus, uncheck Auto Exposure. As below figure



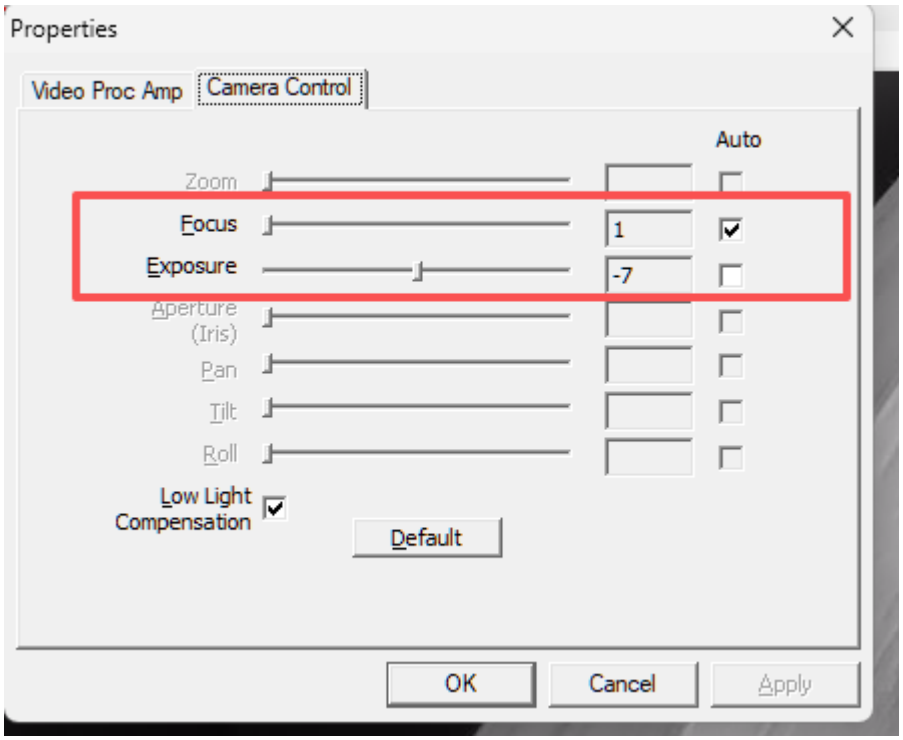
You can see the preview stop and enter

3.2 Adjust exposure Manually

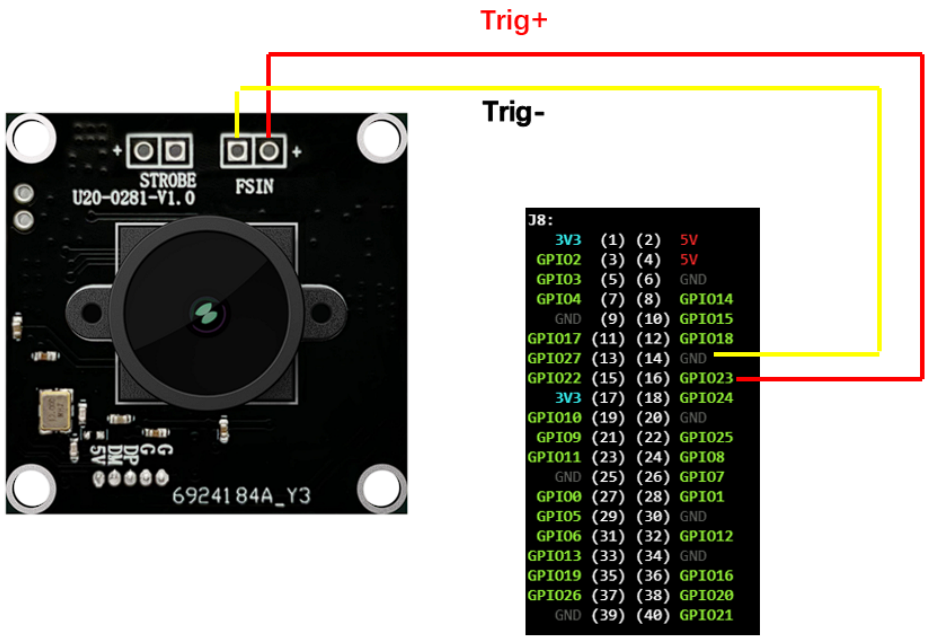
This is necessary for fast move object.



There is a switch behind the Exposure slide in the Camera Control. Select it to start manual exposure mode.



3.2 Wire Connection and Script On Raspberry PI

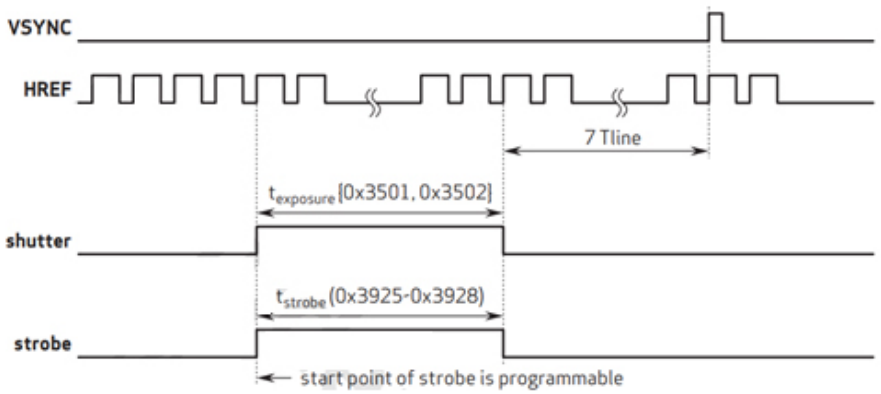


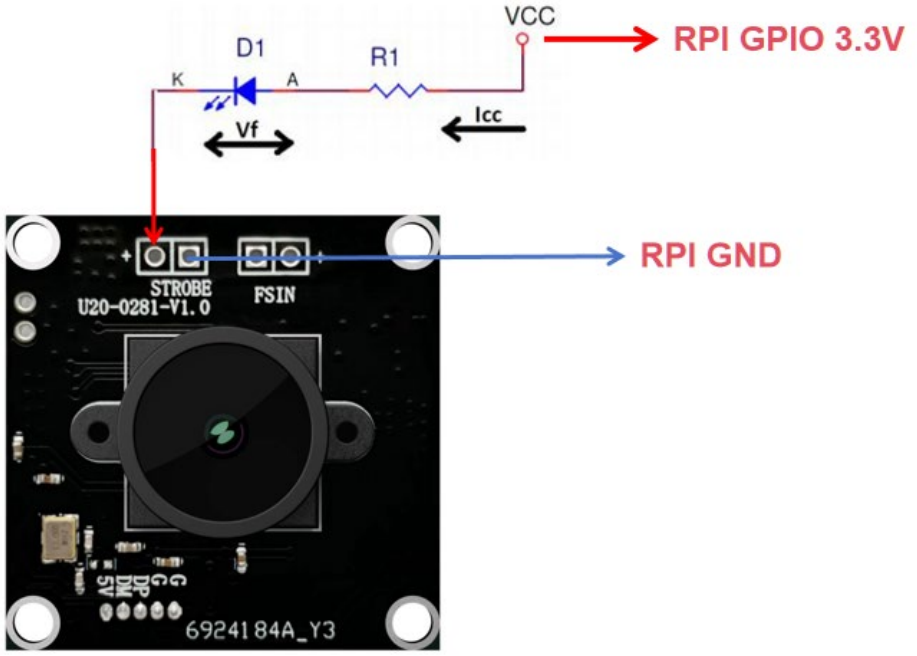
Our sample use for raspberry pi, more information please refer to chapter 2.3,We use Raspberry PI GPIO 23 generate 3.3V pulse signal.Connect Raspberry PI GPIO23 to FSIN+, GND To FSIN-,Run command to start. Script will provide on github.

```
while true;do
        gpioset gpiochip0 23=1
        sleep 0.005
        gpioset gpiochip0 23=0
        sleep 0.005
done
```

3.2 Strobe Connection

When the image is output, a flash signal output from S can drive flash to enhance exposure. The stroboscopic signal of the sensor can set the light point or time parameter. However, it can only output a fixed strobe signal because of a fixed UVC Camera configuration.


VSYNC
HREF
7 Tline
$t_{exposure}$ (0x3501, 0x3502)
shutter
t_{strobe} (0x3925-0x3928)
strobe
start point of strobe is programmable


VCC
RPI GPIO 3.3V
D1
R1
K
A
Vf
Icc
RPI GND
STROBE
FSIN
U20-0281-V1.0
6924184A_Y3

4 UVC Camera Software Manual

Date	Version	Description
2023-10-19	V1.0	First Released

4.1 Description

- UVC cameras comply with UVC protocol and work with web-camera applications out-of-box
- UVC Cameras support windows, linux, MacOs Compatible with UVC drivers

What is UVC Camera

- UVC Camera is camera with a USB interface that meets the standards set for the USB Video Class. This means that every UVC Camera is a USB camera, but not all USB cameras are UVC Cameras, because they might adopt the USB interface without meeting the Video class requirements.
- Therefore, a major advantage of the UVC cameras is their universal compatibility and flexibility. As they meet the video class standard, you can easily use them on different platforms with a USB port without handling the driver issue, like the Raspberry Pi or a smartphone. It also makes it easier for you to migrate your applications from one platform to another.
- At present, our UVC cameras support Windows, Linux, MAC, and Android systems, but do not support the iPhone system.

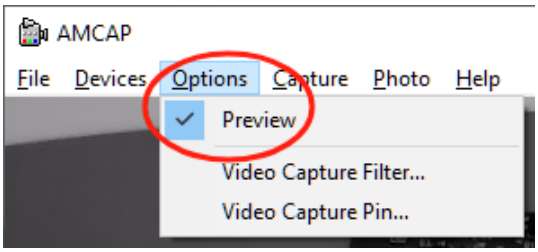
4.2 Works on Windows

4.2.1 AMCAP

AMCAP is a free and easily use UVC Camera test tools.

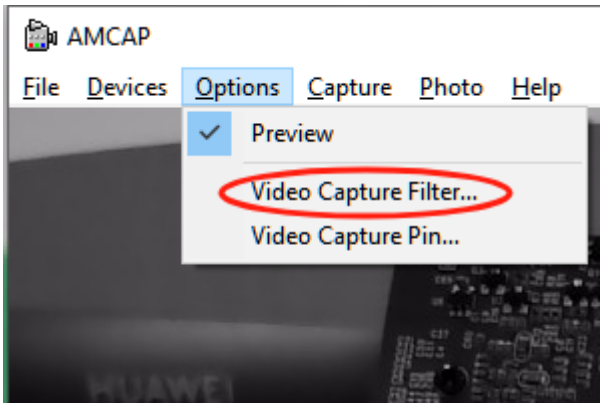
Preview

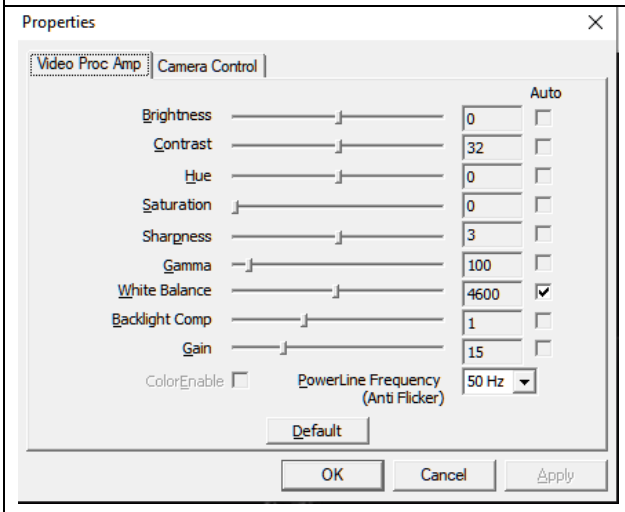
Open AMCAP.EXE, Select USB Camera From “Devices”, Select “Preview” from “Options”



Video Capture Filter

You Can find most of Controllable Parameters from “Options”, “Video Capture Filter”.

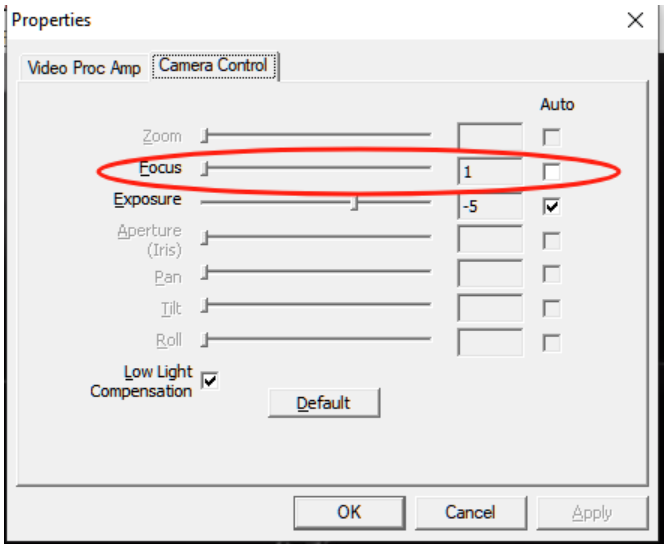




Brightness,Contrast,Hue,Saturation,Sharpness,Gamma,White Balance,Backlight Comp,Gain,Exposure,PowerLine Frequency,Low Light Compensation

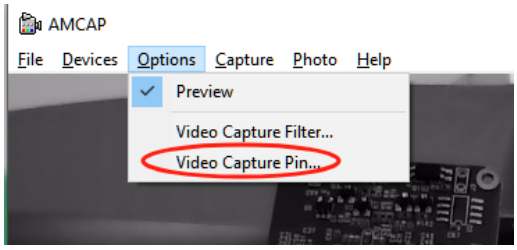
External Trigger Parameters

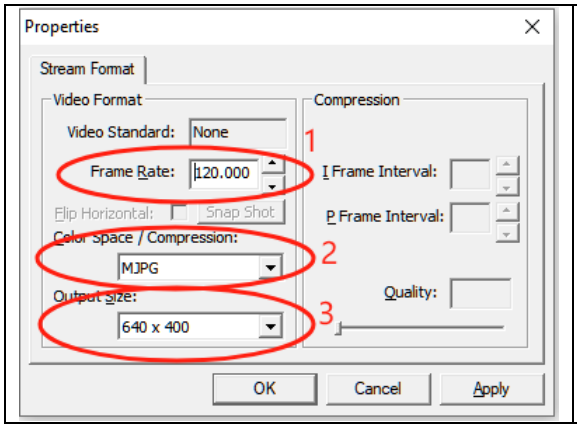
From “Video Capture Filter” “Camera Control” ,The “Focus” Parameter is for external trigger signal Enable.



Video Capture Pin

You Can find most of Controllable Parameters from “Options”, “Video Capture Pin”.





1	You can chage Frame rate
2	Choose Output format like MJPG/YUV2
3	Choose Resolution Camera Support

Status Bar

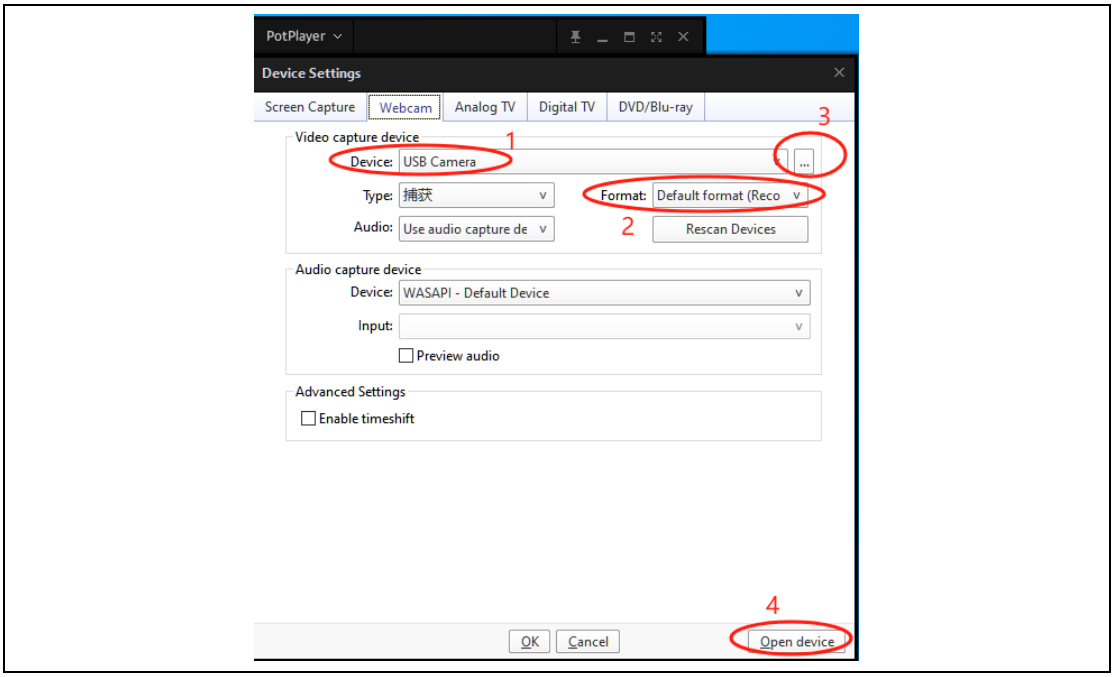
You can find live frame Rate, Output Resolution

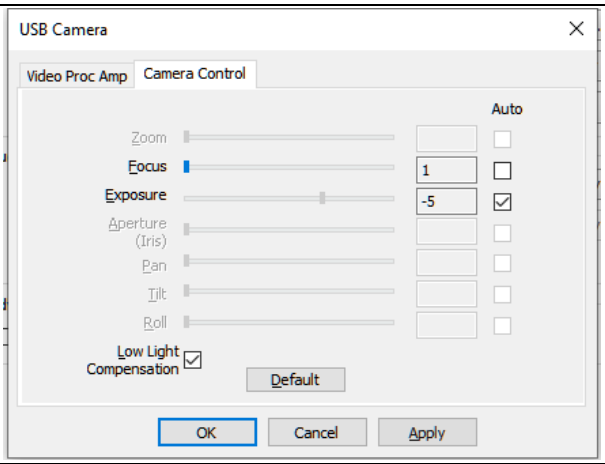
4.2.3 PotPlayer

Potplayer is another free Windows Tools which easily get video and images of UVC and U3V,UVC3.0 Cameras.

Open UVC Camera

Use Shortcut Key ALT+D open window as above

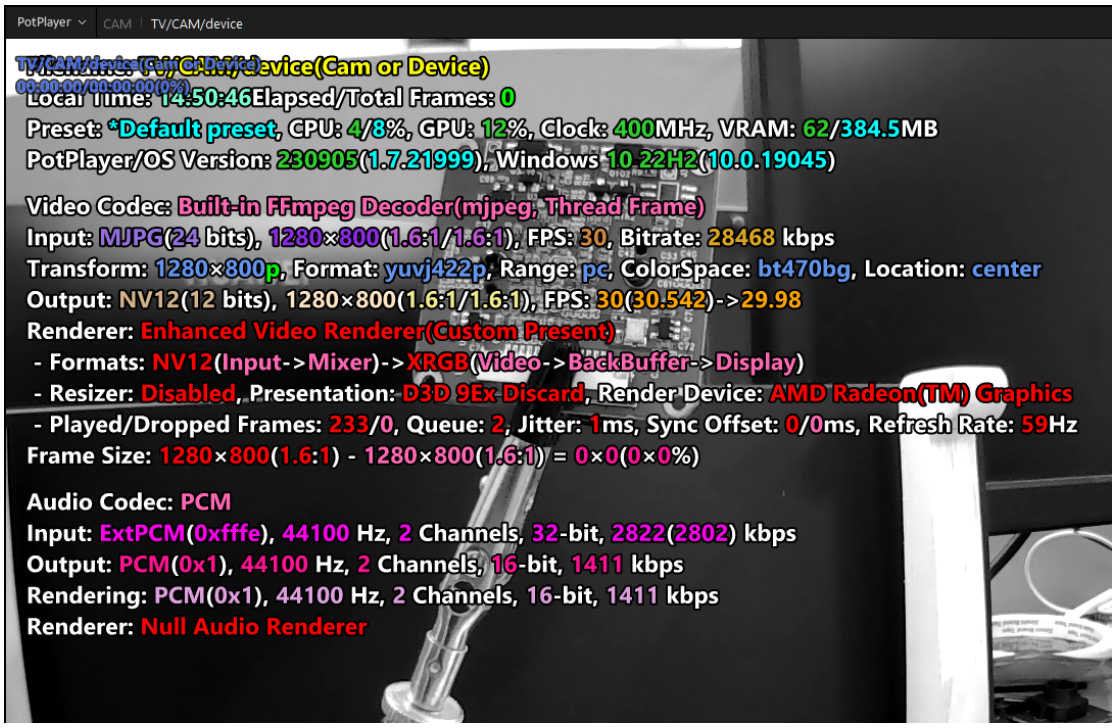




1	Choose UVC Camera Deivce
2	Choose Output format ,resolution,frame rate
3	Camera Parameters Settings
4	Open Device

Live Working Status

Use shortkey TAB Open window as below



4.2.4 OpenCV Python

Install Python3

Download from below link,check from cmd.exe after install successfully
https://www.python.org/downloads/release/

python --version
pip --version

```
C:\Users\zhouj>python --version
Python 3.11.6
```

```
C:\Users\zhouj>pip --version
pip 23.3 from C:\Users\zhouj\AppData\Local\Packages\P
packages\Python311\site-packages\pip (python 3.11)
```

Install numpy

pip install numpy

Install Opencv

```
pip install opencv-python
```

If you have error for installing, update your pip by below command:

```
python -m pip install --upgrade pip
```

Run OpenCV Python

Example1:

```
import cv2

cv2.namedWindow("preview")
vc = cv2.VideoCapture(0)

if vc.isOpened(): # try to get the first frame
    rval, frame = vc.read()
else:
    rval = False

while rval:
    cv2.imshow("preview", frame)
    rval, frame = vc.read()
    key = cv2.waitKey(20)
    if key == 27: # exit on ESC
        break

vc.release()
cv2.destroyWindow("preview")
```

Example2：

```
# import the opencv library
import cv2


# define a video capture object
vid = cv2.VideoCapture(0)

while(True):
```

```
        # Capture the video frame
        # by frame
        ret, frame = vid.read()

        # Display the resulting frame
        cv2.imshow('frame', frame)

        # the 'q' button is set as the
        # quitting button you may use any
        # desired button of your choice
        if cv2.waitKey(1) & 0xFF == ord('q'):
                break

# After the loop release the cap object
vid.release()
# Destroy all the windows
cv2.destroyAllWindows()
```

Cited information

You can refer to the below link for any updates:
https://stackoverflow.com/a/606154
https://www.geeksforgeeks.org/python-opencv-capture-video-from-camera/

4.3 Works on Linux

4.3.1 Guvcview

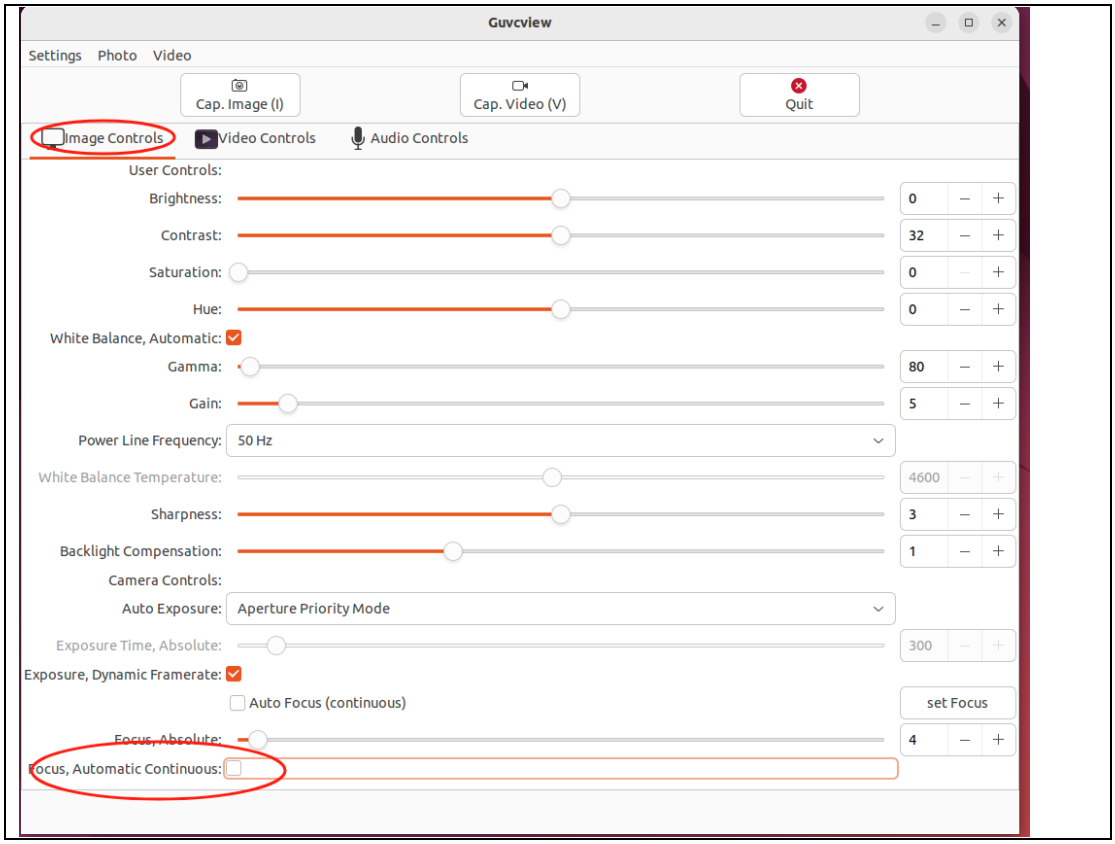
Install

Guvcview is free and easy operation tools for linux, Install and run :

```
sudo apt install guvcview
```

```
sudo guvcview
```

Image Controls

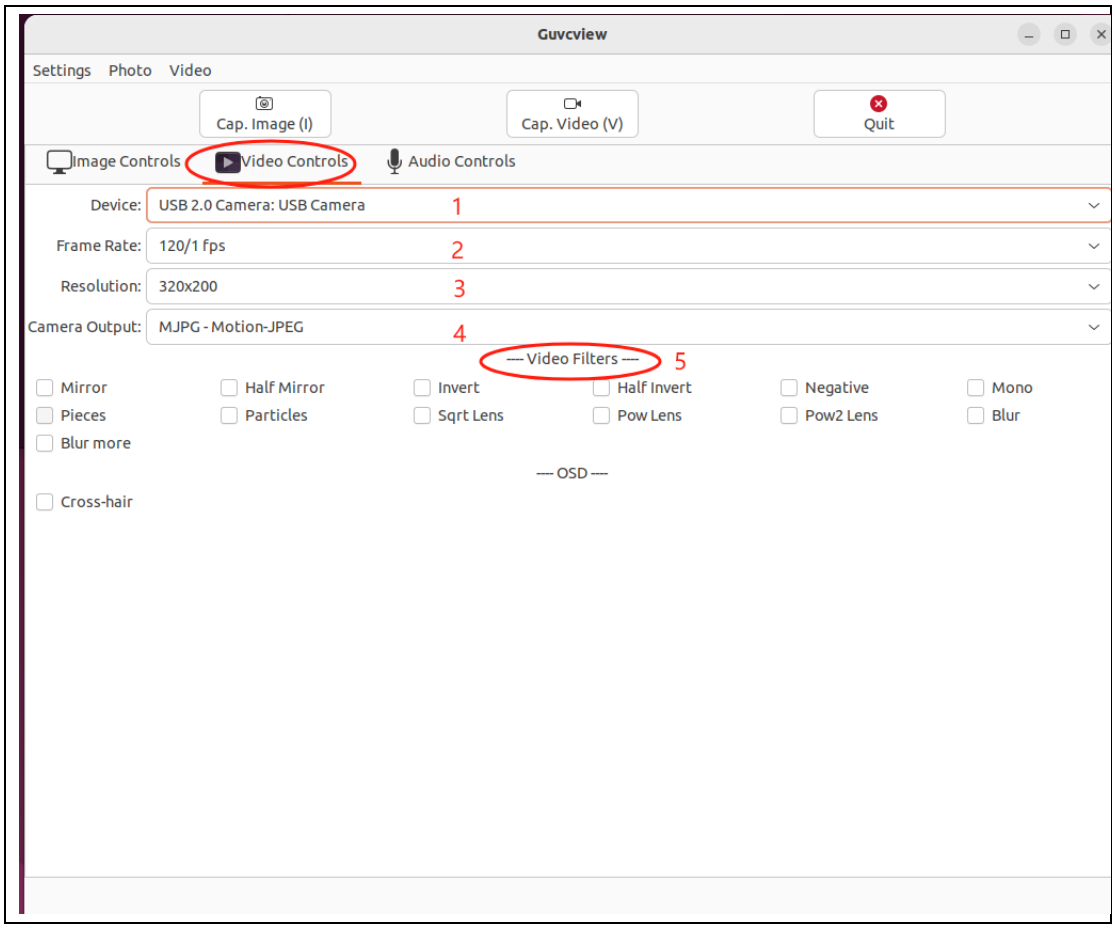


You can find the control parameters from Image Controls.

External Trigger Control

Focus,Automatic Continuous is for external trigger. check it to enable external trigger mode.

3.1.4 Video Controls



From Video Controls,

1	Select Device
2	Select Frame Rate
3	Select Resolution
4	Select Output format
5	Video Filters

4.3.2 qv4l2

Install

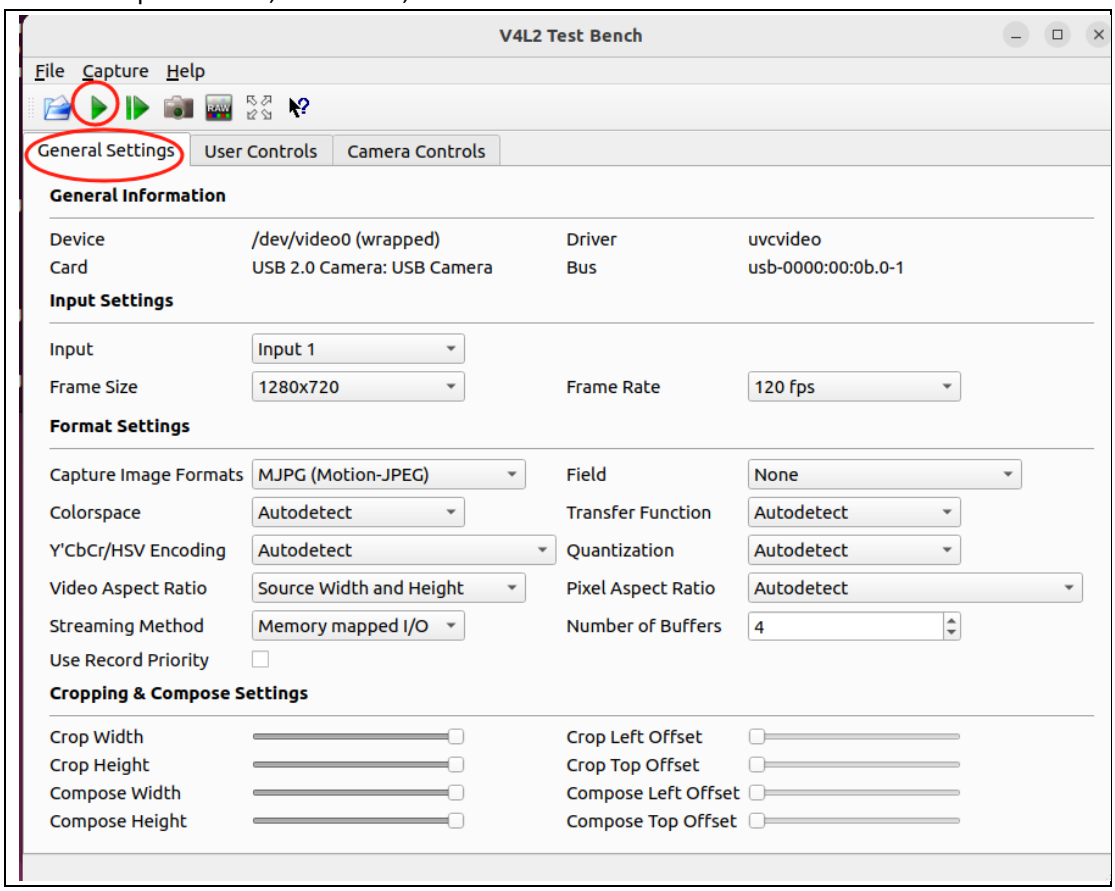
qv4l2 is free and easy operation tools for linux, Install and run :

```
sudo apt install qv4l2
```

```
sudo qv4l2
```

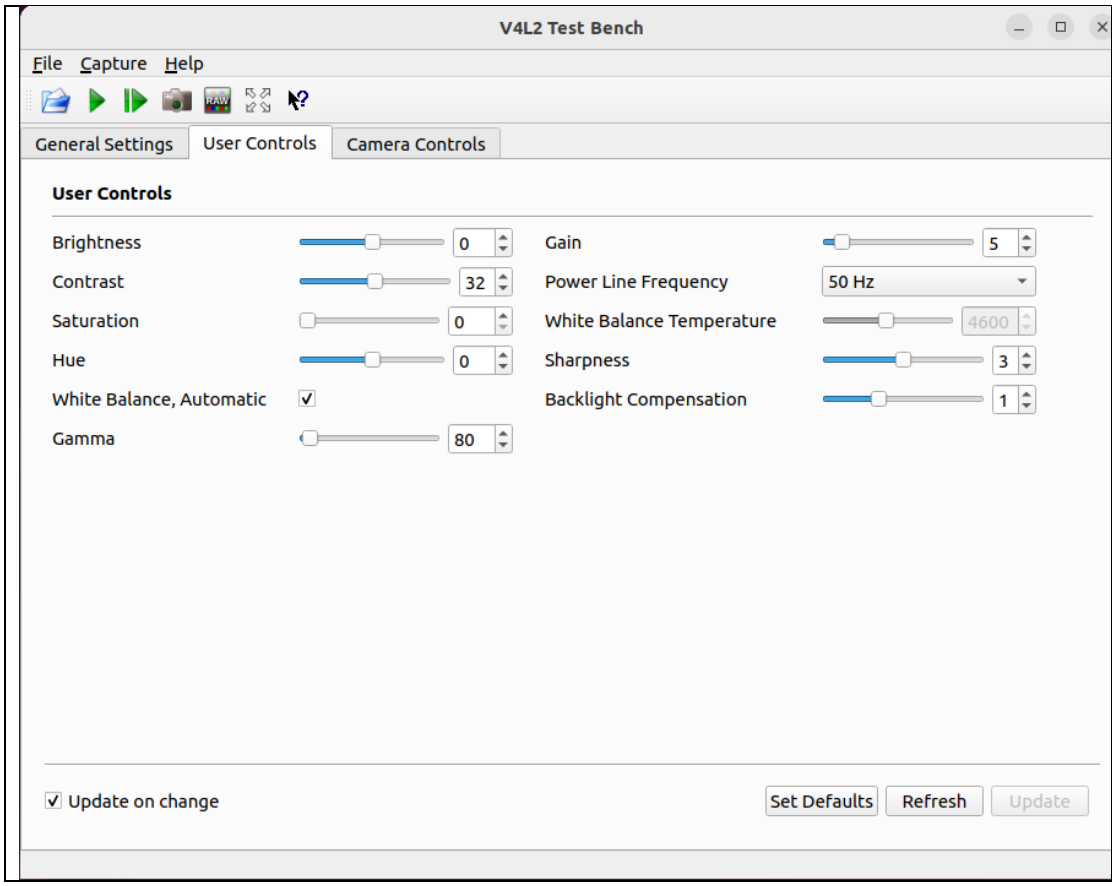
3.2.2 General Settings

Choose Output Devices, Resolution, Frame Rate



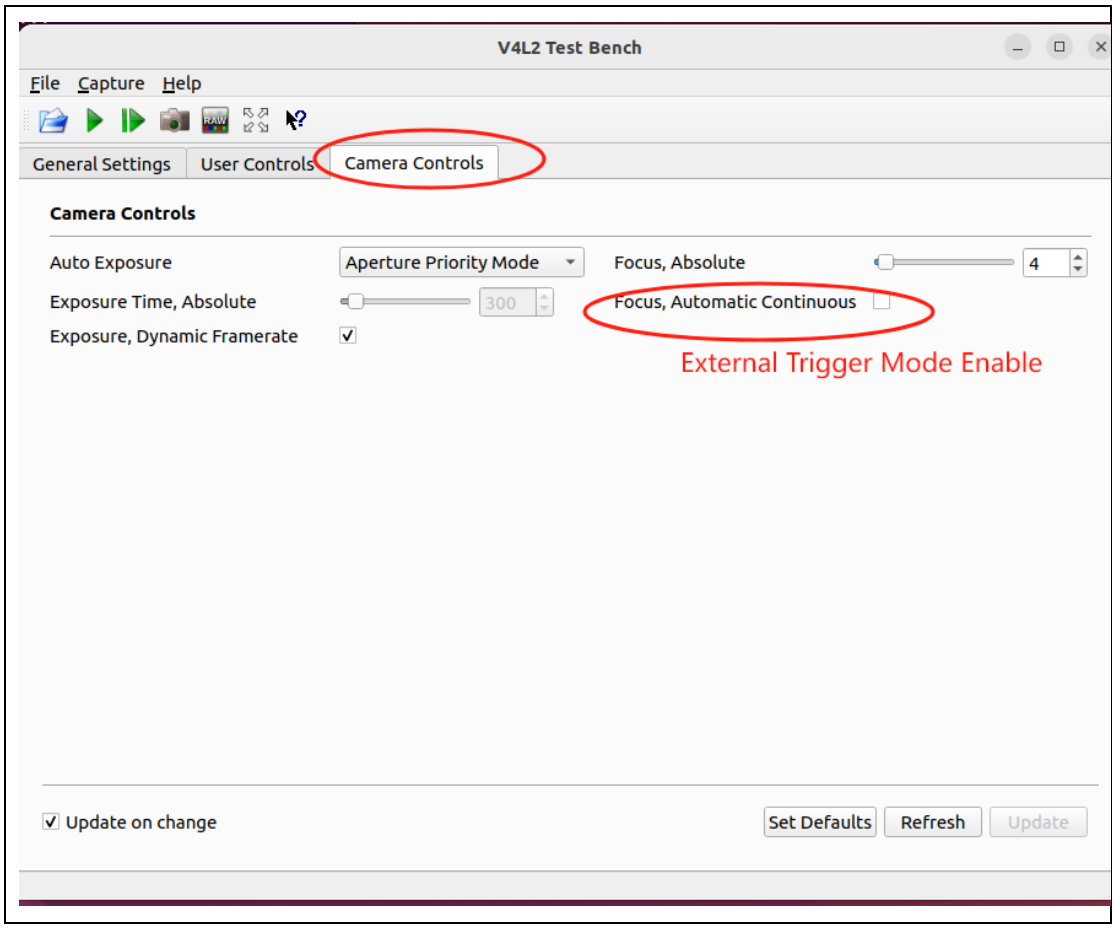
User Controls

control parameters



Camera Controls

You can check the External Trigger from this options.



4.3.3 V4L utility Tools

Install V4L utility packages

sudo apt-get update
sudo apt-get install v4l-utils

List UVC devices

v4l2-ctl --list-devices

```
joez@joez-VirtualBox:~$ v4l2-ctl --list-devices
USB Camera: USB Camera (usb-0000:00:0c.0-2):
        /dev/video0
        /dev/video1
        /dev/media0
```

List the supported formats

v4l2-ctl --list-formats -d

```
joez@joez-VirtualBox:~$ v4l2-ctl --list-formats -d 0
ioctl: VIDIOC_ENUM_FMT
        Type: Video Capture

        [0]: 'MJPG' (Motion-JPEG, compressed)
        [1]: 'YUYV' (YUYV 4:2:2)
```

List resolutions and frame

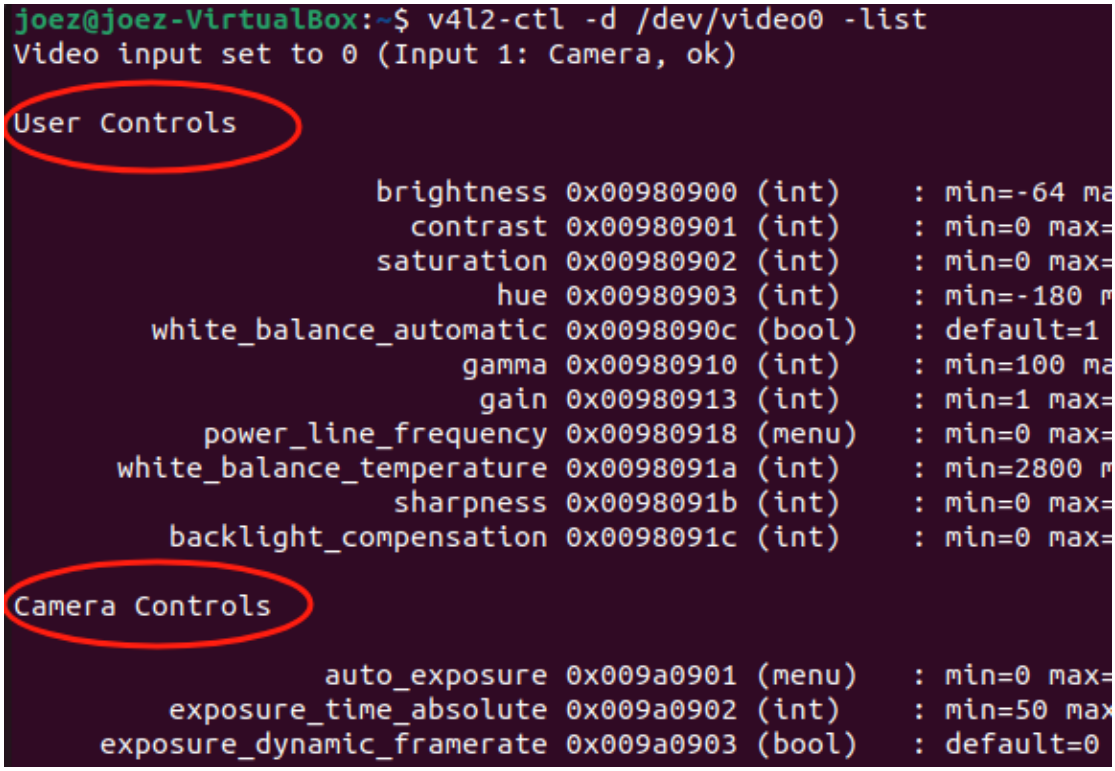
v4l2-ctl --list-formats-ext -d 0

```
joez@joez-VirtualBox:~$ v4l2-ctl --list-formats-ext -d 0
ioctl: VIDIOC_ENUM_FMT
        Type: Video Capture

        [0]: 'MJPG' (Motion-JPEG, compressed)
                Size: Discrete 640x480
                        Interval: Discrete 0.033s (30.000 fps)
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 800x600
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 1024x768
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 1280x720
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 1920x1080
                        Interval: Discrete 0.033s (30.000 fps)
        [1]: 'YUYV' (YUYV 4:2:2)
                Size: Discrete 1920x1080
                        Interval: Discrete 0.200s (5.000 fps)
                Size: Discrete 640x480
                        Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 800x600
                        Interval: Discrete 0.050s (20.000 fps)
                        Interval: Discrete 0.067s (15.000 fps)
                        Interval: Discrete 0.100s (10.000 fps)
                        Interval: Discrete 0.200s (5.000 fps)
                Size: Discrete 1024x768
                        Interval: Discrete 0.200s (5.000 fps)
                Size: Discrete 1280x720
                        Interval: Discrete 0.100s (10.000 fps)
                        Interval: Discrete 0.200s (5.000 fps)
                Size: Discrete 1280x1024
                        Interval: Discrete 0.200s (5.000 fps)
```

List Control parameters

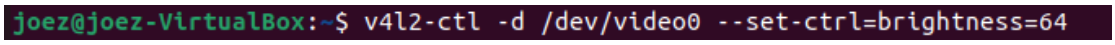
v4l2-ctl -d /dev/video0 -list



Set User/Camera controls

For example, set camera brightness to 64

v4l2-ctl -d /dev/video0 --set-ctrl=brightness=64



4.3.4 OpenCV Python

Install Opencv-Python

Check python pip version

python3 --version

pip --version

Run below command if not find the pip.

```
joez@joez-VirtualBox:~$ pip --version
Command 'pip' not found, but can be installed with:
sudo apt install python3-pip
```

sudo apt install python3-pip

Install opencv-python

sudo pip install OpenCV-python

* If you en count download errors sudo pip install opencv-python -i https://pypi.tuna.tsinghua.edu.cn/simple

Set user controls parameters.

Below code sample set brightness as 64, contrast as 0

```
import cv2

# open video0
cap = cv2.VideoCapture(0)

# The control range can be viewed through v4l2-ctl -L
cap.set(cv2.CAP_PROP_BRIGHTNESS, 64)
cap.set(cv2.CAP_PROP_CONTRAST, 0)

while(True):
    # Capture frame-by-frame
    ret, frame = cap.read()
```

```
        # Display the resulting frame
        cv2.imshow('frame', frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
                break

# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()
```

SAVE File name as 1.py, then run

```
sudo python3 1.py
```

Controlling values through code

```
import cv2
import time
# open video0
cap = cv2.VideoCapture(0)
cap.grab()

cap.set(cv2.CAP_PROP_AUTOFOCUS, 1)
time.sleep(2)
cap.set(cv2.CAP_PROP_AUTOFOCUS, 0)
time.sleep(2)
cap.set(cv2.CAP_PROP_FOCUS, 123)


cap.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)

while(True):
        # Capture frame-by-frame
        ret, frame = cap.read()
        # Display the resulting frame
        cv2.imshow('frame', frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
                break

# When everything done, release the capture
```

```
cap.release()
cv2.destroyAllWindows()
```

SAVE File name as 2.py, then run

```
sudo python3 2.py
```

Controlling values through UI interface

```
import cv2
import argparse
import configparser
from pathlib import Path
import time


parser = argparse.ArgumentParser()
parser.add_argument("-v", "--vid", default="0", help="Video sourse, default 0")
parser.add_argument(
    "-f", "--auto_focus", action="store_true", default=False, help="Turn on auto focus"
)
parser.add_argument(
    "-c",
    "--config",
    default="focus.ini",
    help="Focus config file, default focus.ini",
)
args = parser.parse_args()

try:
    vid = int(args.vid)
except ValueError:
    vid = args.vid

config_path = (Path(__file__).parent / Path(args.config)).resolve().absolute()
print("config file :", config_path)

config = configparser.ConfigParser()

config.read(config_path, encoding="utf-8")
```

```
cap = cv2.VideoCapture(vid)
cap.grab()
cap.set(cv2.CAP_PROP_AUTOFOCUS, 1)

if not args.auto_focus and config.has_section("Focus"):
    auto_focus = (
        config.getint("Focus", "auto_focus")
        if config.has_option("Focus", "auto_focus")
        else 1
    )
    focus = (
        config.getint("Focus", "focus")
        if config.has_option("Focus", "focus")
        else int(cap.get(cv2.CAP_PROP_FOCUS))
    )
else:
    auto_focus = 1
    focus = None
print("config auto_focus = %s" % auto_focus)
print("config focus = %s" % focus)
print("*" * 10)


if not auto_focus:
    cap.set(cv2.CAP_PROP_AUTOFOCUS, 0)

time.sleep(2)
if focus:
    cap.set(cv2.CAP_PROP_FOCUS, focus)

cv2.namedWindow("frame")


def set_auto_focus(x):
    cap.set(cv2.CAP_PROP_AUTOFOCUS, x)


cv2.createTrackbar(
    "0: OFF\r\n 1: ON\r\nauto_focus",
    "frame",
    int(cap.get(cv2.CAP_PROP_AUTOFOCUS)),
```

```
        1,
        set_auto_focus,
)


def set_focus(x):
        cap.set(cv2.CAP_PROP_FOCUS, x)


cv2.createTrackbar("focus", "frame", int(cap.get(cv2.CAP_PROP_FOCUS)), 1023, set_focus)

while cap.isOpened():
        # cap frame-by-frame
        ret, frame = cap.read()
        if not ret:
                break
        focus = int(cap.get(cv2.CAP_PROP_FOCUS))
        cv2.setTrackbarPos("focus", "frame", focus)

        af = int(cap.get(cv2.CAP_PROP_AUTOFOCUS))
        cv2.setTrackbarPos("0: OFF\r\n 1: ON\r\nauto_focus", "frame", af)

        cv2.imshow("frame", frame)

        if cv2.waitKey(1) & 0xFF == ord("q"):
                break

# When everything done, release the cap
cap.release()
cv2.destroyAllWindows()

if not config.has_section("Focus"):
        config.add_section("Focus")

print("set auto_focus = 0")
config.set("Focus", "auto_focus", "0")

print("set focus = %s" % focus)
config.set("Focus", "focus", str(focus))

config.write(open(config_path, "w"))
```

SAVE File name as cvtui.py, then run

```
sudo python3 cvtui.py
```

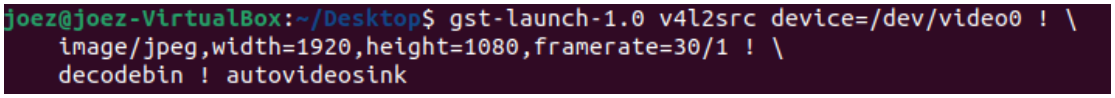
4.3.5 Gstreamer

GStreamer becomes a popular and powerful open-source multimedia framework to help users to build their own video streaming, playback, editing applications with various codec and functionalities on top of its high-level APIs.

Set Video Output Format

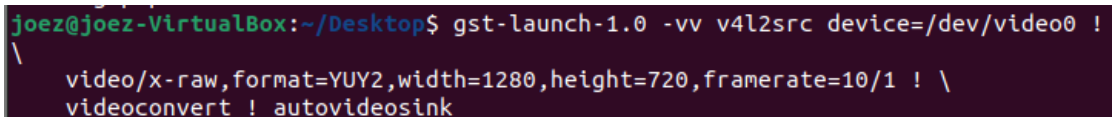
MJPEG

```
gst-launch-1.0 v4l2src device=/dev/video0 ! \
      image/jpeg,width=1920,height=1080,framerate=30/1 ! \
      decodebin ! autovideosink
```



YUV

```
gst-launch-1.0 -vv v4l2src device=/dev/video0 ! \
      video/x-raw,format=YUY2,width=1280,height=720,framerate=10/1 ! \
      videoconvert ! autovideosink
```



Streaming

MJPEG

```
# server
gst-launch-1.0 v4l2src device=/dev/video0 ! \
      image/jpeg,width=1280,height=720,framerate=30/1 ! \
      tcpserversink host=0.0.0.0 port=5001

# client
# change xxx.xxx.xxx.xxx to the actual ip address
```

```
gst-launch-1.0 -v tcpclientsrc host=xxx.xxx.xxx.xxx port=5001 ! \
      decodebin ! autovideosink
```

Save Video

```
gst-launch-1.0 v4l2src device=/dev/video0 !
image/jpeg,width=1280,height=720,framerate=30/1 ! jpegdec ! qtmux ! filesink
location=test.mp4 -e
```

Save Image

```
gst-launch-1.0 v4l2src device=/dev/video0 num-buffers=1 ! jpegenc ! filesink sync=false
location=file.jpg
```

Preview

```
gst-launch-1.0 v4l2src device=/dev/video0 !
image/jpeg,width=1280,height=720,framerate=30/1 ! jpegdec ! autovideosink
```

4.3.6 Read Serial Number

When you need to use multiple cameras,we need to use unique serial ID.

Linux udev

```
sudo udevadm info --query=all /dev/video0 | grep 'VENDOR_ID\|MODEL_ID\|SERIAL_SHORT'
```

```
joez@joez-VirtualBox:~/Desktop$ sudo udevadm info --query=all /dev/video0 | grep
 'VENDOR_ID\|MODEL_ID\|SERIAL_SHORT'
E: ID_VENDOR_ID=0bda
E: ID_MODEL_ID=3035
E: ID_SERIAL_SHORT=200901010001
```

4.4 More Software

- (1) OpenCV (opencv-python): OpenCV is an open-source computer vision library that allows easy access to UVC cameras via cv2.VideoCapture. Official documentation link: https://opencv.org/ PyPI installation link: https://pypi.org/project/opencv-python/
- (2) PyUVC: PyUVC is a Python library for accessing UVC cameras that interacts directly

with the UVC protocol. GitHub repository: https://github.com/pyuvc/pyuvc PyPI installation link: https://pypi.org/project/pyuvc/

- (3) VideoCapture (Python wrapper for V4L2):If you're working on a Linux system, VideoCapture is a simple interface that allows you to interact with cameras via V4L2. GitHub repository: https://github.com/charlesw/VideoCapture
- (4) libuvc:libuvc is an open-source UVC (USB Video Class) driver that provides APIs to control UVC devices. Python wrappers can be used via ctypes or cffi. GitHub repository: https://github.com/libuvc/libuvc