



U20CAM-IMX335 User Manual



Normally We will update our development Manual here

Date	Revision	Change Details
2025/09/10	v1.0	First Released

	U20CAM-IMX335 IMX335 HDR Camera Module UVC 2.0 Series
---	--

1 General

1.1 Description

U20CAM-IMX335 is a 5MP resolution camera module with STARVIS IMX335 CMOS sensor, image quality is very high and stable.

Features like a dynamic range up to 72 dB, advanced HDR functions and digital overlap HDR further improve the image quality and contrast behavior allowing for more detailed image analysis.

These features when combined with the low dark current performance and no smear further improve its performance in applications looking at both bright and dark regions at once.

The current module size is 32MM*32MM after your testing, we can adjust module size & shape according to your need, it will require some engineering cost. Compatible with Raspberry Pi ,Jetson Nano,PC Computers, Windows Linux Android Mac OS with UVC Driver.

1.2 Features

- 5MP resolution with IMX335 CMOS sensor, STARVIS technology with high sensitivity performance, HDR (high dynamic range) function improves image quality and contrast behavior Excellent low illumination feature
- Excellent image quality, Low power consumption
- Plug And Play: UVC-compliant, support Windows, Linux, Mac OS and Android device with UVC, just connect the camera to PC, laptop or Raspberry Pi with the USB cable without extra drivers to be installed.

	U20CAM-IMX335 IMX335 HDR Camera Module UVC 2.0 Series
---	--

2 Specification

Sensor	• 1/2.8" IMX335 • 5 Mega Pixel • 2592(H) x 1944(V) • 2.0μm x 2.0μm • Color
Shutter Type	Electronic rolling shutter
Dynamic range (HDR)	72dB
Sensitivity	>= 2000mV
Focus type	Fixed focus / Manually adjustable
Output Format	MJPEG/YUY2
Output Resolution	MJPEG Resolution & Frame rate • 1920x1080@30fps • 2592x1944@30fps,25fps,20fps,15fps,10fps,5fps,2fps • 800x600@30fps • 1024x768 @30fps • 1024x720@30fps • 1280x920 @ 30fps • 2048x1536 @25fps • 320x240 @ 30fps • 160x120 @ 30fps • 640x480 @ 30fps
	YUY2 Resolution &Frame rate • 1920x1080@3fps • 1024x720@8fps • 800x600@20fps • 960X540@15fps • 2592x1944@2pfs • 1280x920@8fps • 640x480@30fps • 2045x1536@3fps
Auto Parameters	Auto Exposure Control (AEC) /Auto White Balance (AEB) /Auto Gain Control (AGC)
Controllable	Brightness,Contrast,Hue,Saturation,Sharpness,Gamma,White

Support: support@inno-maker.com

Sales : sales@inno-maker.com

3 / 28

Website: www.inno-maker.com

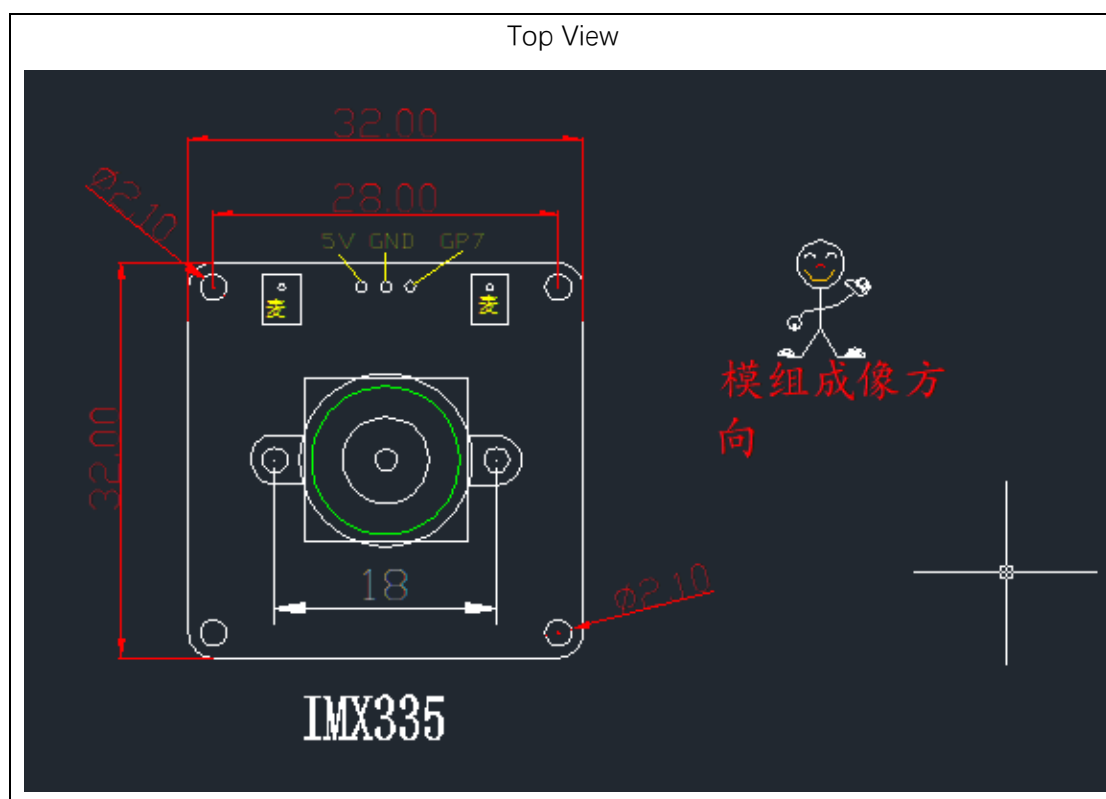
Github: <https://github.com/INNO-MAKER>

	U20CAM-IMX335 IMX335 HDR Camera Module UVC 2.0 Series
---	--

Parameters	Balance,Backlight Comp, Gain, Exposure ,PowerLine Frequency,Low Light Compensation
USB Type	USB2.0 High Speed USB2.0 OTG USB Video Class (UVC)
Cable Length	1M
Support OS	Windows,Mac Os,Android,Linux with UVC Driver
Working Condition	- Power:DC 5V, 500mA - Operating Temperature:0°C-70°C, - Storage temperature: -20°C to 70°C - Dimension:32mmx32mm

3 Hardware

3.1 Module Size



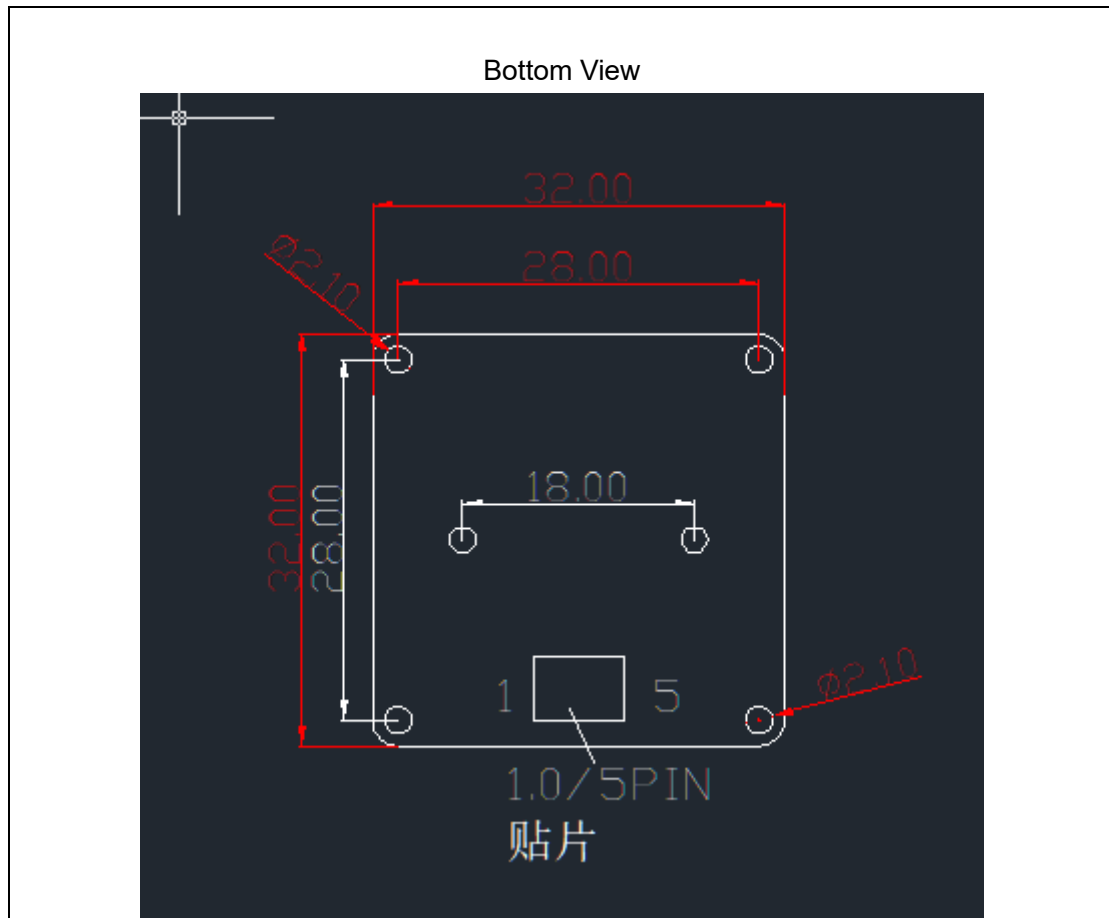
Support: support@inno-maker.com

Sales : sales@inno-maker.com

4 / 28

Website: www.inno-maker.com

Github: <https://github.com/INNO-MAKER>



3.2 USB Connector

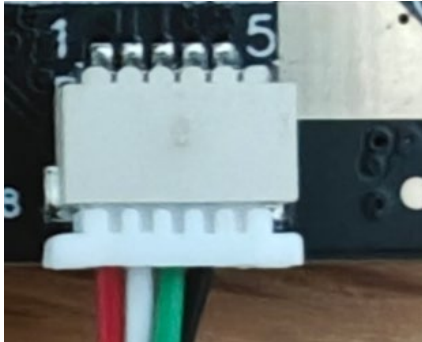


Vendor: JST
Name: SM05B-SRSS-TB(LF)(SN)



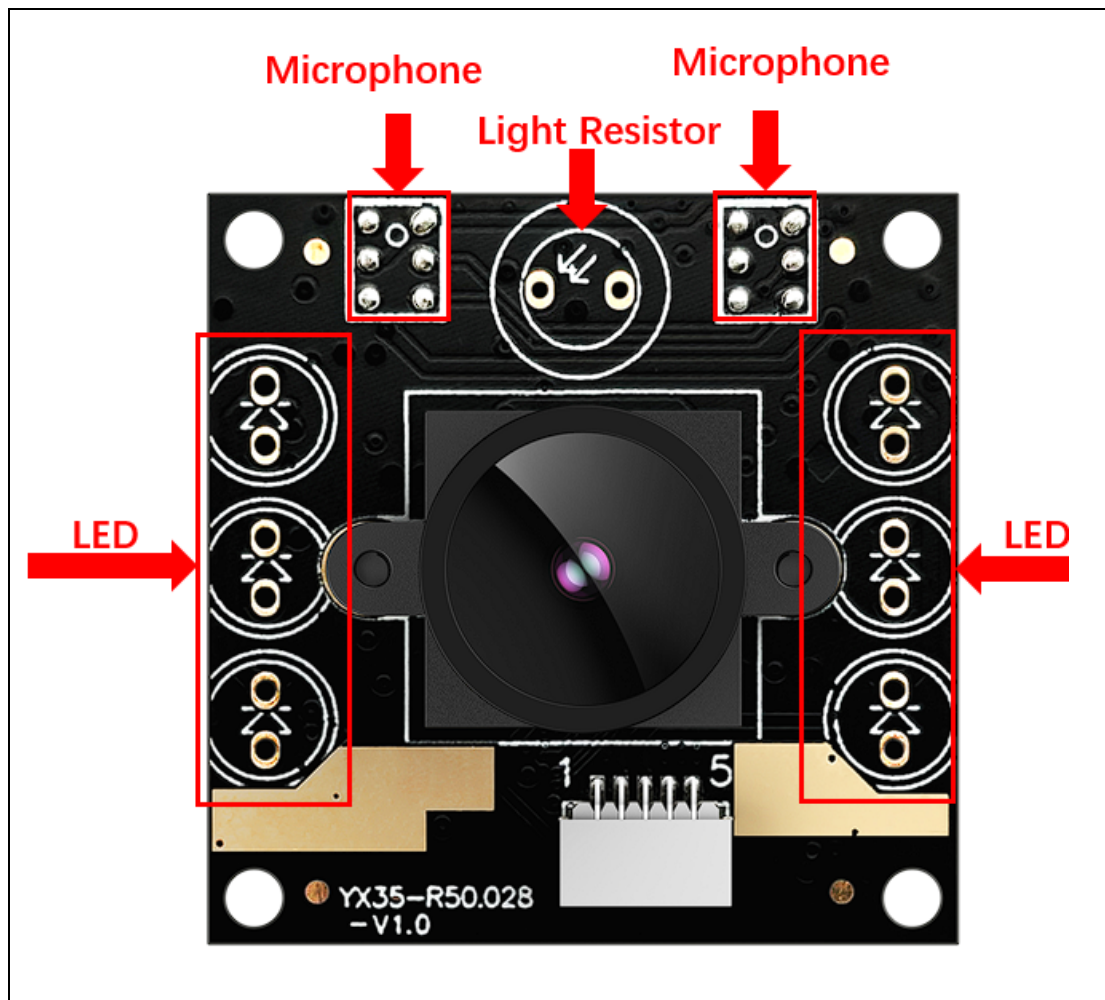
U20CAM-IMX335

IMX335 HDR Camera Module
UVC 2.0 Series



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

3.3 Reserve Pins



3.4 Lens Specification

F/N 2.3

Focus Length 3mm

CRA 14°

Distortion 25%

FOV 135 (H) x 110(V)x 58(D)

Lens Seat distance 18mm

Fixed Type Fixed Len

IR 650±10nm

4 UVC Camera Software Manual

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.

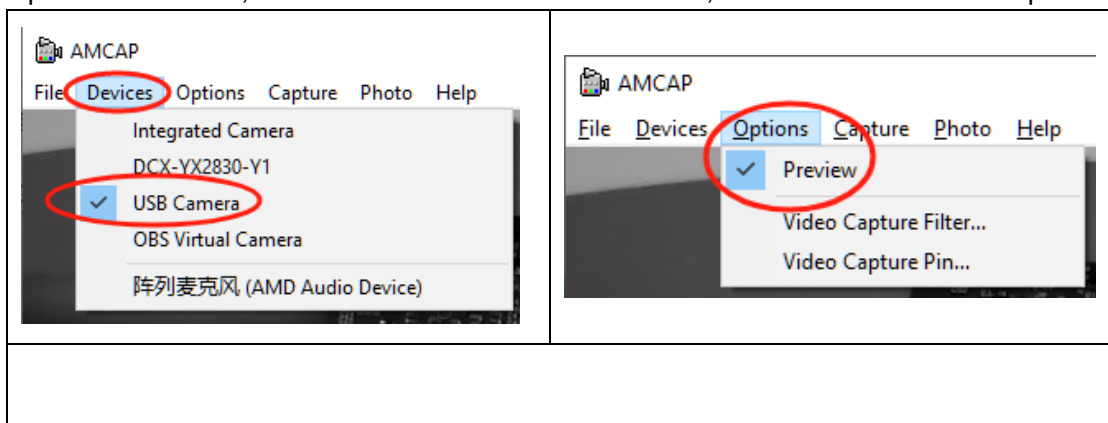
4.2 Works on Windows

4.2.1 AMCAP

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

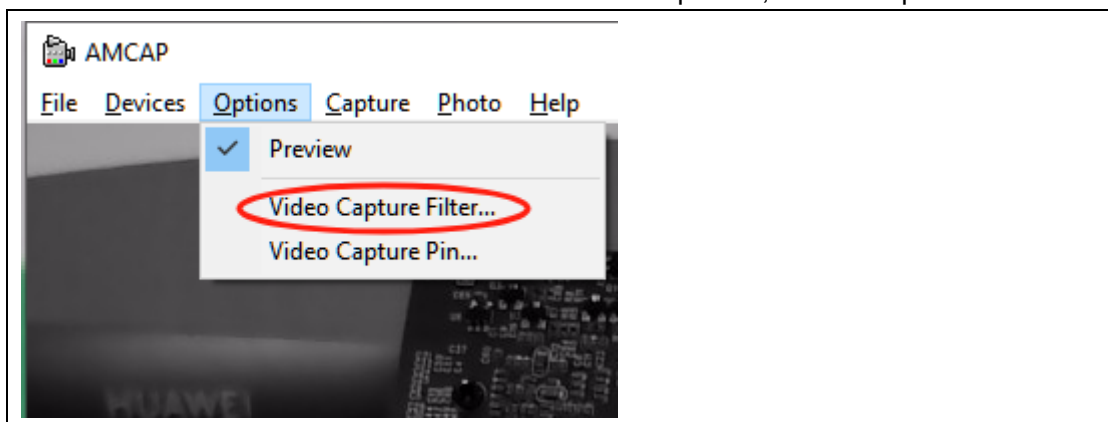
Preview

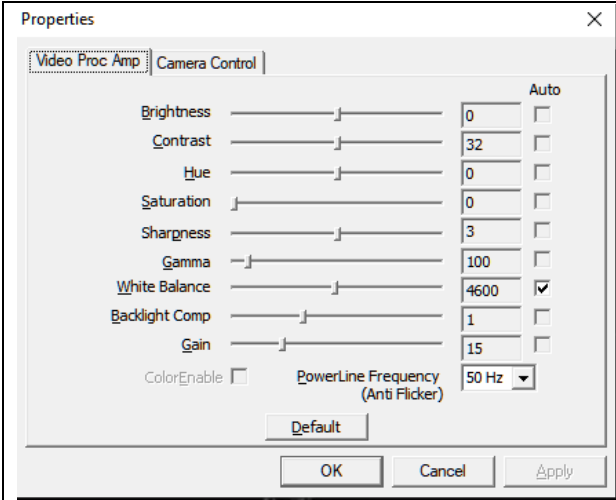
Open AMCAP.EXE, Select USB Camera From “Devices”, Select “Preview” from “Options”



Video Capture Filter

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

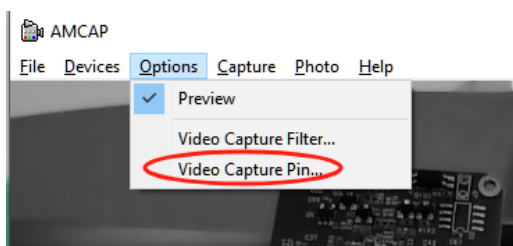


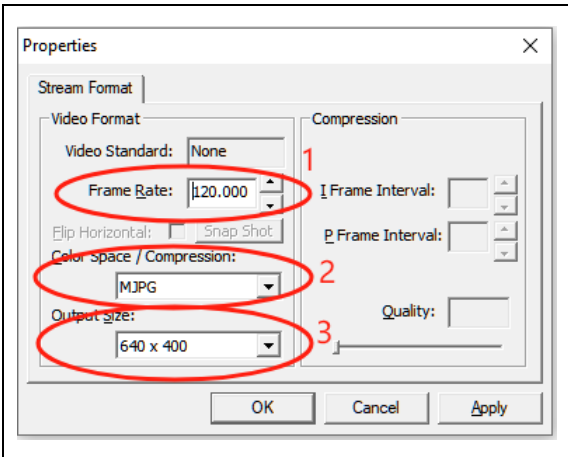


Brightness, Contrast, Hue, Saturation, Sharpness, Gamma, White Balance, Backlight Comp, Gain, Exposure, PowerLine Frequency, Low Light Compensation

Video Capture Pin

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

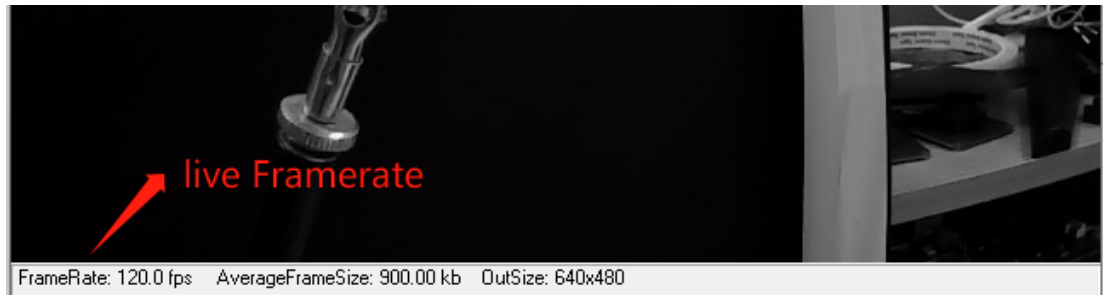




1	You can chage Frame rate
2	Choose Output format like MJPG/YUV2
3	Choose Resolution Camera Support

Status Bar

You can find live frame Rate, Output Resolution

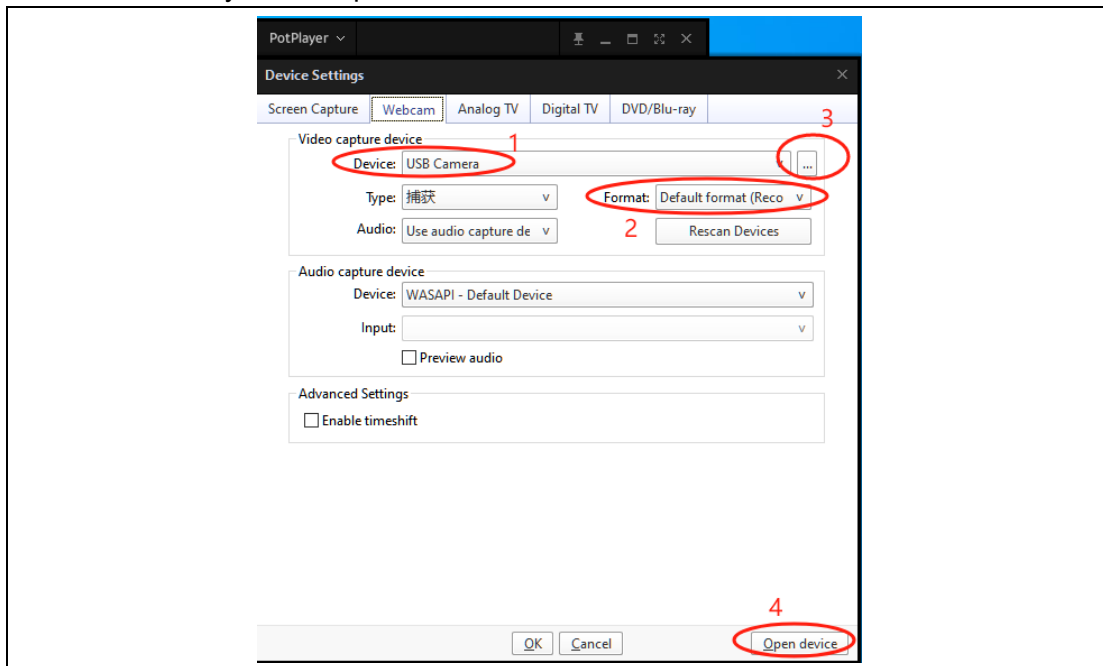


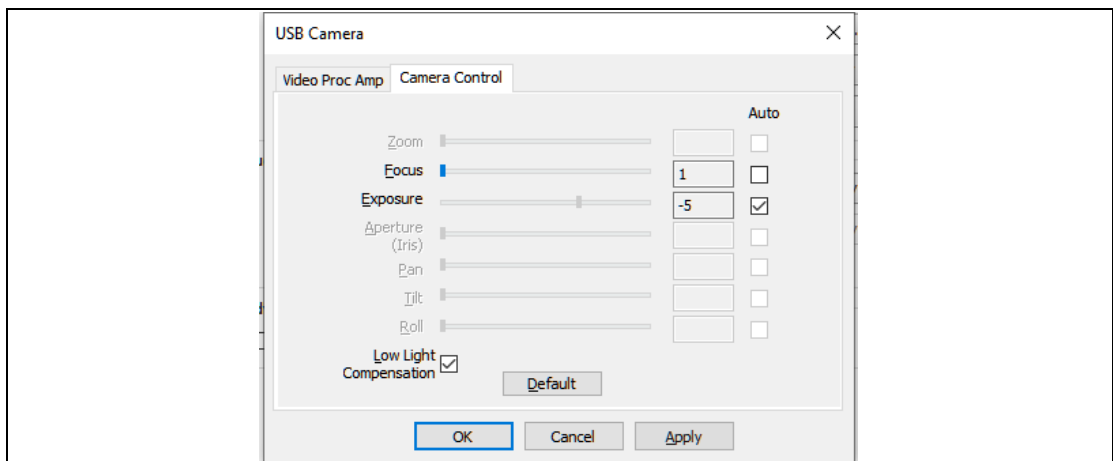
4.2.3 PotPlayer

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

Live Working Status

Use shortcutkey 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\Python311\site-packages\pip (python 3.11)
```

Install numpy

```
pip install numpy
```

Support: support@inno-maker.com

Sales : sales@inno-maker.com

12 / 28

Website: www.inno-maker.com

Github: <https://github.com/INNO-MAKER>



U20CAM-IMX335

IMX335 HDR Camera Module
UVC 2.0 Series

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.destroyAllWindows("preview")
```

Example2:

```
# import the opencv library
import cv2

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

while(True):
```

Support: support@inno-maker.com

Sales : sales@inno-maker.com

13 / 28

Website: www.inno-maker.com

Github: <https://github.com/INNO-MAKER>



U20CAM-IMX335

IMX335 HDR Camera Module
UVC 2.0 Series

```
# 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 Gvvcview

Install

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

```
sudo apt install gvvcview
```

Support: support@inno-maker.com

Sales : sales@inno-maker.com

14 / 28

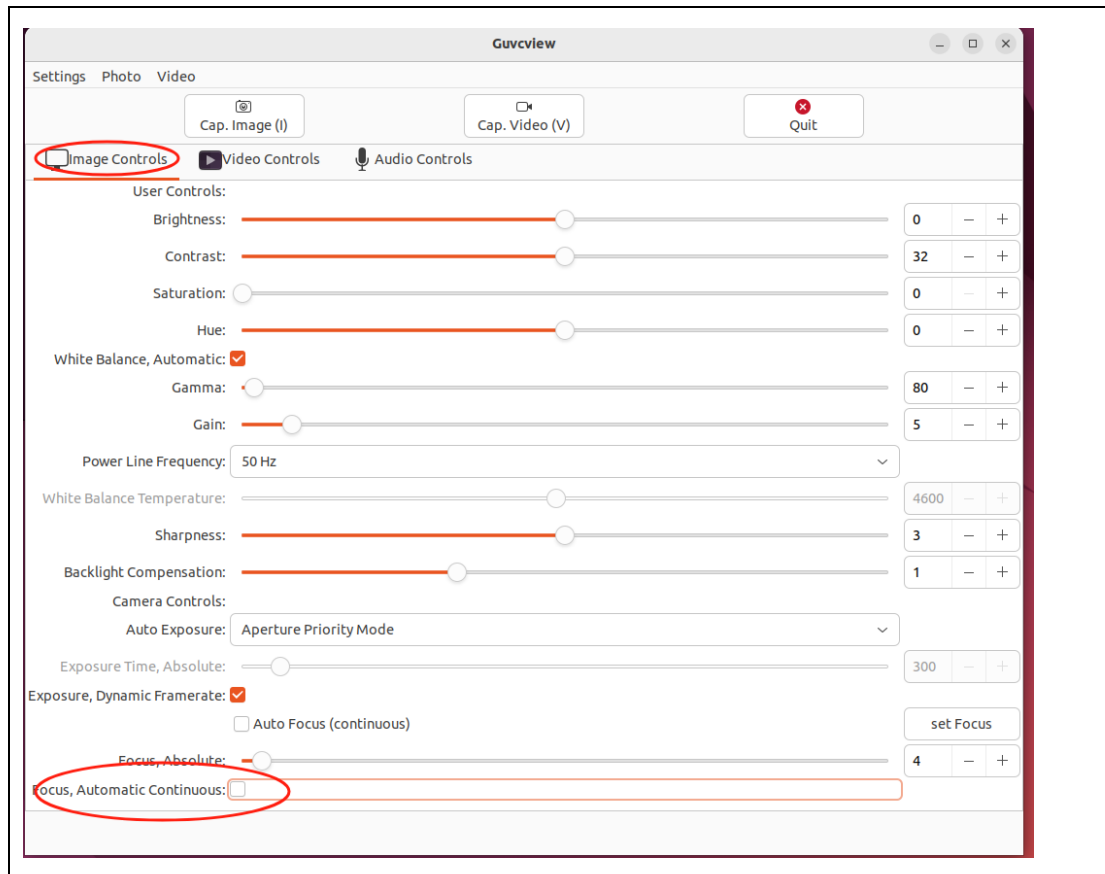
Website: www.inno-maker.com

Github: <https://github.com/INNO-MAKER>



```
sudo guvcview
```

Image Controls



You can find the control parameters from Image Controls.

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

Support: support@inno-maker.com

Sales : sales@inno-maker.com

15 / 28

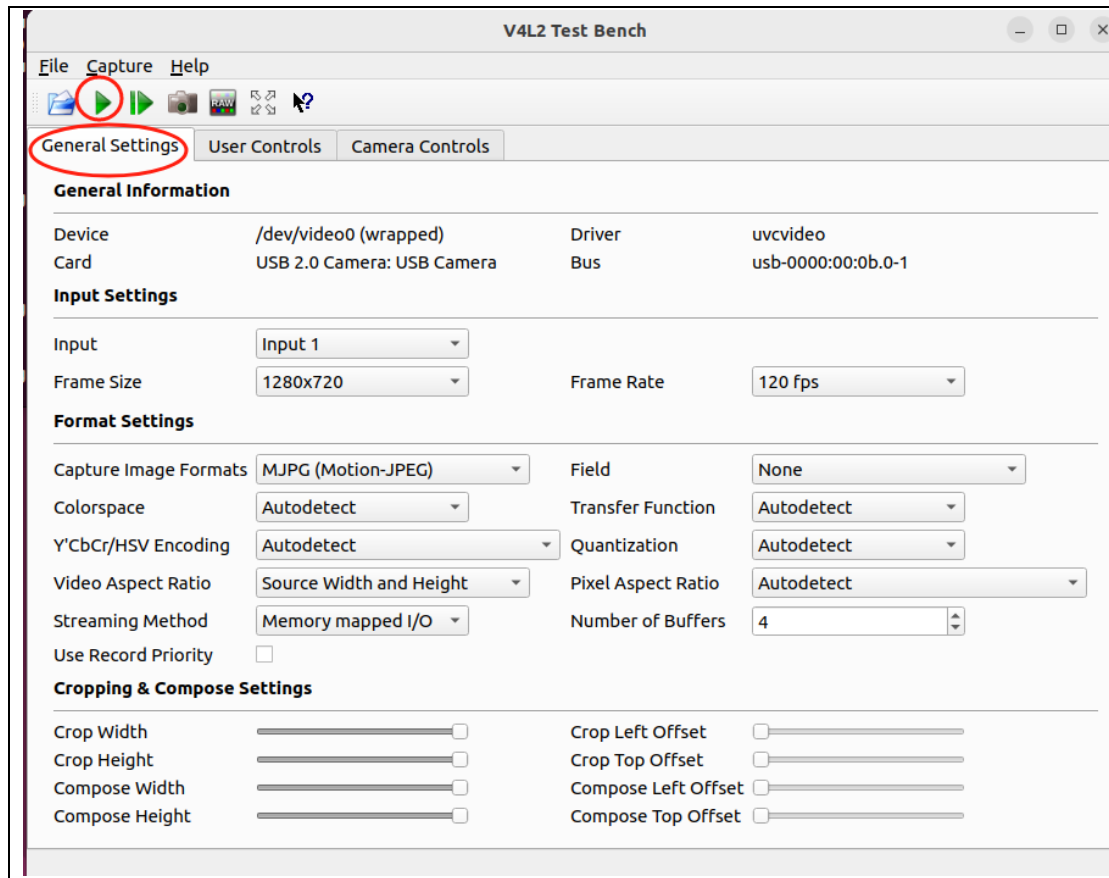
Website: www.inno-maker.com

Github: <https://github.com/INNO-MAKER>



U20CAM-IMX335

IMX335 HDR Camera Module
UVC 2.0 Series



User Controls

control parameters

Support: support@inno-maker.com

Sales : sales@inno-maker.com

16 / 28

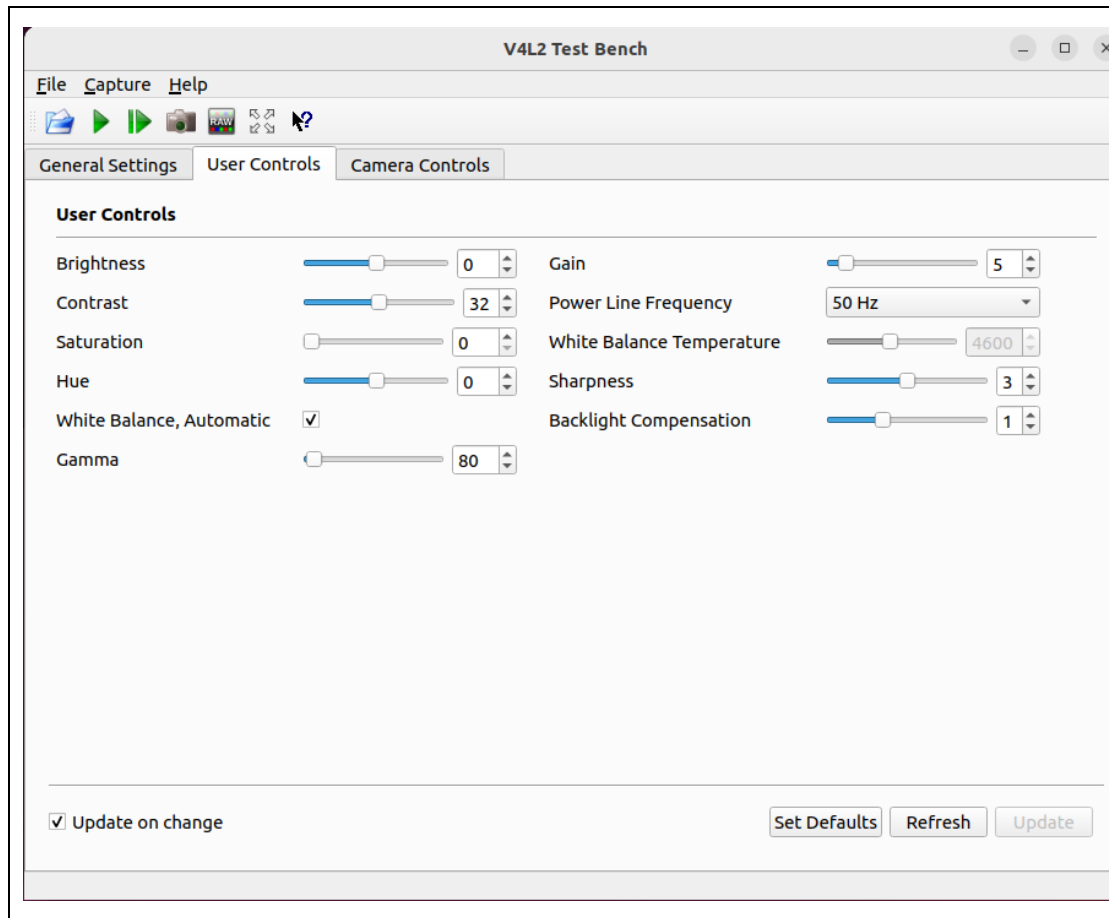
Website: www.inno-maker.com

Github: <https://github.com/INNO-MAKER>



U20CAM-IMX335

IMX335 HDR Camera Module
UVC 2.0 Series



Camera Controls

You can uncheck the External Trigger from this options.

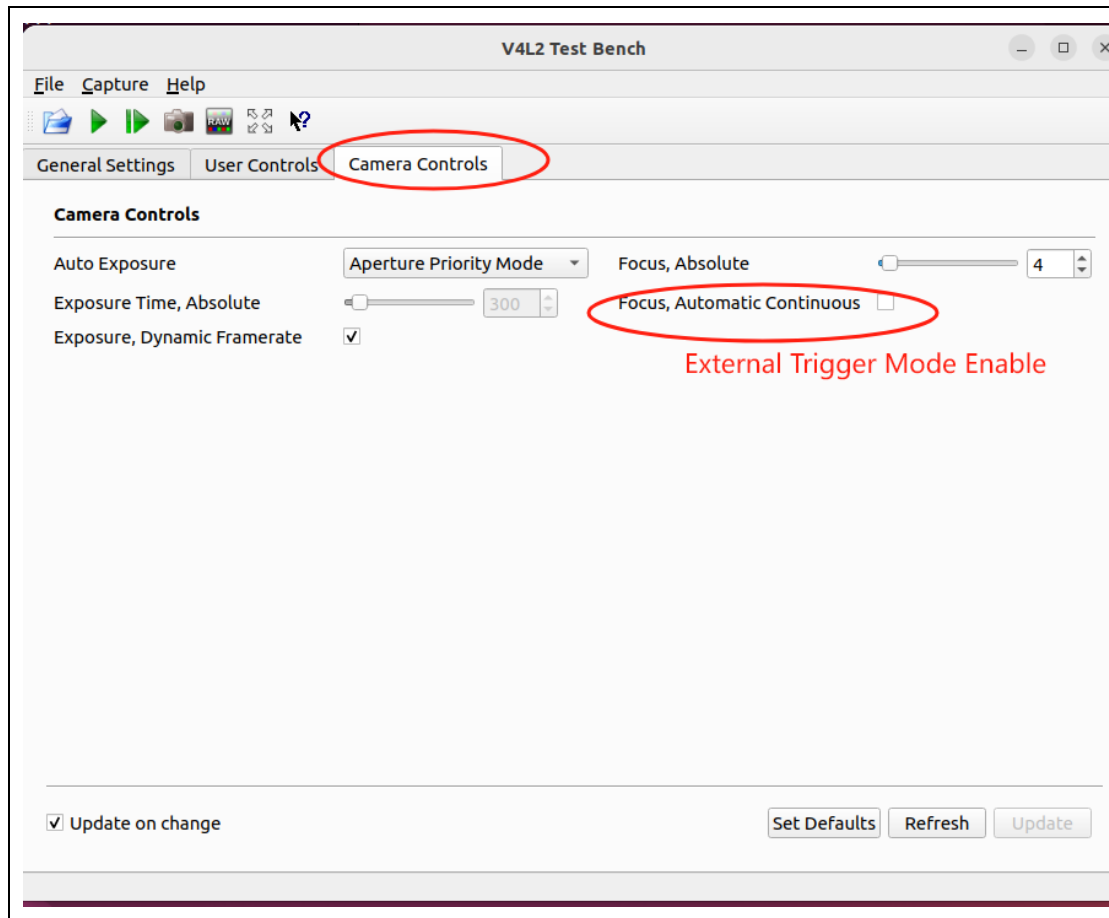
Support: support@inno-maker.com

Sales : sales@inno-maker.com

17 / 28

Website: www.inno-maker.com

Github: <https://github.com/INNO-MAKER>



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
```

Support: support@inno-maker.com

Sales : sales@inno-maker.com

Website: www.inno-maker.com

Github: <https://github.com/INNO-MAKER>

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
```

```
joez@joez-VirtualBox:~$ v4l2-ctl -d /dev/video0 -list
Video input set to 0 (Input 1: Camera, ok)

User Controls
    brightness 0x00980900 (int)      : min=-64 max=
    contrast 0x00980901 (int)       : min=0 max=
    saturation 0x00980902 (int)      : min=0 max=
    hue 0x00980903 (int)            : min=-180 m
    white_balance_automatic 0x0098090c (bool) : default=1
    gamma 0x00980910 (int)          : min=100 ma
    gain 0x00980913 (int)           : min=1 max=
    power_line_frequency 0x00980918 (menu)  : min=0 max=
    white_balance_temperature 0x0098091a (int) : min=2800 m
    sharpness 0x0098091b (int)       : min=0 max=
    backlight_compensation 0x0098091c (int)  : min=0 max=

Camera Controls
    auto_exposure 0x009a0901 (menu)  : min=0 max=
    exposure_time_absolute 0x009a0902 (int) : min=50 max=
    exposure_dynamic_framerate 0x009a0903 (bool) : default=0
```

Set User/Camera controls

For example, set camera brightness to 64

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

```
joez@joez-VirtualBox:~$ 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 encounter 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
```



U20CAM-IMX335

IMX335 HDR Camera Module
UVC 2.0 Series

```
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 source, 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")
```

Support: support@inno-maker.com

Sales : sales@inno-maker.com

Website: www.inno-maker.com

Github: <https://github.com/INNO-MAKER>



U20CAM-IMX335

IMX335 HDR Camera Module
UVC 2.0 Series

```
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)),
```

Support: support@inno-maker.com

Sales : sales@inno-maker.com

Website: www.inno-maker.com

Github: <https://github.com/INNO-MAKER>



U20CAM-IMX335

IMX335 HDR Camera Module
UVC 2.0 Series

```
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))
```

Support: support@inno-maker.com

Sales : sales@inno-maker.com

25 / 28

Website: www.inno-maker.com

Github: <https://github.com/INNO-MAKER>



```
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
```

```
joez@joez-VirtualBox:~/Desktop$ 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
```

```
joez@joez-VirtualBox:~/Desktop$ 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 ! \
    tcpserver sink host=0.0.0.0 port=5001

# client
```



U20CAM-IMX335

IMX335 HDR Camera Module
UVC 2.0 Series

```
# 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/>

Support: support@inno-maker.com

Sales : sales@inno-maker.com

Website: www.inno-maker.com

Github: <https://github.com/INNO-MAKER>

	U20CAM-IMX335 IMX335 HDR Camera Module UVC 2.0 Series
---	--

- (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>