RPi Camera (G)

来自Waveshare Wiki
跳转至： 导航、 搜索

硬件连接

要测试树莓派摄像头，需要给树莓派接入HDMI屏幕或者DSI屏幕

树莓派主板上的CSI （摄像头） 和 DSI （显示器） 两个接口的封装是相似的，接线的时候注意不要接错了。CSI接口在音频接口和HDMI接口中间，树莓派Zero系列的摄像头接口在Power接口边上。 如果你使用的计算模块，具体以底板的布局为准。

功能简介

特性	500万像素 鱼眼镜头
模组	OV5647
视场角	160
接口	CSI

- 接入树莓派Zero 系列

将排线的金属面朝下，接入摄像头接口

- 其他树莓派主板

将排线的金属面朝向HDMI接口，接入摄像头接口

关于型号

感光芯片型号	支持的树莓派主板型号	支持的驱动类型
OV5647	所有树莓派主板	libcamera / Raspicam
OV9281	所有树莓派主板	libcamera
IMX219 (树莓派官方)	所有树莓派主板	libcamera / Raspicam
IMX219 (第三方)	树莓派计算模块	libcamera
IMX290/ IMX327	所有树莓派主板	libcamera
IMX378	所有树莓派主板	libcamera
IMX477 (树莓派官方)	所有树莓派主板	libcamera / Raspicam
IMX477 (第三方)	树莓派计算模块	libcamera

开启摄像头接口

如果你是用的是最新的bullseye镜像，摄像头接口已经默认开启，可以跳过改步骤。

- 打开树莓派终端, 并使用指令进入设置界面

```
sudo raspi-config
```

- 设置开启摄像头

选择Interface Options -> Camera -> Yes -> Finish -> Yes

- 重启树莓派

测试摄像头(Bullseyes系统)

配置

如果使用的是OV9281, IMX290, IMX378, 或者非树莓派官方的IMX219和IMX477 摄像头需要另外配置config.txt 文件

```
sudo nano /boot/config.txt
```

找到camera-auto-detect=1 语句，修改为 camera_auto_detect=0

在文件结尾，根据摄像头型号加入以下设置语句

型号	设置语句
OV9281	dtoverlay=ov9281
IMX290/IMX327	dtoverlay=imx290,clock-frequency=37125000
IMX378	dtoverlay=imx378
IMX219	dtoverlay=imx219
IMX477	dtoverlay=imx477

打开树莓派终端，并开启摄像头预览：

```
sudo libcamera-hello -t 0
```

如果要关掉预览窗口，可以直接组合按键Alt-F4，或者点击x关掉。 也可以回到终端界面，用ctrl-c终止程序。

测试摄像头（Buster系统）

打开树莓派终端，并开启摄像头预览：

```
sudo raspistill -t 0
```

如果要关掉预览窗口，可以用ctrl-c终止程序。

前言

树莓派镜像在Bullseye版本之后，底层的树莓派驱动由Raspicam切换成libcamera。libcamera是一个开源的软件栈（后面会称呼做驱动，方便理解），方便于第三方移植和开发自己的摄像头驱动。截止到2021-12-20， libcamra还是存有很多bug，并且当前的libcamera不支持python， 因此树莓派官方还是提供了Raspicam的安装下载的方法。对于切换成libcamera困难，但是又需要用最新系统的用户，请直接移步到Raspicam的使用说明。

调用摄像头

libcamera软件栈提供了六个指令方便用户预览测试摄像头接口.

libcamera-hello

这个是一个简单的“hello world" 程序，用来预览摄像头并将摄像头画面显示在屏幕上。

使用示例

```
libcamera-hello
```

这个指令会在屏幕上预览摄像头大概5秒时间，用户可以用-t <duration>参数来设置预览的时间， 其中<duration>的单位是毫秒，如果设置为0的话就是保持一直预览。比如：

```
libcamerahello -t 0
```

调谐文件

树莓派的libcamera驱动会针对不同的摄像头模块调用一个调谐文件，调谐文件中提供了各种参数，器调用摄像头的时候，libcamera会调用调谐文件中的参数，结合算法对图像进行处理最终输出成预览画面。由于libcamera驱动只能自动感光芯片信号，但是摄像头的最终显示效果还会受整个模块的影响，调谐文件的使用就是为了可以灵活处理不同模块的摄像头，调整提高图像质量。

如果在使用默认调谐文件的情况下，摄像头的输出图像不理想的话，用户可以通过调用自定义的调谐文件来对图像进行调整。比如，如果你使用的是官方的NOIR版本摄像头，相对于于Raspberry Pi Camera V2 常规款，NOIR摄像头可能需要不同的白平衡参数，这种情况下就可以通过调用调谐文件来进行切换。

```
libcamera-hello --tuning-file /usr/share/libcamera/ipa/raspberrypi/imx219_noir.json
```

用户可以复制默认的调谐文件，根据自己的需求进行修改。

备注：调谐文件的使用适用于其他的libcamera指令，后续指令中就不再做介绍

预览窗口

大部分的libcamera指令都会显示一个预览窗口在屏幕上，用户可以通过--info-text 参数来自定义预览窗口的标题信息， 同时也可以通过%directives 调用一些摄像头参数并显示在窗口上比如如果使用HQ Camera： 可以通过--info-txe "%focus" 在窗口上显示摄像头的焦距

```
libcamera-hello --info-text "focus %focus"
```

备注：更多关于参数的设置，可以参考后续的指令参数设置章节

libcamera-jpeg

libcamera-jpeg 是一个简单的静态图片拍摄程序，不同于libcamera-still的复杂功能，libcamera-jpeg代码更简洁，并且有很多相同的功能来完成图片拍摄。

拍摄一张全像素的JPEG图像

```
libcamera-jpeg -o test.jpg
```

这个拍摄指令会显示一个5秒左右的预览串口，然后拍摄一张全像素的JPEG图像，保存为test.jpg

用户可以通过-t 参数来设置预览时间，同时可以通过--width 和 --height来设置拍摄图像的分辨率。例如：

```
libcamera-jpeg -o test.jpg -t 2000 --width 640 --height 480
```

曝光控制

所有的libcamera指令都允许用户自己设置快门时间和增益，比如：

```
libcamera-jpeg -o test.jpg -t 2000 --shutter 20000 --gain 1.5
```

这个指令会拍摄一张图像，拍摄中会曝光20ms并且摄像头增益被设置为1.5倍。设置的增益参数，会首先设置感光芯片内部的模拟增益参数，如果设置的增益超过了驱动内置的最大的模拟增益的数值，会先设置芯片的最大模拟增益，然后将剩下的增益倍数作为数字增益来生效。

备注：数字增益是通过ISP(图像信号处理）来实现的，并非是直接调整芯片内置寄存器。正常情况下，默认的数字增益趋近于1.0，除非是有有以下三种情况。

1. 整体的增益参数需求，也就是当模拟增益无法满足设定的增益参数需求的时候，会需要数字增益来做补偿
2. 其中一个颜色增益小于1 （颜色增益是通过数字增益来实现的），在这种情况下， 最后获得增益就是稳定为 1/min(red_gain, blue_gain), 也就是实际是上应用了统一的数字增益， 且是其中一个颜色通道的增益值（非绿色通道）。
3. AEC/AGC被修改了。如果AEC/AGC有变化的时候， 数值增益也会一定程度上会发生变化，从何来来消除任何波动，但是这种变化会被快速恢复到"正常"值

树莓派的AEC/AGX算法允许程序指定曝光补偿，也就是通过设置光圈数值来调整图像的亮度。

比如：

```
libcamera-jpeg --ev -0.5 -o darker.jpg
libcamera-jpeg --ev 0 -o normal.jpg
libcamera-jpeg --ev 0.5 -o brighter.jpg
```

libcamera-still

libcamera-still和libcamera-jpeg非常相似，不同的是libcamera继承了更多raspistill的功能。和之前一样，用户可以用以下指令拍摄一张图片.

测试指令

```
libcamera-still -o test.jpg
```

编码器

libcamea-still支持不同格式的图像文件，可以支持png和bmp编码，也支持直接不带编码或者任何图像格式地将RGB或者YUV像素的二进制转储保存成文件。如果是直接保存RGB或者YUV数据，程序在读取此类文件的时候必须了解文件的像素排列方式。

```
libcamera-still -e png -o test.png
libcamera-still -e bmp -o test.bmp
libcamera-still -e rgb -o test.data
libcamera-still -e yuv420 -o test.data
```

备注：图像保存的格式是通过-e参数控制的， 如果没有调用-e参数设置的话，默认按照输出的文件名的格式保存。

原始图像拍摄

原始图像（Raw image)图像就是直接图像传感器输出的图像， 没有经过任何ISP或者CPU处理。对于彩色相机传感器，一般来说原始图像的输出格式是Bayer. 注意原始图和我们之前说的位编码的RGB和YUV图像不同，RGB和YUV也是经过ISP处理后的图像的。
拍摄一张原始图像的指令：

```
libcamera-still -r -o test.jpg
```

原始图像一般是以DNG （Adobe digital Negative) 格式保存的，DNG格式可以兼容大部分标准程序， 比如dcraw 或者RawTherapee. 原始图像会被保存为.dng后缀的相同名字的文件,比如如果运行上面的指令，为被另存为test.dng， 并同时生成一张jpeg文件。DNG文件中包含了已图像获取有关的元数据， 比如白平衡数据，ISP颜色矩阵等， 下面是用exiftool工具显示的元数据编码信息：

```
File Name                       : test.dng
Directory                       : .
File Size                       : 24 MB
File Modification Date/Time     : 2021:08:17 16:36:18+01:00
File Access Date/Time           : 2021:08:17 16:36:18+01:00
File Inode Change Date/Time     : 2021:08:17 16:36:18+01:00
File Permissions                : rw-r--r--
File Type                       : DNG
File Type Extension             : dng
MIME Type                       : image/x-adobe-dng
Exif Byte Order                 : Little-endian (Intel, II)
Make                            : Raspberry Pi
Camera Model Name               : /base/soc/i2c0mux/i2c@1/imx477@1a
Orientation                     : Horizontal (normal)
Software                        : libcamera-still
Subfile Type                    : Full-resolution Image
Image Width                     : 4056
Image Height                    : 3040
Bits Per Sample                 : 16
Compression                     : Uncompressed
Photometric Interpretation      : Color Filter Array
Samples Per Pixel               : 1
Planar Configuration            : Chunky
CFA Repeat Pattern Dim          : 2 2
CFA Pattern 2                   : 2 1 1 0
Black Level Repeat Dim          : 2 2
Black Level                     : 256 256 256 256
White Level                     : 4095
DNG Version                     : 1.1.0.0
DNG Backward Version            : 1.0.0.0
Unique Camera Model             : /base/soc/i2c0mux/i2c@1/imx477@1a
Color Matrix 1                  : 0.8545269369 -0.2382823821 -0.09044229197 -0.1890
484985 1.063961506 0.1062747385 -0.01334283455 0.1440163847 0.2593136724
As Shot Neutral                 : 0.4754476844 1 0.413686484
Calibration Illuminant 1        : D65
Strip Offsets                   : 0
Strip Byte Counts               : 0
Exposure Time                   : 1/20
ISO                             : 400
CFA Pattern                     : [Blue,Green][Green,Red]
Image Size                      : 4056x3040
Megapixels                      : 12.3
Shutter Speed                   : 1/20
```

超长曝光

如果要拍摄一张超长曝光的图片，我们需要禁用AEC/AGC和白平衡，否则这些算法会导致图片在收敛的时候多等待很多帧数据。禁用这些算法需要另设置显式值，另外, 用户可以通过--immediate设置来跳过预览过程。

这里是拍摄一张曝光100秒的图像指令：

```
libcamera-still -o long_exposure.jpg --shutter 100000000 --gain 1 --awbgains 1,1 --immediate
```

备注：几款官方摄像头的最长曝光时间参考表格.

模块	最长曝光时间(s)
V1(OV5647)	6
V2(IMX219)	10
HQ(IMX477)	230

libcamera-vid

libcamera-vid是一个视频录制程序, 默认使用的是树莓派的硬件H.264编码器。这个程序运行之后会在屏幕上显示一个预览窗口，同时将比特流编码输出到指定文件。比如，录制一个10s的视频。

```
libcamera-vid -t 10000 -o test.h264
```

如果要查看视频可以用vlc来进行播放

```
vlc test.h264
```

备注：录制的是未打包的视频流， 用户可以使用--save-pts 设置输出时间戳，方便后续将比特流转成其他视频格式。

```
libcamera-vid -o test.h264 --save-pts timestamps.txt
```

如果想要输出mkv文件，可以用以下指令：

```
mkvmerge -o test.mkv --timecodes 0:timestamps.txt test.h264
```

编码器

树莓派支持JPEG格式以及没有压缩和格式的YUV420：

```
libcamera-vid -t 10000 --codec mjpeg -o test.mjpeg
libcamera-vid -t 10000 --codec yuv420 -o test.data
```

--codec选项设置的是输出格式，而不是输出文件的扩展名。
使用--segment参数可以将输出文件分割成段（单位是ms)，适用于需要脚JPEG视频流分割成单独的时间比较短（大概1ms)的JPEG文件

```
libcamera-vid -t 10000 --codec mjpeg --segment 1 -o test%05d.jpeg
```

libcamera-raw

libcamera-raw类似于视频录制程序，不同的地方时，libcamera-raw录制的是直接传感器输出的Bayer格式的数据，也就是原始图像数据。libcamera-raw不会显示预览窗口。比如录制一个2秒的原始数据片段.

```
libcamera-raw -t 2000 -o test.raw
```

程序会在没有格式信息的情况下直接转储原始帧， 程序会将像素格式和图像尺寸直接打印在终端，用户可以根据输出的数据查看像素数据.
默认情况下，程序会将原始帧保存成一个文件，文件通常比较大，用户可以通过--segement参数将文件进行分割。

```
libcamera-raw -t 2000 --segment 1 -o test%05d.raw
```

如果内存条件比较好（比如使用SSD), libcamera-raw可以以大概10帧每秒的速度将官方的HQ Camera的数据（大概18MB每帧）写入到硬盘中， 为了达到这个速度，程序写入的是没有格式化过的原始帧，没有办法像libcamera-still那样保存成DNG文件。 如果想要确保不出现丢帧的情况，可以使用--framerate 降低帧率。

```
libcamera-raw -t 5000 --width 4056 --height 3040 -o test.raw --framerate 8
```

通用的指令设置选项

通用的指令设置选项适用于libcamera的所有指令

```
--help,         -h
```

打印程序帮助信息，可以打印每个程序指令的可用设置选项，然后退出.

```
--version
```

打印软件版本，打印libcamera和libcamera-app的软件版本，然后退出.

```
--timeout,      -t
```

-t选项 设置了libcamera程序运行时间， 如果运行的是视频录制指令，timeout选项设置的是录制时长，如果运行的是图像拍摄指令，timeout设置的拍摄并输出图像之前的预览时间。
如果在运行libcamera程序的时候没有设置timeout，默认的timeout数值就是5000（5秒），如果将timeout设置为0，那程序就会一直运行。
示例： *libcamera-hello -t 0*

```
--preview,  -p
```

-p 设置预览预览窗口大小以及窗口的位置（则合格设置在 X和DRM版本的窗口中都有效），设置格式为 --preview <x. y, w, h> 其中x y设置预览窗口在显示屏上的坐标位置， w和h设置的是预览窗口的宽度和长度
预览串口的设置不会影响摄像头图像预览的分辨率和宽高比。 程序会将预览图像缩放到预览窗口中显示，并会按照原来的图像宽高比做适配。
示例: *libcamera-hello -p 100,100,500,500*

```
--fullscreen, -f
```

-f选项设置预览窗口全屏显示，全屏显示模式的预览窗口和边框。同-p一样，不会影响分辨率和宽高比，会自动适配。
示例： *libcamera-still -f -o test.jpg*

```
--qt-preview
```

使用基于QT框架的预览窗口， 正常情况下不推荐用这个设置，因为这个预览程序不会使用零拷贝缓冲区共享以及GPU加速， 这个会导致占用资源过高。 QT预览窗口支持X转发（默认预览程序不支持）.
Qt预览串口不支持--fullscreen设置选项， 如果用户要使用Qt预览，建议保持小预览窗口，避免资源占用过高影响系统正常运行。
示例： *libcamera-hello --qt-preview*

```
--nopreview, -n
```

不预览图像。这个设置会关掉图像预览功能.

示例：*libcamera-hello -n*

```
--info-text
```

设置预览窗口的标题和信息显示（只在X图形窗口下有效）使用格式为 --info-text <string>。调用改选项，有多个参数可以设置，参数通常以%指令格式调用。 程序会按照指令调用图形元数据中的对应数值。

如果没有指定窗口信息，默认的--info-text设置为 *"#%frame (%fps fps) exp %exp ag %ag dg %dg"*

示例： *libcamera-hello --info-test "Focus measure: %focus* 可用参数:

指令	说明
%frame	帧序列号
%fps	瞬时帧速率
%exp	捕捉图像时的快门速度， 单位是ms
%ag	感光芯片控制的图像模拟增益
%dg	通过ISP控制的图像数值增益
%rg	每个像素点红色组件的增益
%bg	每个像素点蓝色组件的增益
%focus	图像的角点度量，数值越大表示图像越清晰

```
--width
--height
```

这两个参数分别设置图像的宽度和高度。对于libcamera-still， libcamera-jpeg和libcamera-vid指令，这两个参数可以设置输出图像/视频的分辨率。

如果使用libcamera-raw指令，这两个参数会影响获取的元数据帧的大小。 摄像头有一个2x2的分块读取模式，如果设置的分辨率小于分开模式， 摄像头会按照2x2的分块大小获取元数据帧。

libcamera-hello无法指定分辨率.< br /> 示例：

libcamera-vid -o test.h264 --width 1920 --height 1080 录制1080p视频

libcamera-still -r -o test.jpg --width 2028 --height 1520 拍摄一张分辨率为2028x1520的JPEG图像。

```
--viewfinder-width
--viewfinder-height
```

这个设置选项也是用来设置图像的分辨率，不同的是只设置的预览的图像大小。并不会影响最终输出的图像或者视频的分辨率。 预览图像大小的设备不会影响预览窗口尺寸，会根据窗口适配。

示例: *libcamera-hello --viewfinder-width 640 --viewfinder-height 480*

```
--rawfull
```

这个设置强制感光芯片活了--width和--height的设置， 在全分辨率读取模式下输出静态图像和视频。这个设置libcamera-hello无效。

使用该设置，会牺牲帧率。全分辨率模式下，帧读取速度会比较慢。

示例： *libcamera-raw -t 2000 --segment 1 --rawfull -o test%03d.raw* 示例指令会捕获多张全分辨率模式下的元数据帧。 如果你使用的是HQ摄像头。 每个帧的大小为18MB， 而如果没有设置--rawfull， HQ摄像头默认的是2x2模式， 每帧的数据大小只有4.5MB.

```
--lores-width
--lores-height
```

这个两个选项设置低分辨率图像。 低分辨率数据流会压缩图像，导致图像的纵横比改变。在使用libcamera-vid录制视频的时候，如果设置了低分辨率，会禁用掉颜色去噪处理等功能。

示例: *libcamera-hello --lores-width 224 --lores-height 224* 注意，低分辨率设置通常要结合图像后处理使用，否则效用不大。

```
--hflip    #水平翻转图像
--vflip     #垂直翻转图像
--rotation      #根据给出的角度，水平或者垂直翻转图像 <angle>
```

这三个选项用来翻转图像。 --rotation的参数目前只支持0 和180， 其实就是相当于--hflip和--vflip.

示例： *libcamera-hello --vflip --hflip*

```
--roi    #裁剪图像<x, y, w, h>
```

--roi允许用户从传感器提供的完整图像中根据坐标裁剪自己想要的图像区域，也就是数字缩放，注意坐标值要是在有效范围的。 比如 --roi 0, 0, 1, 1就是无效的指令。

示例： *libcamera-hello --roi 0.25,0.25,0.5,0.5* 示例指令会从图像中心裁剪1/4图像出来。

```
--sharpness #设置图像的锐度 <number>
```

通过<number>数值调整图像的锐度。 如果设置为0，就是不应用锐化。 如果设置的值超过1.0，会使用而外的锐化量。

示例: *libcamera-still -o test.jpg --sharpness 2.0*

```
--contrast  #设置图像对比度 <number>
```

示例： *libcamera-still -o test.jpg --contrast 1.5*

```
--brightness  #设置图像亮度<number>
```

设置范围是-1.0 ~ 1.0

示例： *libcamera-still -o test.jpg --brightness 0.2*

```
--saturation  #设置图像色彩饱和度 <number>
```

示例：*libcamera-still -o test.jpg --saturation 0.8*

```
--ev  #设置EV补偿<number>
```

以光圈为单位来设置图像的EV补偿，设置范围是-10 ~ 10， 默认值是0. 程序通过提高获奖度AEC/AGC算法的目标方式来作用。

示例：*libcamera-still -o test.jpg --ev 0.3*

```
--shutter  #设置曝光时间 ，单位是ms <number>
```

注意：如果相机的帧率太快，可能会导致无法按照设定的快门时间工作，如果遇到这种情况，可以尝试用--framerate来降低帧速率。

示例：*libcamera-hello --shutter 30000*

```
--gain  #设置增益值 （数值增益和模拟增益组合） <number>
--analoggain  #--gain的代名词
```

--analoggain和--gain是一样的，使用analoggain只是为了兼容raspicam的程序。

```
--metering  #设置测光模式 <string>
```

设置AEC/AGC算法的测光模式， 可用的参数有：

- centre - 中心测光 （默认）

- spot -点测光
- averag -平均或者全幅测光
- custom -自定义测光模式， 可以通过调谐文件设置

示例：*libcamera-still -o test.jpg --metering spot*

```
--exposure   #设置曝光配置文件 <string>
```

曝光模式可以设置为normal或者sport. 这两种模式的报告配置文件不会影响图像的整体曝光，但是如果是sport模式的话，程序会缩短曝光时间和提高正义来达到同样的曝光效果。

示例：*libcamera-still -o test.jpg --exposure sport*

```
--awb  #设置白平衡模式<string>
```

可用的白平衡模式：

模式	**色温**
auto	2500K ~ 8000K
incadescent	2500K ~ 3000K
tungsten	3000K ~3500K
fluorescent	4000K ~ 4700K
indoor	3000K ~ 5000K
daylight	5500K ~ 6500 K
cloudy	7000K ~ 8500K
custom	自定义范围， 通过调谐文件设置

示例： *libamera-still -o test.jpg --awb tungsten*

```
--awbgains   #设置固定的颜色增益<number,number>
```

设置红色和蓝色增益。

示例： *libcamera-still -o test.jpg --awbgains 1.5, 2.0*

```
--denoise  #设置去噪模式 <string>
```

支持的去噪模式：

- auto -默认模式, 使用标准空间去噪， 如果是视频，会使用快速色彩降噪，拍摄静态图片的时候会使用高质量的色彩降噪。预览图像不会使用任何色彩去噪
- off - 关闭空间去噪和色彩去噪
- cdn_off -关闭色彩去噪
- cdn_fast - 使用快速色彩去噪

- cdn_hq - 使用高质量色彩去噪，不适用于视频录制。

示例： *libcamera-vid -o test.h264 --denoise cdn_off*

```
--tuning-file #指定摄像头调谐文件 <string>
```

关于更多调谐文件的说明， 可以参考官方教程

示例: *libcamera-hello --tuning-file ~/my~camera-tuning.json*

```
--output, -o  #输出文件名 <string>
```

设置输出图像或者视频的文件名。 除了设置文件名之外，还可以指定输出的udp或者tcp服务器地址，从而将图像输出到服务器上。 有兴趣的可以查看后续tcp和udp的相关设置说明。

示例: *libcamera-vid -t 100000 -o test.h264*

```
--wrap  #将输出文件计数器打包<number>
```

示例： *libcamera-vid -t 0 --codec mjpeg --segment 1 --wrap 100 -o image%d.jpg*

```
--flush  #马上刷新输出文件
```

--flush会将每一帧图像写入的同时都马上更新到硬盘中，降低延迟。

示例：*libcamera-vid -t 10000 --flush -o test.h264*

静态图片拍摄设置参数

```
--qiality, -q  #设置JPEG图像质量 <0 ~ 100>
--exif, -x #添加而外的EXIF标志
--timelapse  #延时拍照的时间间隔， 单位是ms
--framestart #帧计数的开始数值
--datetime  #用日期格式命名输出文件
--timestamp #用系统时间戳命名输出文件
-- restart  #设置JPEG重启时间间隔
--keypress, -k  # 设置回车按键拍照模式
--signal, -s  #设置信号触发拍照
--thumb #设置缩略图参数 <w:h:q>
--ebcoding, -e  #设置图像编码类型。 jpg / png / bmp / rgb / yuv420
--raw, -r  #保存原始图像
--latest #关联符号到最新保存的文件
```

视频录制图像设置参数

```
--quality, -q  # 设置JPEG指令 <0 - 100>
--bitrate, -b  # 设置H.264比特率
--intra, -g #设置内部帧周期 （只支持H.264）
--profile #设置H.264配置
--level  #设置H.264等级
--codec  #设置编码类型 h264 / mjpeg / yuv420
--keypress, -k  #设置回车暂停和录制
--signal,  -s #设置信号暂停和录制
--initial #在录制或者暂停状态下启动程序
--split #切割视频并保存到另外的文件
--segment #将视频分割成多个视频段
--circular  #将视频写入循环缓冲区中
--inline #在每个I帧中写入数据头（只支持H.264）
--listen #等待TCP连接
```

- 更多摄像头设置说明，请参考官方文档官方摄像头文档

前言

如果你使用的是Buster版本的镜像，系统默认安装的是Raspicam驱动，可以直接开启摄像头，然后使用。

如果使用的是最新的Bullseye系统，需要另外安装配置一下。

安装Raspicam（可选）

打开树莓派终端，并输入以下指令安装驱动. 注意：此驱动暂时无法支持64位的树莓派系统。

```
sudo apt-get update
cd
sudo apt install cmake
git clone https://github.com/raspberrypi/userland
cd userland
./buildme
cp build/bin/* ~/bin/
```

按照后之后需要重启系统

```
sudo reboot
```

调用摄像头

raspistill

raspistill指令用于拍摄静态图片。 示例：*raspistill -o cam.jpg*

raspivid

raspivid指令用于录制视频。 示例: *raspivid -o vid.h264*

raspiyuv

raspiyuv指令具有跟raspisitll相同的功能，不同的是raspiyuv不是苏楚。jpgs等标准图像文件，而是从相机的ISP输出中生成YUV420或者RGB888图像文件。
大部分情况下，拍摄图像，用raspistill比较好，但是如果你想用未压缩的黑白图像的时候，可以选择使用raspiyuv指令。 示例：*raspiyuv -o cam.jpg*

指令设置选项

raspicam 指令在使用的时候，可以通过设置选项来调整图像最终成像效果。以下列出可用的设置选项以供参考.

```
--preview, -p  #设置预览窗口<x, y, w, h>
```

用户可以通过设置x， y的值来设置预览窗口的位置， 设置w, h的值来调整预览图像的分辨率。

```
--fullscreen, -f  #全屏预览窗口
```

将预览窗口全屏显示

```
--nopreview, -n  #不显示预览窗口
```

关掉预览窗口，指令会直接输出图像或者视频到文件。

```
--opacity, -op  #设置预览窗口透明度
```

设置预览窗口的透明度， 0 是不可见， 255 是全透明

```
--sharpness, -sh  #设置图像锐度 (-100 - 100)
```

默认锐化值为0

```
--contrast. -co  #设置图像对比度 （-100 - 100）
```

默认的对比度为0

```
--brightness, -br  #设置图像对比度 （0 - 100）
```

默认亮度为50， 0是全黑，100是全白。

```
--saturation, -sa  #设置图像饱和度 (-100 - 100)
```

默认饱和度是0

```
--ISO  #设置快门感光度 (100 - 800)
```

设置拍照时的感光度

```
--vstab, -vs  #开启视频稳定
```

只有视频录制模式有效，开启视频防抖。

```
--ev  #设置EV补偿
```

设置图像的EV补偿值， 默认0

```
--exposure,  -ex  #设置曝光模式
```

可设置的曝光选项：

- auto: 自动曝光模式
- night: 夜间拍摄模式
- nightpreview
- backlight: 背光模式
- spotlight
- sports: 运动模式(运动模式下会缩短快门时间
- snow: 雪景模式
- beach: 海滩模式
- verylong: 长曝光模式
- fixedfps: 固定fps
- antishake: 防抖模式
- fireworks: 烟花模式

注：根据摄像头型号，有部分模式不可用

```
--flicker, -fli   #闪烁避免
```

可设置的模式：

- off: 关闭防闪烁模式
- auto: 自动检测注频率
- 50Hz: 设置防闪烁频率是50Hz
- 60Hz: 设置防闪烁频率是60Hz

```
--awb,  #设置自动白平衡模式
```

- off: 关闭自动白平衡
- auto： 自动白平衡模式（默认）
- sun: 晴天模式 （5000K ~ 6500K）
- cloud: 多云模式 (6500K ~ 12000K)
- shade：阴影模式
- tungsten: 钨丝灯模式（2500K ~ 3500K）
- incandescent： 白炽灯模式(2500K ~ 4500K)
- flash: 闪光灯模式
- horizon: 地平线模式
- greyworld: 如果使用的是不带红外滤光片的相机（比如NoIR），可以修复因为缺少IR滤镜而导致的白平衡失调的情况

注：根据摄像头型号，有部分模式不可用

```
--imfx, -ifx  #设置图像滤镜效果
```

可设置的图像滤镜效果：

- none: 无效果（默认）
- negative：颜色翻转
- solarise: 日照效果
- posterise: 海报效果
- whiteboard: 白板效果
- backboard：黑板效果
- sketch: 素描
- denoise: 去噪

- emboss: 浮雕
- oilpaint: 油画
- hatch: 纹理
- gpen: 铅笔素描
- pastel: 彩色铅笔
- watercolour: 水彩
- film: 胶片
- blur: 模糊
- saturation: 饱和

注：根据摄像头型号，有部分模式不可用

```
--colfx, -cfx  #设置颜色效果<U:V>
```

U和V参数的设置范围是 0 ~ 255， 用来调整U和Y通道的数值。 比如： --colfx 128:128 会讲图像设置成单色图。

```
--metering, -mm  #设置测光模式
```

可设置选项：

- average: 平均或者全幅测光
- spot:点测光
- backlit: 预设一个背光图像
- matrix: 矩阵测光

```
--rotation, -rot  #设置图像旋转 (0 - 359)
```

可以通过角度参数设置图像旋转角度

```
--hflip, -hf  #设置图像水平翻转
```

设置图像水平翻转

```
--vflip, -vf  #设置图像垂直翻转
```

垂直翻转图像

```
--roi, #裁剪图像 <x, y, w, h>
```

根据参数裁剪图像，注意参数都是规划化在（0,0 ~ 1.0）, 比如，如果要裁剪1/4图像，可以用指令 *-roi 0.5,0.5,0.25,0.25*

```
--shutter, --ss  设置快门速度/时间
```

设置快门时间（单位:ms）。 快门时间根据感光芯片的不同，可设置的最大快门时间不同。

型号	最大数字（ms）
V1 (OV5647)	6000000 (6s)
V2 (IMX219)	10000000 (10s)
HQ (IMX477	200000000 (200s)

```
--drc, -drc  #启用/关闭动态范围压缩
```

- off (默认)
- low
- med
- high

```
--stats, -st  #通过静态图像帧来统计图像
```

```
--awbgains, -awbg
```

设置蓝色和绿色增益， 在设置 --awb off 的情况下生效

```
--analoggain, -ag
```

设置模拟增益值

```
--digitalgain, -dg
```

设置数字增益值

```
--mode, -md
```

设置传感器模式：

- OV5647

模式	分辨率	纵横比	帧率	FoV
0	自动选择			

1	1920 x 1080	16:9	1-30fps	局部
2	2592 x 1944	4:3	1-15fps	全幅
3	2592 x 1944	4:3	0.1666 - 1fps	全幅
4	1296 x 972	4:3	1 - 42fps	全幅
5	1296 x 730	16:9	1- 49fps	全幅
6	640 x 480	4:3	42.1 - 60fps	全幅
7	640 x 480	4:3	60.1 - 90fps	全幅

- IMX219

模式	**分辨率**	**纵横比**	**帧率**	**FoV**
0	自动选择			
1	1920 x 1080	16:9	0.1-30fps	局部
2	3280 x 2464	4:3	0. 1-15fps	全幅
3	3280 x 2464	4:3	0.1 - 15fps	全幅
4	1640 x 972	4:3	1 - 42fps	全幅
5	1296 x 730	16:9	1- 49fps	全幅
6	640 x 480	4:3	42.1 - 60fps	全幅
7	640 x 480	4:3	60.1 - 90fps	全幅

- HQ Camera

模式	**分辨率**	**纵横比**	**帧率**	**FoV**
0	自动选择			
1	2028 x 1080	169:90	0.1-50fps	局部
2	2028 x 1080	4:3	0. 1-50fps	全幅
3	4056x3040	4:3	0.005 - 10fps	全幅
4	1332x990	74:55	50.1-120 fps	局部

```
--camselect, -cs
```

当系统接入多个摄像头时，选择摄像头， 0 或 1.

```
--annotate, -a #启用/设置注释
```

在元数据中使用位掩码的方式来表示参数，可以直接用加法表示。 比如， 12 可以表示显示时间（4）和显示日期 （8）, 就是通过加法 4+8 =12表示的。

可设置的选项

值	**参数说明**
-a 4	时间
-a 8	日期
-a 12	时间和日期，实际就是 4 + 8

-a 16	快门设置
-a 32	CAF设置
-a 64	增益设置
-a 128	镜头设置
-a 256	动画设置
-a 512	帧数
-a 1024	黑色背景
-a "ABC %Y-%m-%d %X"	显示文本
-a 4 -a "ABC %Y-%m-%d %X"	显示自定义格式的时间/日期
-a 8 -a "ABC %Y-%m-%d %X"	显示自定义格式的时间/日期

```
--annotateex, -ae  #设置额外注解参数
```

```
--stereo, -3d
```

设置双目模式。

- sbs - 并排模式
- tb: -竖排模式
- off -关掉双目模式（默认）

```
--decimate, -dec
```

将双目图像的宽度和高度减半

```
--setting, -set
```

输出当前的摄像头设置

raspistill 设置参数

```
--width, -w  #设置图像宽度
--height, -h  #设置图像高度
--quality, 0q  #设置JPEG质量 <0 ~ 100>
--raw, -r  #将原Bayer数据添加到JPEG元数据中
--output, -o #输出文件 <filename>
--latest, -l  #关联最后一帧图像到文件 <filename>
--verbose, -v #打印详情
--timeout, -t  #设置程序预览时间
--timelapse, -tl  #设置演示摄影
--framestart, -fs #保存第一帧的编号
--datetime, -dt  #用日期时间命名文件
--timestamp, -ts #用时间戳命名文件
--thumb, -th #设置缩略图参数 <x:y:qulity>， 默认为（64:48:35）
--demo, -d  #运行演示模式<ms》
--encoding, -e #按照指定格式编码， jpg，bmp或者png
--restart, -rs  #设置JPEG重启标志
--exif, -x   #设置EXIF标记
--gpsdexif, -gps #设置实时exif 时间 （需要有GPS Dongle接入）
--fullpreview,  -fg  #全屏预览
--keypress, -k  #按键拍照模式
--signal, -s  # 信号模式
--burst, -bm #抓拍模式
```

raspivid 设置参数

```
--width, -w #设置视频宽度
--height, -h #设置视频高度
--bitrate, -b #设置比特率
--output, -o #设置输出文件名 <filename>
--listen, -l #使用网络连接的时候，等待网络连接
--versbose, -v #打印详情
--timeout, -t  #设置预览时间
--demo,  -d #运行演示 <ms>
--framerate, -fs #设置帧率
--penc, -e #显示编码后的预览图像
--intra, -g #设置内部刷新周期
--qp, -qp #设置量化参数
--profile, -pf #指定H264配置文件， baseline / main / high
--level, -lev  #设置H264编码等级
--irefresh, -if #设置 H264内部刷新类型 cyclic / adaptive / both / cyclicrows
--inline, -ih  #插入PPS，SPS头
--spstimings, -stm #将时序信息插入到SPS块中
--timed,  -td  #定时切换录制和暂停功能
--keypress, -k  #按键暂停录制
--signal,  -s  #根据SIGUSR1 切换暂停和录制状态
--split, -sp  #在signal和keypress 模式下，每次重新录制的时候都会重新创建文件
--vectors, -x  #矢量输出
--flush, -fl  #写入视频数据后立即强制刷新输出数据缓冲区， 绕过了写入数据的任何操作系统缓存，并且可以减少延迟。
--save-pts, -pts  #将时间戳信息保存到指定文件。
--codec, -cd  # 指定编码器 H264 / MJPEG
--initial, -i  #设置启动是的初始状态
--segment, -sg #将视频流分割到多个文件中
--wrap, -wr  #设置分割的最大值
--start, -sn #设置初始分割编码
--raw, -r #设置元数据文件名
--raw-format, -rf #指定元数据格式 yuv / rgb / grey
```

资料

文档

- 用户手册

视频

- 演示视频

尺寸图

- 尺寸图

相关链接

树莓派入门教程（新） 展开

树莓派入门教程 展开

树莓派OpenCV教程 展开

树莓派littleGL系列教程 展开

树莓派QT教程 展开

树莓派OpenWrt教程 展开

认证资料

- CE RoHS

FAQ

问题：如果拍摄的图片，白平衡效果不正常?

- 在不同的拍摄环境下，可能会出现白平衡效果不正常的情况。用户可以自己根据实际使用情况调节白平衡参数。
- 以下opencv的调节代码只作参考。（该代码由RPi Camera （G）使用用户分享）

```
import picamera
import picamera.array
import cv2
from time import sleep
import numpy as np

def test_gcamera():
    cv2.namedWindow("img",0)
    with picamera.PiCamera() as camera:
        camera.resolution = (1920, 1080)
        camera.awb_mode = 'off'
        rg, bg = (1.8, 1.4)
        camera.awb_gains = (rg, bg)
        with picamera.array.PiRGBArray(camera) as output:
            for foo in camera.capture_continuous(output, 'rgb', use_video_port=Tr
ue):
                img = cv2.cvtColor(output.array, cv2.COLOR_RGB2BGR)
                cv2.imshow("img", img)
                cv2.waitKey(1)
                cv2.imwrite("test.jpg",img)
                r, g, b = (np.mean(output.array[..., i]) for i in range(3))
                if abs(r - g) > 2:
                    if r > g:
                        rg -= 0.1
                    else:
                        rg += 0.1
                if abs(b - g) > 1:
                    if b > g:
                        bg -= 0.1
                    else:
                        bg += 0.1
                camera.awb_gains = (rg, bg)
                output.seek(0)
                output.truncate(0)
if __name__=="__main__":
    test_gcamera()
```