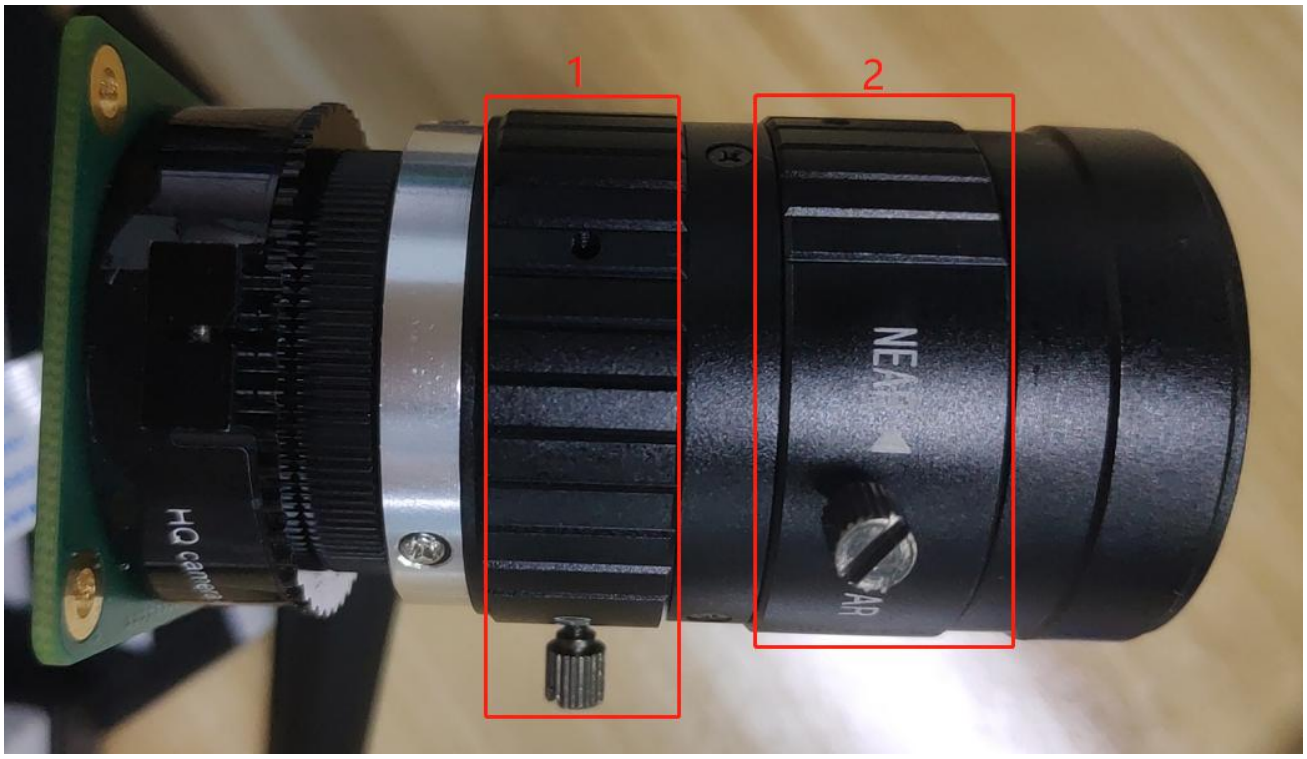
Take video

1.Adjust camera focus

1)The HQ camera possess two adjustable parts, as shown below, inner ring 1 and outer ring 2.
Inner ring 1: Mainly adjust the brightness collected by the camera.
Outer ring 2: Mainly used to adjust the focal length of the camera.

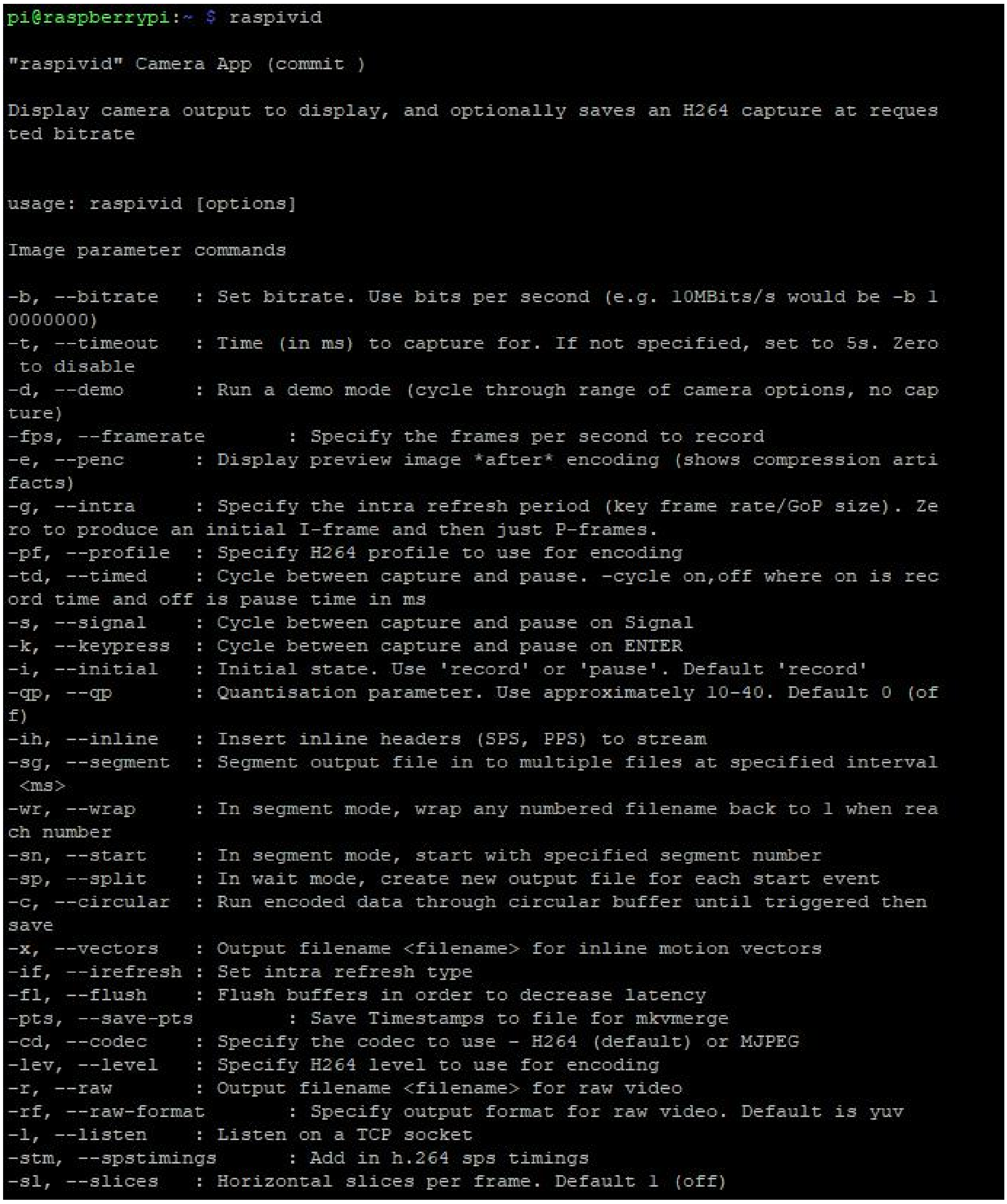


2)After adjust the focal length of the camera is complete, then gently screw on the above screws to fix the focal length.

Note： Video recording is limited to a maximum 1080p(1920×1080) resolution.

2.Take video

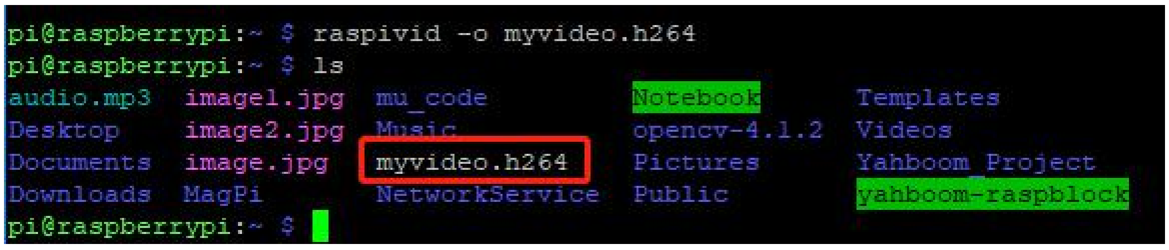
1) raspivid -------This command is to get the video information



```
pi@raspberrypi:~ $ raspivid

"raspivid" Camera App (commit )

Display camera output to display, and optionally saves an H264 capture at reques
ted bitrate


usage: raspivid [options]

Image parameter commands

-b, --bitrate   : Set bitrate. Use bits per second (e.g. 10MBits/s would be -b 1
0000000)
-t, --timeout   : Time (in ms) to capture for. If not specified, set to 5s. Zero
 to disable
-d, --demo      : Run a demo mode (cycle through range of camera options, no cap
ture)
-fps, --framerate       : Specify the frames per second to record
-e, --penc      : Display preview image *after* encoding (shows compression arti
facts)
-g, --intra     : Specify the intra refresh period (key frame rate/GoP size). Ze
ro to produce an initial I-frame and then just P-frames.
-pf, --profile  : Specify H264 profile to use for encoding
-td, --timed    : Cycle between capture and pause. -cycle on,off where on is rec
ord time and off is pause time in ms
-s, --signal    : Cycle between capture and pause on Signal
-k, --keypress  : Cycle between capture and pause on ENTER
-i, --initial   : Initial state. Use 'record' or 'pause'. Default 'record'
-qp, --qp       : Quantisation parameter. Use approximately 10-40. Default 0 (of
f)
-ih, --inline   : Insert inline headers (SPS, PPS) to stream
-sg, --segment  : Segment output file in to multiple files at specified interval
 <ms>
-wr, --wrap     : In segment mode, wrap any numbered filename back to 1 when rea
ch number
-sn, --start    : In segment mode, start with specified segment number
-sp, --split    : In wait mode, create new output file for each start event
-c, --circular  : Run encoded data through circular buffer until triggered then
save
-x, --vectors   : Output filename <filename> for inline motion vectors
-if, --irefresh : Set intra refresh type
-fl, --flush    : Flush buffers in order to decrease latency
-pts, --save-pts        : Save Timestamps to file for mkvmerge
-cd, --codec    : Specify the codec to use - H264 (default) or MJPEG
-lev, --level   : Specify H264 level to use for encoding
-r, --raw       : Output filename <filename> for raw video
-rf, --raw-format       : Specify output format for raw video. Default is yuv
-l, --listen    : Listen on a TCP socket
-stm, --spstimings      : Add in h.264 sps timings
-sl, --slices   : Horizontal slices per frame. Default 1 (off)
```

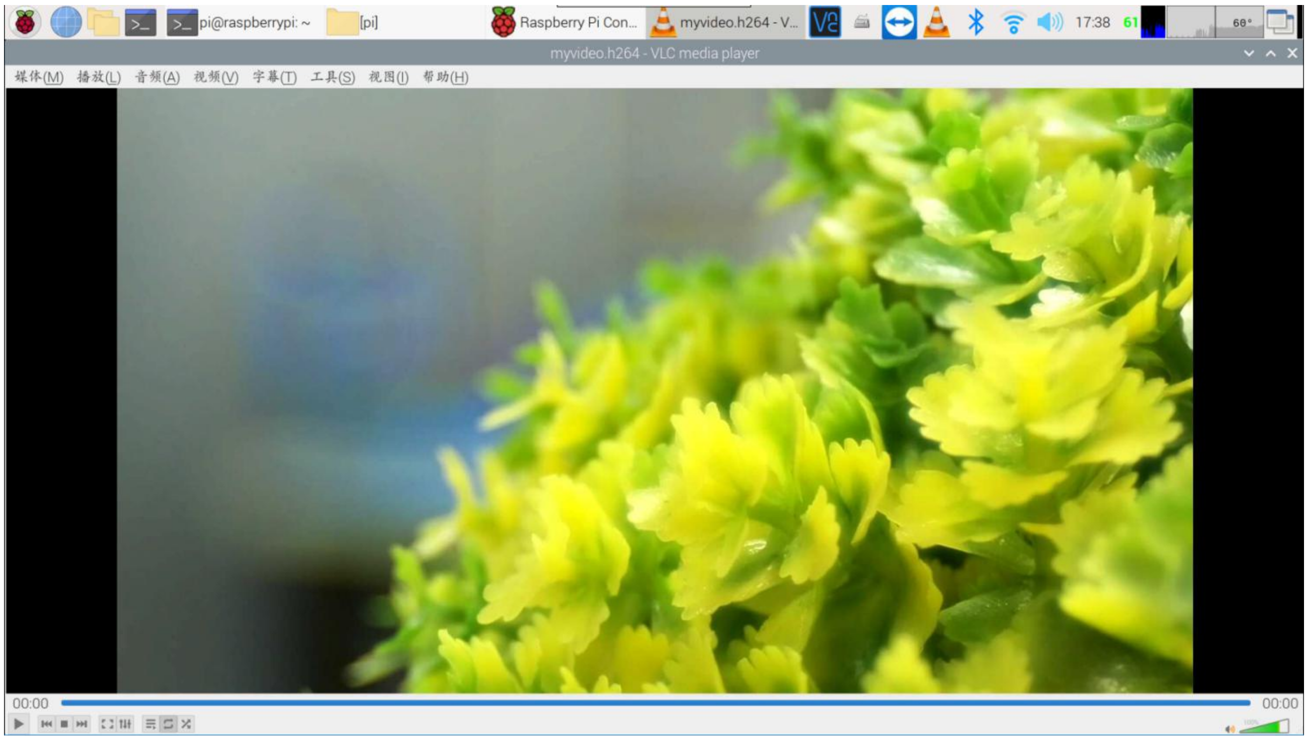
Take a video: The default is video length is 5s, resolution is 1920*1080, frame rate: 25.

raspivid -o myvideo.h264



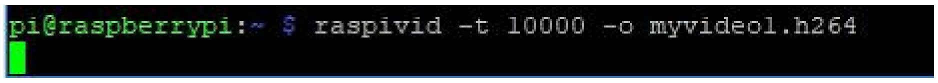
```
pi@raspberrypi:~ $ raspivid -o myvideo.h264
pi@raspberrypi:~ $ ls
audio.mp3  image1.jpg  mu_code         Notebook       Templates
Desktop    image2.jpg  Music           opencv-4.1.2   Videos
Documents  image.jpg   myvideo.h264    Pictures       Yahboom_Project
Downloads  MagPi       NetworkService  Public         yahboom-raspblock
pi@raspberrypi:~ $ 
```

You can see that the myvideo.h264 video has been generated. We can find and open it from Raspberry Pi's file manager.


myvideo.h264 - VLC media player
媒体(M) 播放(L) 音频(A) 视频(V) 字幕(T) 工具(S) 视图(I) 帮助(H)
00:00
00:00

2) Take a 10-second video and save the name as **myvideo1.h264**.

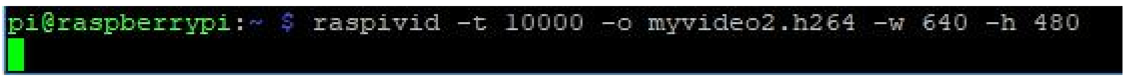
raspivid -t 10000 -o myvideo1.h264



```
pi@raspberrypi:~ $ raspivid -t 10000 -o myvideo1.h264
```

3) Take a 10-second video with 640*480 resolution and save the name as **myvideo2.h264.**

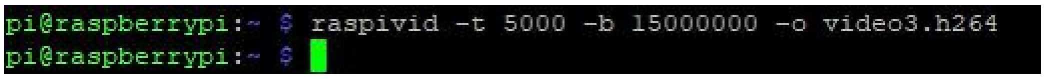
raspivid -t 10000 -o myvideo2.h264 -w 640 -h 480



```
pi@raspberrypi:~ $ raspivid -t 10000 -o myvideo2.h264 -w 640 -h 480
```

4) Take a 5-second video with 1920 * 1080 resolution and a bit rate of 15Mbps, and save it as **myvideo3.h264**. -b sets the bit rate in bits per second.

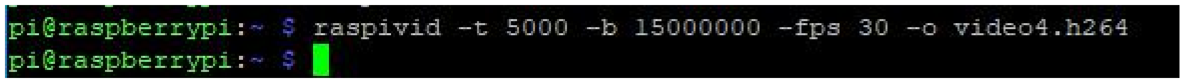
raspivid -t 5000 -b 15000000 -o video3.h264



```
pi@raspberrypi:~ $ raspivid -t 5000 -b 15000000 -o video3.h264
pi@raspberrypi:~ $ 
```

5) Take a 5-second video with 1920*1080 resolution, a bit rate of 15Mbps, and 30 frame rate and the save name is **myvideo4.h264**. -b sets the bit rate, unit is per second.

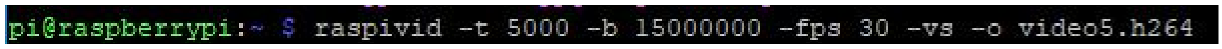
raspivid -t 5000 -b 15000000 -fps 30 -o video4.h264



```
pi@raspberrypi:~ $ raspivid -t 5000 -b 15000000 -fps 30 -o video4.h264
pi@raspberrypi:~ $
```

6) Turn on video stabilization to reduce camera shake. -vs parameter turns on the video stabilization function.

raspivid -t 5000 -b 15000000 -fps 30 -vs -o video5.h264



```
pi@raspberrypi:~ $ raspivid -t 5000 -b 15000000 -fps 30 -vs -o video5.h264
```

7) Divide a 10-second recorded video into 3-second segments and save the name **videoxxxx.h264**. -sg is the time for each segment.

raspivid -t 10000 -sg 3000 -o video%04d.h264

8) Set the color format to yuv.   -rf represents the format, which can be set to **yuv, RGB, gray.**

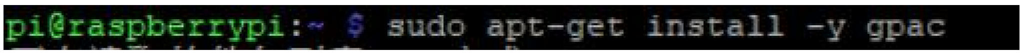
raspivid -t 10000 -rf yuv -o video6.h264

3.Video format conversion

Note: raspivid outputs an uncompressed H.264 video stream. In order to make our ordinary video player can play this video, we need to install gpac package.

1) Input following command:

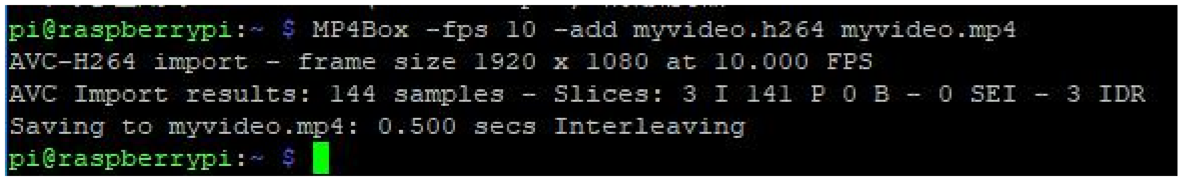
sudo apt-get install -y gpac



```
pi@raspberrypi:~ $ sudo apt-get install -y gpac
```

2)Use the MP4Box application in the gpac package to convert the H.264 format video stream to 10 frames per second MP4 format video.

MP4Box -fps 10 -add myvideo.h264 myvideo.mp4



```
pi@raspberrypi:~ $ MP4Box -fps 10 -add myvideo.h264 myvideo.mp4
AVC-H264 import - frame size 1920 x 1080 at 10.000 FPS
AVC Import results: 144 samples - Slices: 3 I 141 P 0 B - 0 SEI - 3 IDR
Saving to myvideo.mp4: 0.500 secs Interleaving
pi@raspberrypi:~ $
```