

# 合肥工业大学

计算机与信息学院

## 《多媒体技术》课程设计报告 2

专 业 班 级 智能科学与技术 19-1 班

学 生 姓 名 及 学 号 武梓龙 2019212300

指 导 教 师 李佳 胡珍珍

2022 年 6 月 29 日

## 目录

一、RTSP 协议基础.....	3
1、简述 RTSP 协议的基本原理.....	3
1.1. RTSP 客户端状态机.....	4
1.2. RTSP server 保活机制.....	4
2、程序代码.....	4
3、运行代码，并详述一个客户端和服务端的交互过程.....	6
3.1 运行代码.....	6
3.2 客户端与服务端交互过程.....	7
二、流媒体服务器的搭建.....	8
· 实现目标.....	9
· 所用技术：.....	9
1、准备 ffmpeg.....	9
2、下载 nginx.....	10
3、下载 nginx-rtmp-module 插件.....	10
4、配置 conf\nginx-win-rtmp.conf.....	11
5、启动 nginx:.....	11
6、输入 ffmpeg 命令进行推流.....	12
7、打开 VLC 播放器从自己的流媒体服务器上拉流观看.....	14
8、必要的编辑：加上学号.....	15

# 一、RTSP 协议基础

## 1、简述 RTSP 协议的基本原理

### • 简介：

RTSP (Real Time Streaming Protocol) 全称“实时流协议”，是 TCP/IP 协议体系下的一个应用层协议，定义了一对多应用程序如何有效地通过 IP 网络传送多媒体数据，用于多媒体数据的网络控制。

### • 原理：

RTSP 组合使用了可靠传输协议 TCP（控制信息）和高效传输协议 UDP（媒体数据）来串流内容给用户。支持点播和直播。

RTSP 协议本身只负责传输媒体控制信息，并不负责数据传输，使用 RTP (Real-time Transport Protocol) 和 RTCP (Real-time Control Protocol) 完成数据传输和数据检测。

会话参与者(发送端和接收端)周期性的向所有参与者发送 RTCP 包。主要功能是为应用程序提供会话质量或广播性能质量的信息，这些信息包括发送的信息包数目、丢失的信息包数目和信息包抖动等情况。

简而言之：

RTSP (over TCP) --> 媒体控制

RTCP (over UDP) --> 监控媒体数据传输质量

RTP (over UDP) --> 媒体数据传输

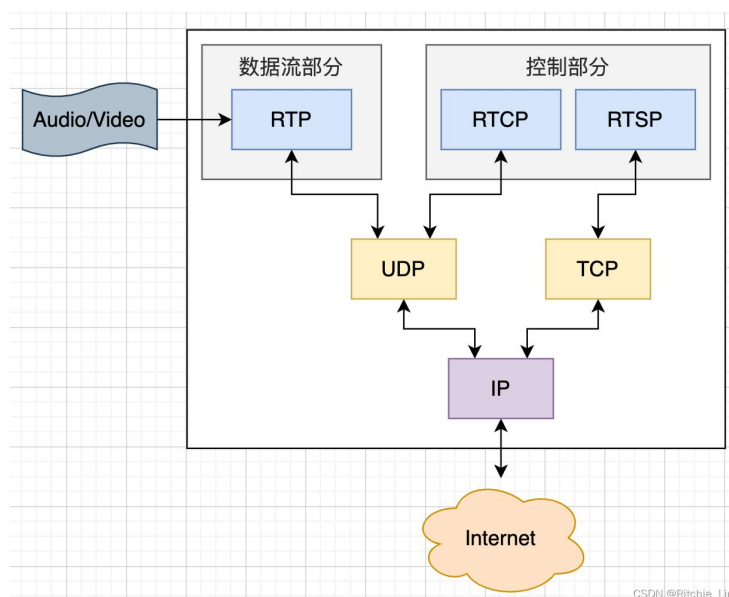
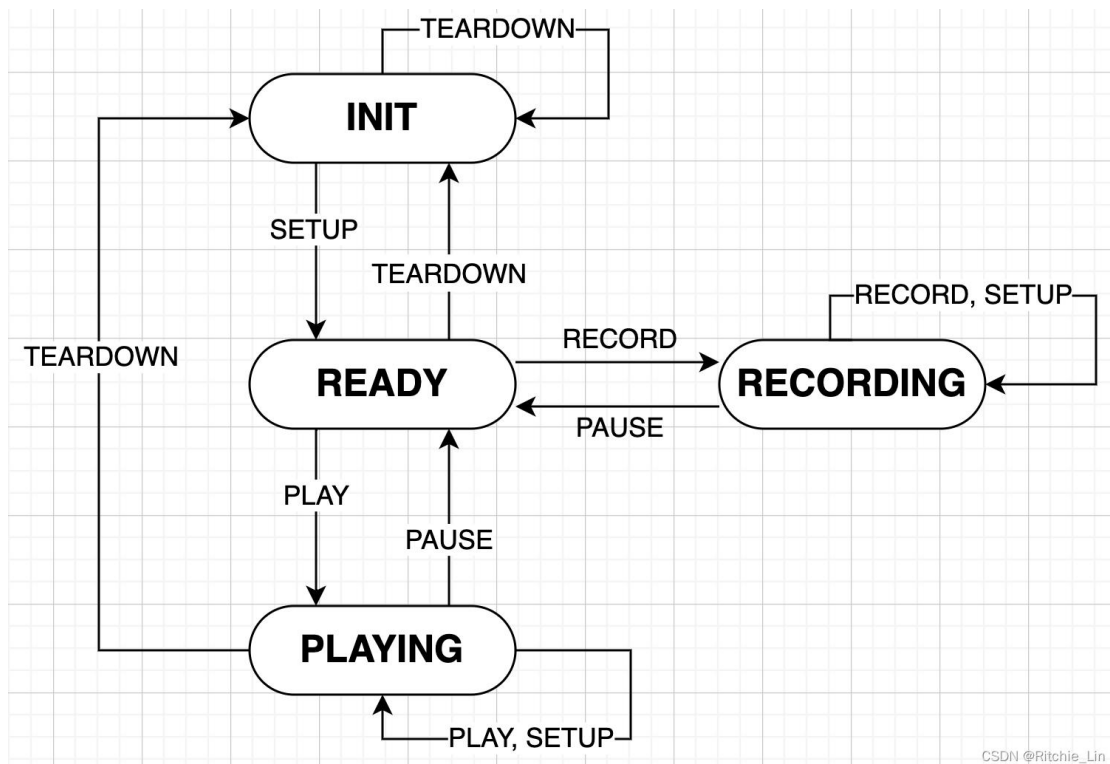


图 1 协议在网络中的结构图

### 1.1. RTSP 客户端状态机

状态机如下图：



- 初始态(Init)：SETUP 请求已经发出，等待回复，尚未创建会话
- 就绪态(Ready)：收到 SETUP 回复，或在播放态收到 pause 回复，会话已创建好，可以进行数据传输。
- 播放态(Playing)：收到 PLAY 回复，媒体数据开始传输，客户端播放媒体。
- 记录态(Recording)：收到 RECORD 回复，客户端开始录制数据。

### 1.2. RTSP server 保活机制

客户端如果一段时间内（默认是 60s）没有任何响应，那么 rtsp 服务器就会关闭该会话，所以客户端需要发送心跳包给服务器。

- rtsp 层面上，定期向 server 发无效的控制信息（要带有 session id 的 cmd，比如，空消息体的 get\_parameter 命令，）
- rtp 层面上，定期向 server 发送 rtp 包，包内容随意。

## 2、程序代码

(1) Client, ClientLauncher：负责启动客户端并发送 RTSP 命令，在代码中，请补全相关的命令。

```

57     def setupMovie(self):
58         """Setup button handler."""
59         if self.state == self.INIT:
60             # self.sendRtspRequest(#writing your code here)
61             self.sendRtspRequest(self.SETUP)

```

```

62     def exitClient(self):
63         """Teardown button handler."""
64         # self.sendRtspRequest(#writing your code here)
65         self.sendRtspRequest(self.TEARDOWN)
66         self.master.destroy() # Close the gui window

```

```

70     def pauseMovie(self):
71         """Pause button handler."""
72         if self.state == self.PLAYING:
73             self.sendRtspRequest(self.PAUSE)

```

```

74     def playMovie(self):
75         """Play button handler."""
76         if self.state == self.READY:
77             # Create a new thread to listen for RTP packets
78             threading.Thread(target=self.listenRtp).start()
79             self.playEvent = threading.Event()
80             self.playEvent.clear()
81             # self.sendRtspRequest(#writing your code here)
82             self.sendRtspRequest(self.PLAY)

```

```

102         if self.teardownAcked == 1:
103             # self.rtpSocket.shutdown(#writing your code here)
104             self.rtpSocket.shutdown(socket.SHUT_RDWR)

```

(2) Server, ServerWorker: 负责相应 RTSP 命令，请补全相关的状态转移。

```

46     try:
47         self.clientInfo['videoStream'] = VideoStream(filename)
48         # self.state = #writing your code here
49         self.state = self.READY
50     except IOError:

```

```

58         elif requestType == self.PLAY:
59             if self.state == self.READY:
60                 print_~("processing PLAY\n")
61                 # self.state = #writing your code here
62                 self.state = self.PLAYING
63                 self.clientInfo["udpSocket"] = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

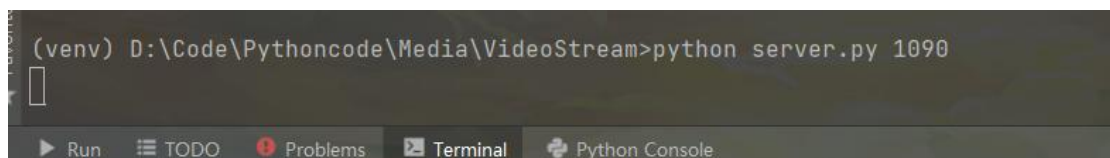
70         elif requestType == self.PAUSE:
71             if self.state == self.PLAYING:
72                 print_~("processing PAUSE\n")
73                 # self.state = #writing your code here
74                 self.state = self.READY
75                 self.clientInfo["udpSocket"].setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)

```

### 3、运行代码，并详述一个客户端和服务端的交互过程

#### 3.1 运行代码

python server.py 1090

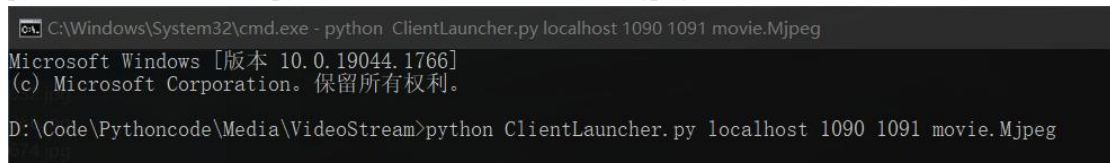


```

(venv) D:\Code\Pythoncode\Media\VideoStream>python server.py 1090

```

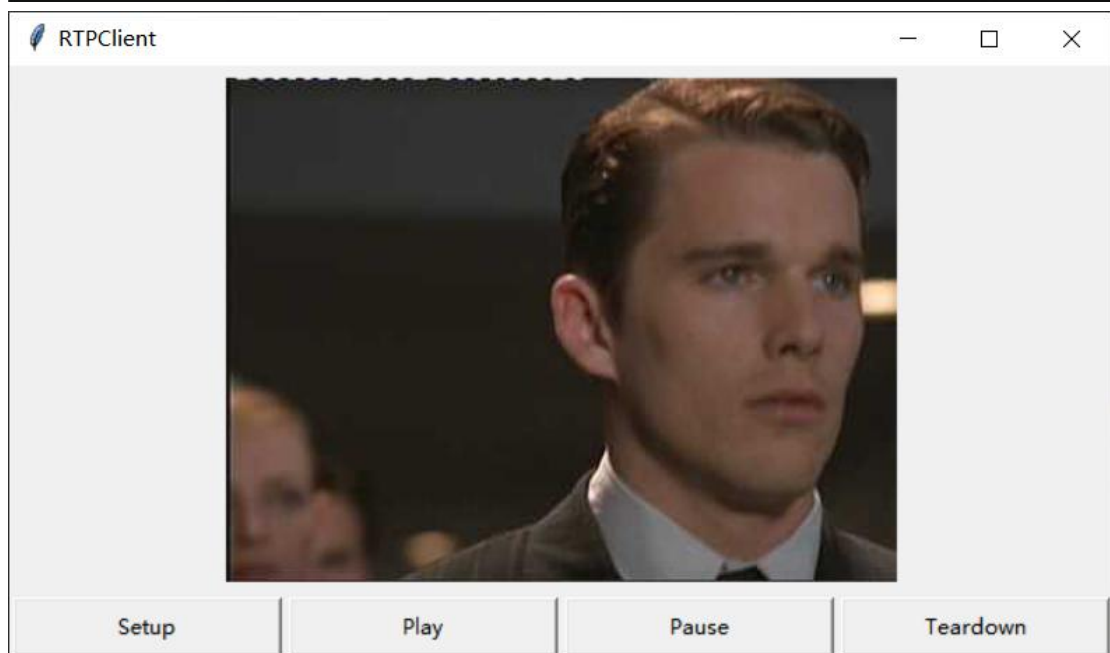
python ClientLauncher.py localhost 1090 1091 movie.Mjpeg

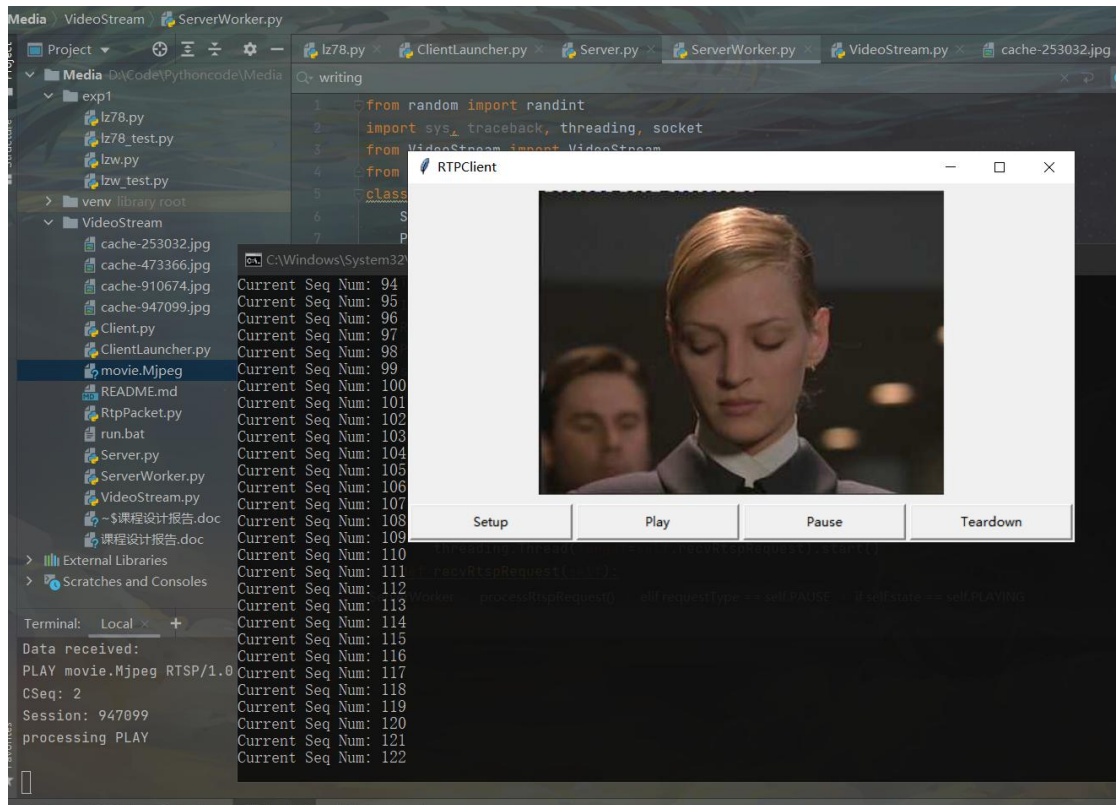


```

C:\Windows\System32\cmd.exe - python ClientLauncher.py localhost 1090 1091 movie.Mjpeg
Microsoft Windows [版本 10.0.19044.1766]
(c) Microsoft Corporation。保留所有权利。
D:\Code\Pythoncode\Media\VideoStream>python ClientLauncher.py localhost 1090 1091 movie.Mjpeg

```





### 3.2 客户端与服务端交互过程



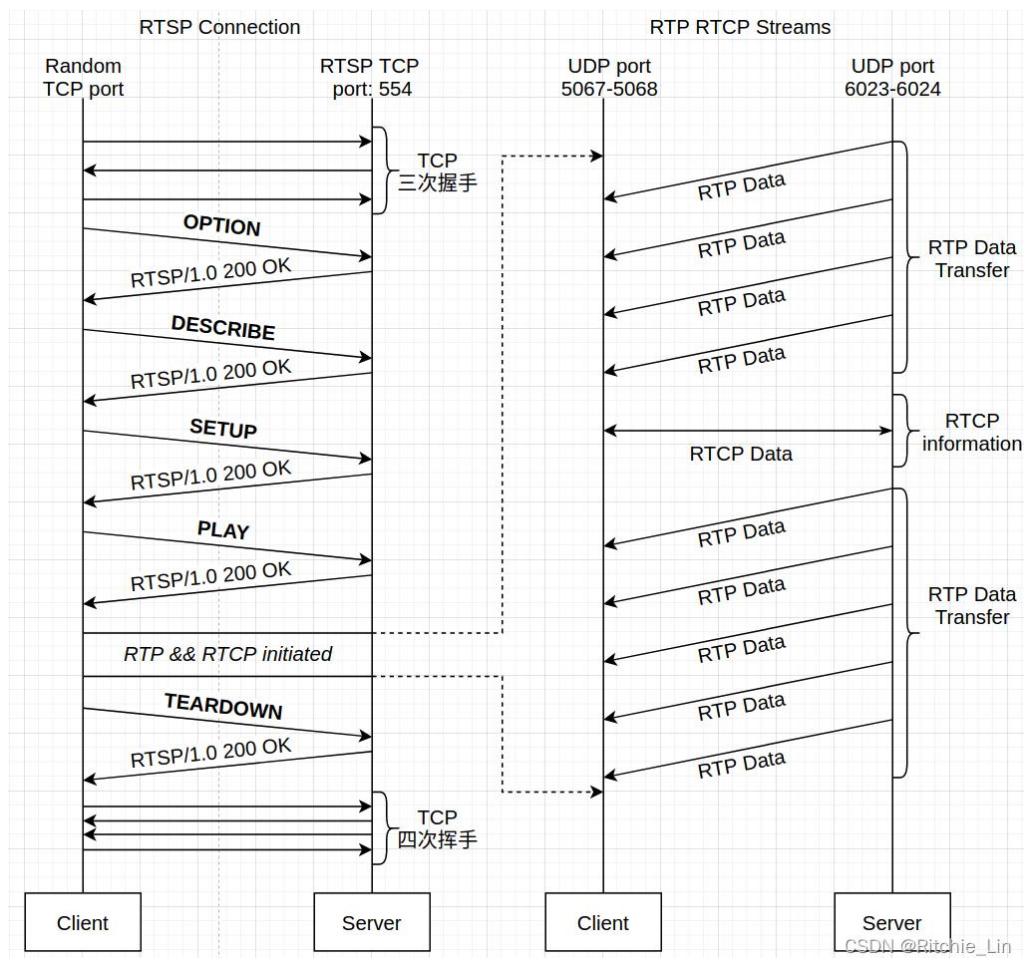


图 2 会话交互过程

- 1、首先发 OPTION cmd 请求服务器可使用的方法。
- 2、然后发 DESCRIBE cmd 请求服务器返回媒体的 metadata, 对于 audio 和 video 分开传输的情况, 返回信息里面还有各自的 url。
- 3、然后发送 SETUP cmd 请求服务器创建会话, 请求体中需带数据传输协议以及客户端 RTCP 和 RTP 的接收端口号。  
服务器返回会话 ID 和对应的服务器端口, 客户端拿到服务器 IP 和端口, 通过 RTCP 和 RTP 协议与服务器连接。
- 4、最后发送 PLAY cmd 请求服务器传输数据, 此时客户端开始接收数据并播放。
- 5、播放结束后, 发送 TEARDOWN cmd 请求服务销毁会话。

## 二、流媒体服务器的搭建

采用 ffmpeg+Ngnix 方案, 从摄像头采集数据, 进行必要的编辑 (比如: 左下角加上学号) 和处理 (选作: 人脸识别), 通过流媒体协议传输 (rtmp、hls、http-flv 任选), 并在客户端进行播放。

参考

[https://zjdoc-stream.readthedocs.io/zh\\_CN/latest/push/ffmpeg-rtsp-rtmp/](https://zjdoc-stream.readthedocs.io/zh_CN/latest/push/ffmpeg-rtsp-rtmp/)



[https://github.com/ageitgey/face\\_recognition](https://github.com/ageitgey/face_recognition)

## · 实现目标

将电脑摄像头捕捉到的视频制作成视频流 nginx+ nginx-rtmp-module

## · 所用技术:

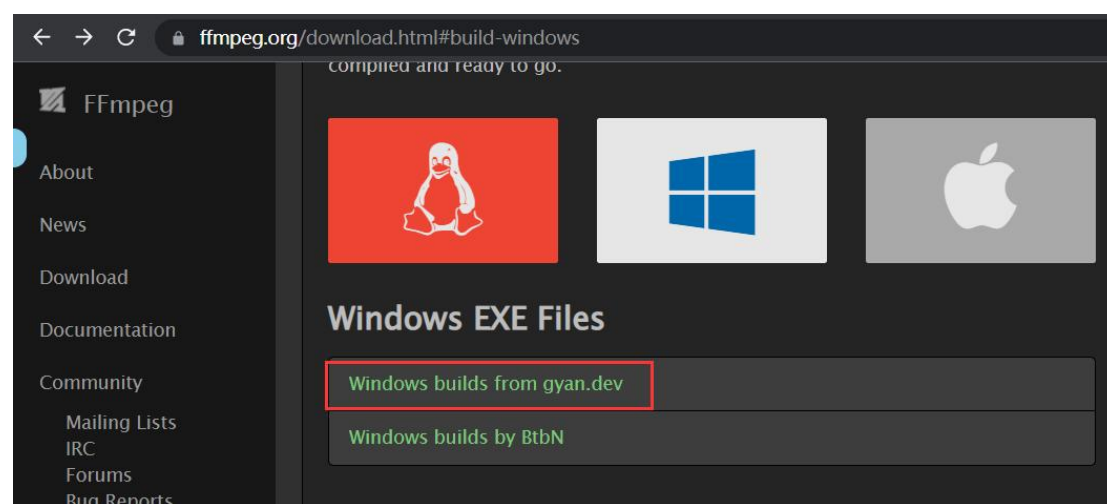
- FFmpeg 是一套可以用来记录、转换数字音频、视频，并能将其转化为流的开源计算机程序。采用 LGPL 或 GPL 许可证。它提供了录制、转换以及流化音视频的完整解决方案。它包含了非常先进的音频/视频编解码库 libavcodec，能够解码、编码、转码、复用、解复用、流化、滤波和播放几乎任何人类和机器创造的多媒体文件。它具有高可移植性、高性能、高度安全、高度易用性、支持的格式多样性、高度可扩展等特点。

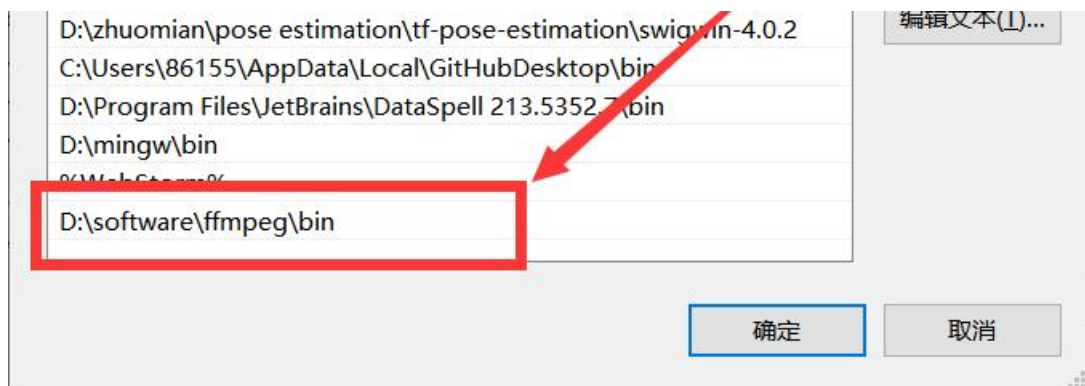
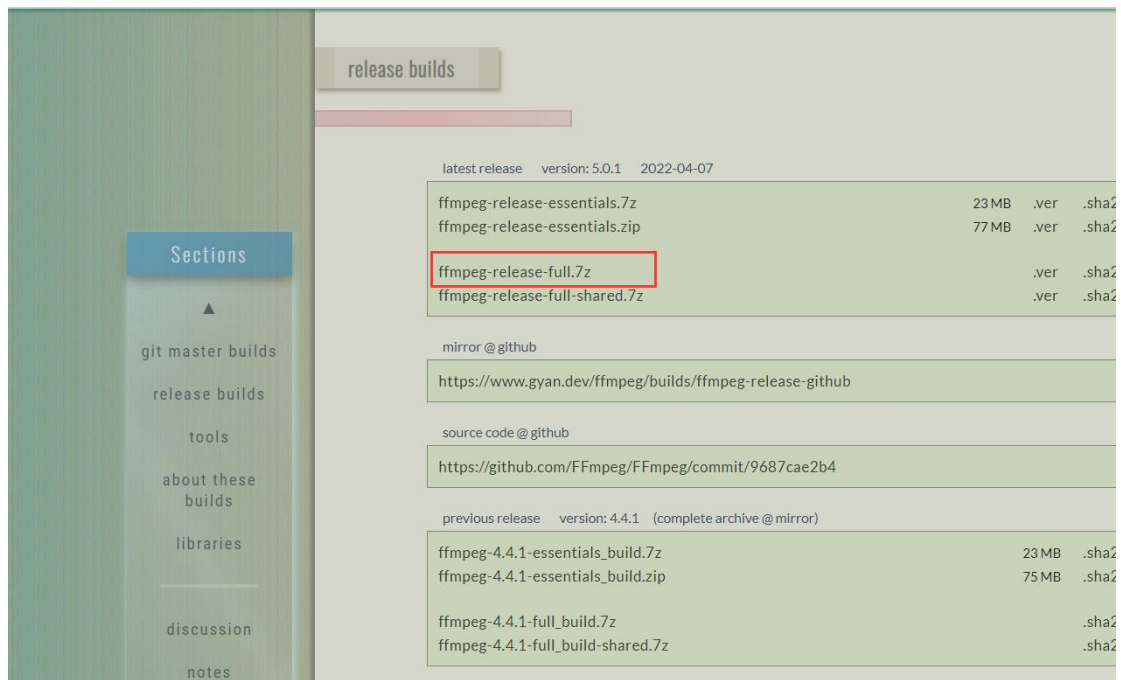
- Nginx 本身是一个非常出色的 HTTP 服务器,ffmpeg 是非常好的音视频解决方案.这两个东西通过一个 nginx 的模块 nginx-rtmp-module,组合在一起即可以搭建一个功能相对比较完善的流媒体服务器.这个流媒体服务器可以支持 RTMP 和 HLS(Live Http Stream)。

- nginx 配合 ffmpeg 做流媒体服务器的原理是:

nginx 通过 rtmp 模块提供 rtmp 服务, ffmpeg 推送一个 rtmp 流到 nginx, 然后客户端通过访问 nginx 来收看实时视频流. HLS 也是差不多的原理,只是最终客户端是通过 HTTP 协议来访问的,但是 ffmpeg 推送流仍然是 rtmp 的.

## 1、准备 ffmpeg



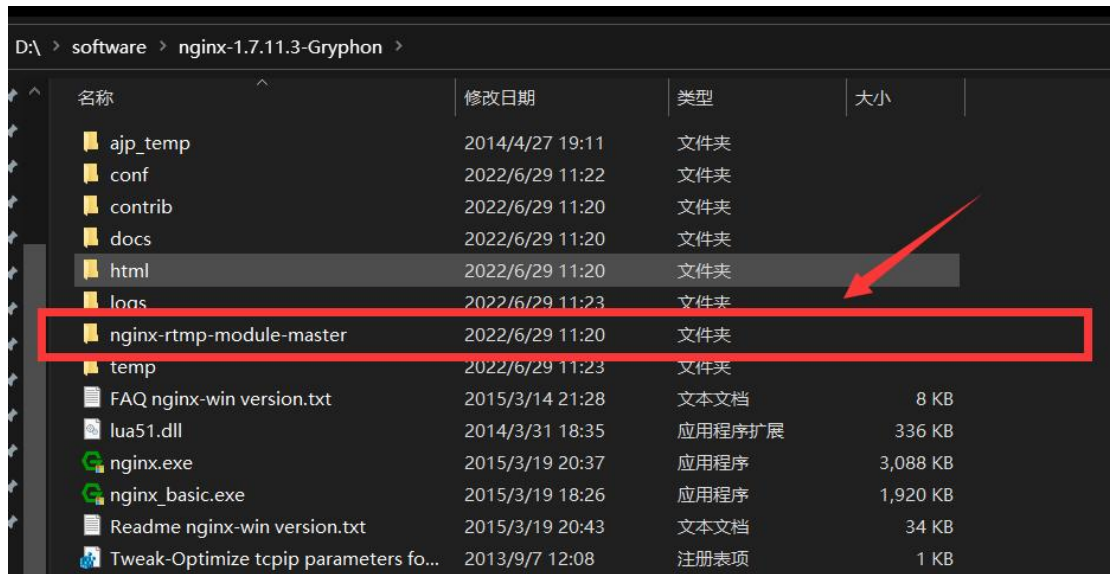


## 2、下载 nginx

下载链接: [http://nginx-win.ecsds.eu/download/nginx 1.7.11.3 Gryphon.zip](http://nginx-win.ecsds.eu/download/nginx%201.7.11.3%20Gryphon.zip),  
 下载完成后解压到需要盘符, 将解压后的目录命名为 nginx-1.7.11.3-Gryphon

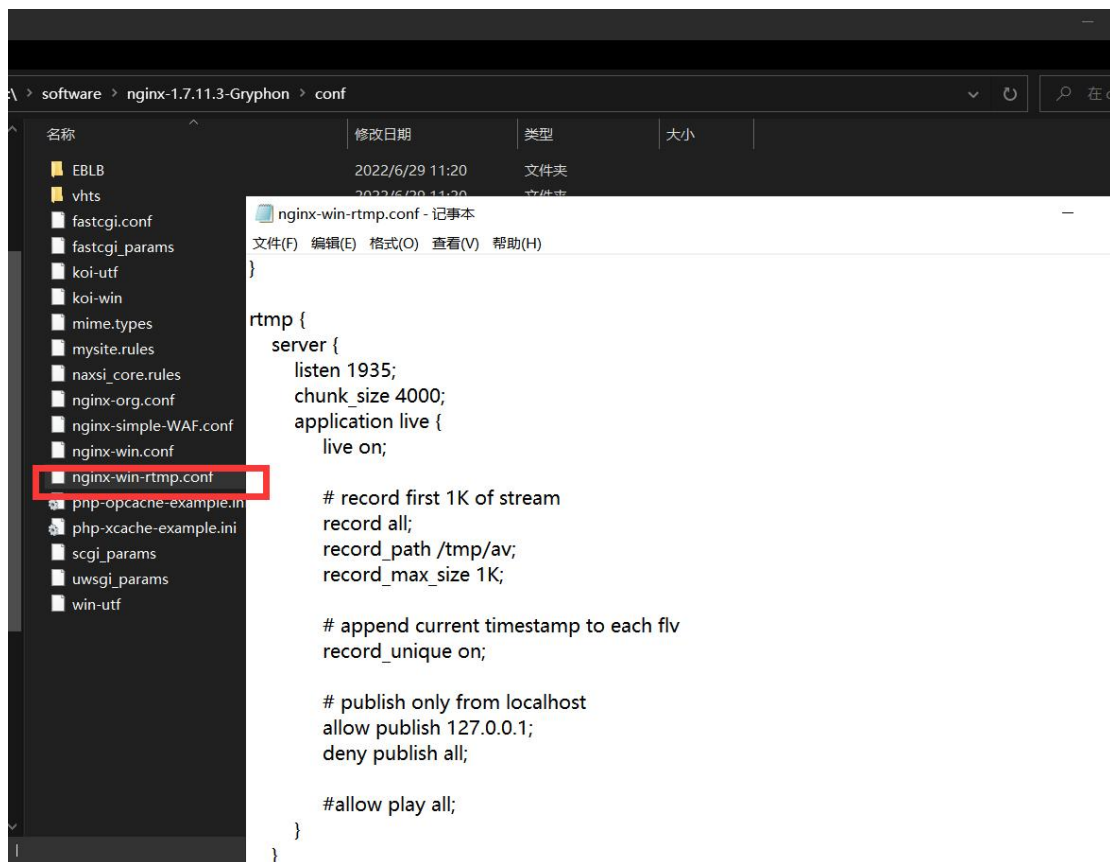
## 3、下载 nginx-rtmp-module 插件

下载地址 <https://github.com/arut/nginx-rtmp-module/>  
下载完成后解压到刚刚解压的 nginx-1.7.11.3-Gryphon 目录中



#### 4、配置 conf\nginx-win-rtmp.conf

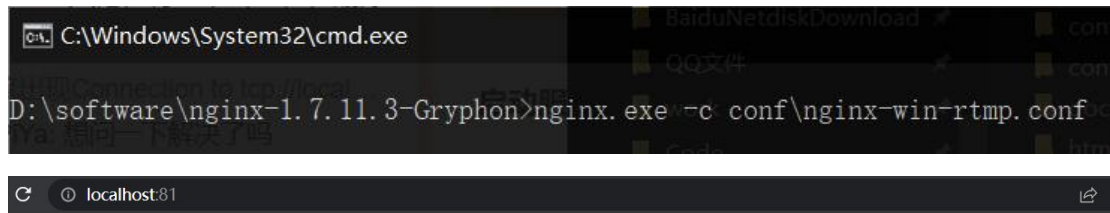
配置 nginx-1.7.11.3-Gryphon 文件下 conf\nginx-win-rtmp.conf 内容:



#### 5、启动 nginx:

打开 cmd 窗口进入 D:\software\nginx-1.7.11.3-Gryphon 目录输入  
打开 nginx:

```
D:\software\nginx-1.7.11.3-Gryphon>
nginx.exe -c conf\nginx-win-rtmp.conf
```



## Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](http://nginx.org).  
Commercial support is available at [nginx.com](http://nginx.com).

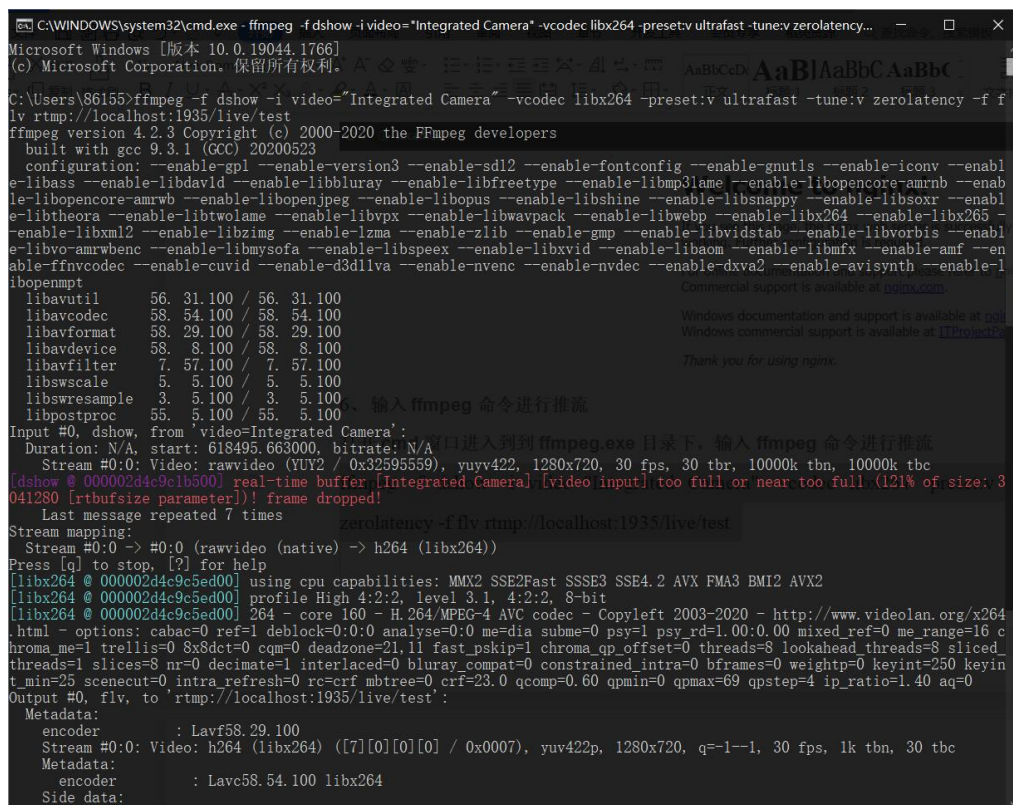
Windows documentation and support is available at [nginx for Windows](http://nginx.org/en/windows).  
Windows commercial support is available at [ITProjectPartner](http://projectpartner.com).

Thank you for using nginx.

## 6、输入 ffmpeg 命令进行推流

①打开 cmd 窗口进入到 ffmpeg.exe 目录下，输入 ffmpeg 命令进行推流

```
ffmpeg -f dshow -i video="Integrated Camera" -vcodec libx264 -preset:v ultrafast -tune:v
zerolatency -f flv rtmp://localhost:1935/live/test
```



②也可以使用 python 程序进行推流:

```

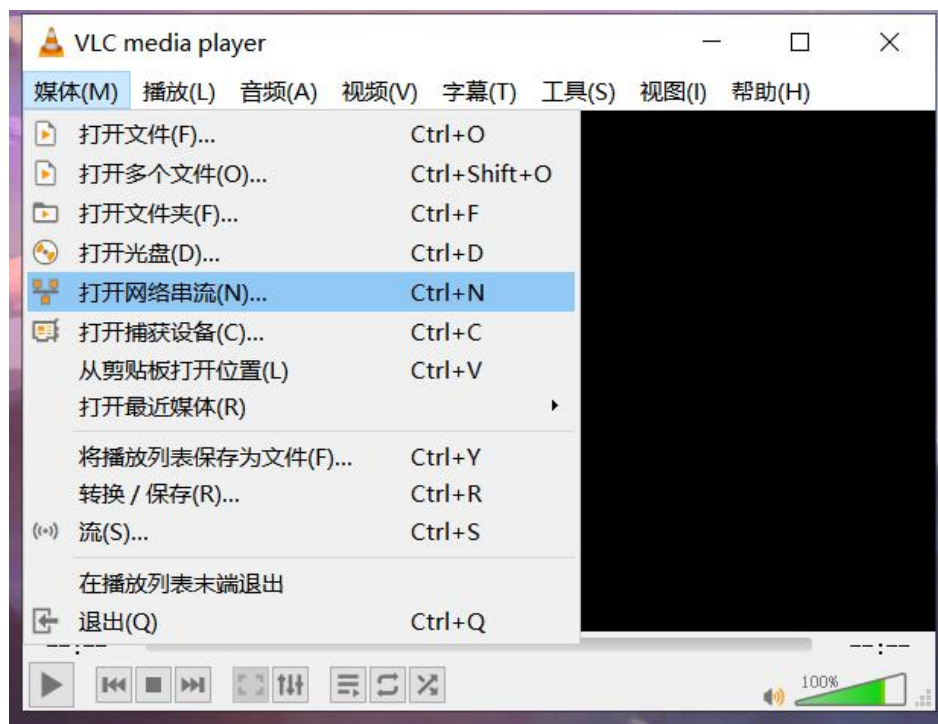
1.  # -*- coding: utf-8 -*-
2.  # @Time : 2022/6/23 15:09
3.  # @Author : 武梓龙 2019212300
4.  # @File : FFMPEG.py
5.  # @Software: PyCharm
6.  import cv2
7.  import subprocess as sp
8.
9.  way = 'rtmp'
10. push_url = "rtmp://localhost:1935/live/test"
11. camera_path = 0
12.
13. cap = cv2.VideoCapture(camera_path)
14.
15. # Get video information
16. fps = int(cap.get(cv2.CAP_PROP_FPS))
17. width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
18. height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
19. print(fps, width, height)
20. # ffmpeg command
21. command = ['ffmpeg',
22.             '-y',
23.             '-f', 'rawvideo',
24.             '-vcodec', 'rawvideo',
25.             '-pix_fmt', 'bgr24',
26.             '-s', "{}x{}".format(width, height),
27.             '-r', str(fps),
28.             '-i', '-',
29.             '-c:v', 'libx264',
30.             '-pix_fmt', 'yuv420p',
31.             '-preset', 'ultrafast',
32.             '-f', 'flv',
33.             push_url]
34.
35. # 管道配置
36. p = sp.Popen(command, stdin=sp.PIPE)
37. while (cap.isOpened()):
38.     ret, frame = cap.read()
39.     # print("running.....")
40.     if not ret:
41.         print("Opening camera is failed")
42.         break
43.     p.stdin.write(frame.tobytes())
44.

```



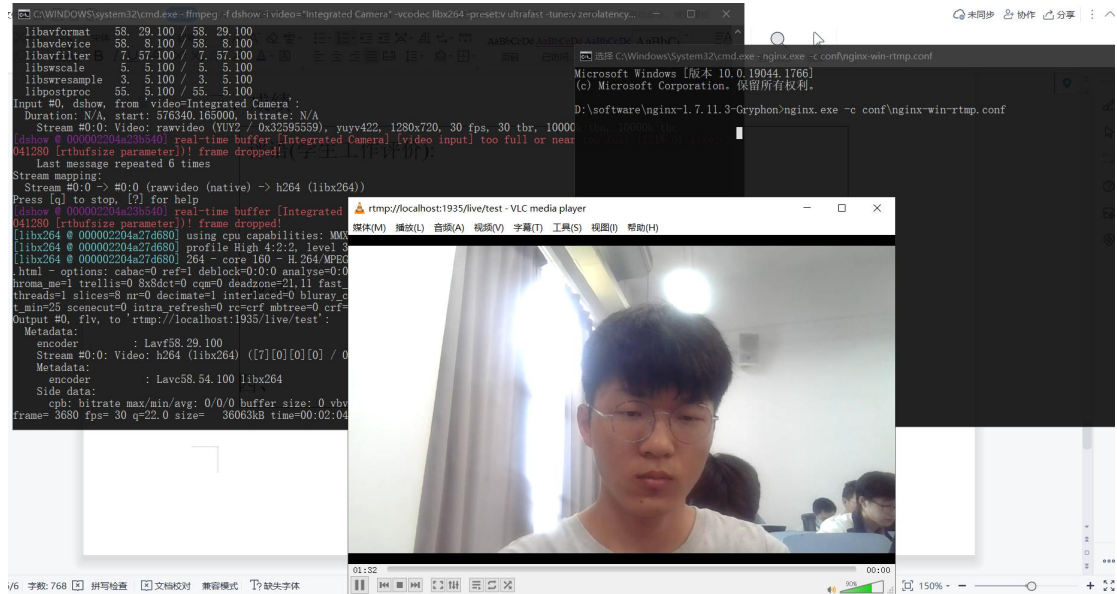
45. `return_value, frame = cap.read()`

## 7、打开 VLC 播放器从自己的流媒体服务器上拉流观看



`rtmp://localhost:1935/live/test`



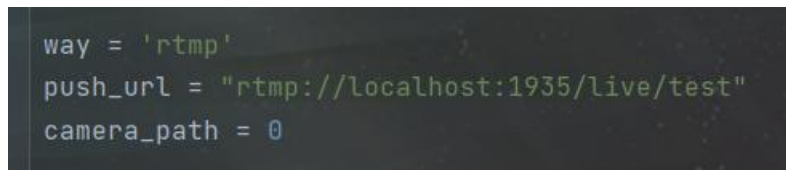


## 8、必要的编辑：加上学号

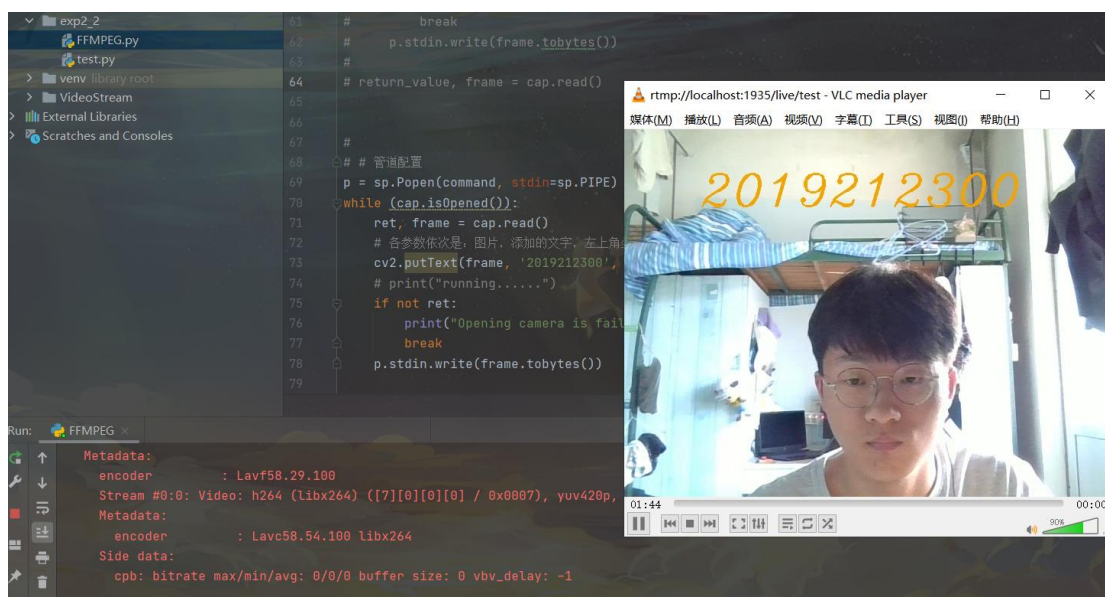
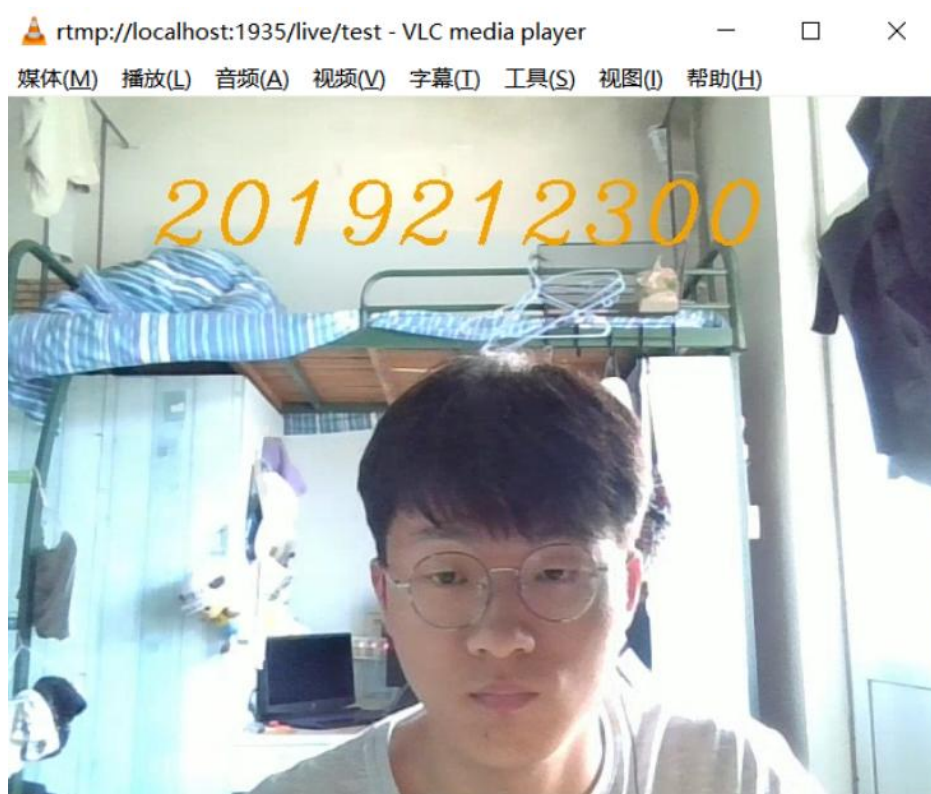
在步骤 6 中的代码中加入以代码片段：

```
cv2.putText(frame, '2019212300', (100,100), cv2.FONT_HERSHEY_SCRIPT_COMPLEX, 2, (0, 165, 255), 2)
```

# 各参数依次是：图片，添加的文字，左上角坐标，字体，字体大小，颜色，字体粗细







## 一、成绩

评语(学生工作评价):

成绩：

指导教师签字：

二、

年

月

日