RTP 实验报告

唐建宇 2017012221

环境：Windows 10

Python 3.6

一、代码结构

```
Server/
      GBN_Sender.py          // 基于 UDP 实现的 GBN 可靠传输的发送方
      Server.py              // 服务器逻辑
      RtpPacket.py           // RTP 打包/解码
      VideoLoader.py         // 加载视频用
      videos/                // 视频库，里面存放视频文件

Client/
      GBN_Receiver           // 基于 UDP 实现的 GBN 可靠传输接收方
      RtpPacket.py           // RTP 打包/解码
      Client.py              // 客户端入口及主界面（播放列表和搜索界面）
      RtpClient.py           // 播放视频的界面及逻辑
      SubtitleLoader.py      // 加载字幕用
```

二、客户端

客户端分为两个窗口，用户进入程序时为主窗口，有搜索框和视频列表，用户选择视频点击播放就会调出播放界面。

主界面

上方为搜索框和按键、中间是播放列表。在列表中选择一个视频（默认为第一项），点击播放后则进入播放界面播放。点击搜索列表会出现搜索结果，右下角清空键可清空搜索结果，列表重新加载所有视频。

播放界面

进度条可拖动并定位。下方按键依次为播放（播放状态时自动切换为暂停）、全屏、1 倍速、1.5 倍速、0.5 倍速。使用空格键也可在播放和暂停间切换。

进入全屏状态可使用 ESC 键退出。

三、实现功能

基本功能

- RTSP 控制指令：SETUP/PLAY/PAUSE/DESCRIPTION/REPOSITION
- 有可拖动的进度条

- 支持 0.5、1、1.5 倍速播放
- 支持 mp4 等格式
- 支持多客户端连接

额外功能

- 支持 srt 格式**字幕**

  用户可在客户端目录中 srt 目录下存放字幕文件，播放视频时自动加载与视频同名的字幕文件（字幕功能需要字幕编码为 GB2312 格式）

- **多种视频格式**

  包含 mp4、avi 等

- **播放列表**

  起始界面为所有视频的列表，可从中选择播放

- **搜索**视频

  可在其实界面的搜索栏中通过关键字搜索服务器存放的视频

- **全屏**

  支持全屏播放

- **快捷键**

  - 全屏时按下 ESC 即可退出全屏
  - 播放界面（包括全屏时）按下空格可在暂停和播放间切换

- **自动调整画质**

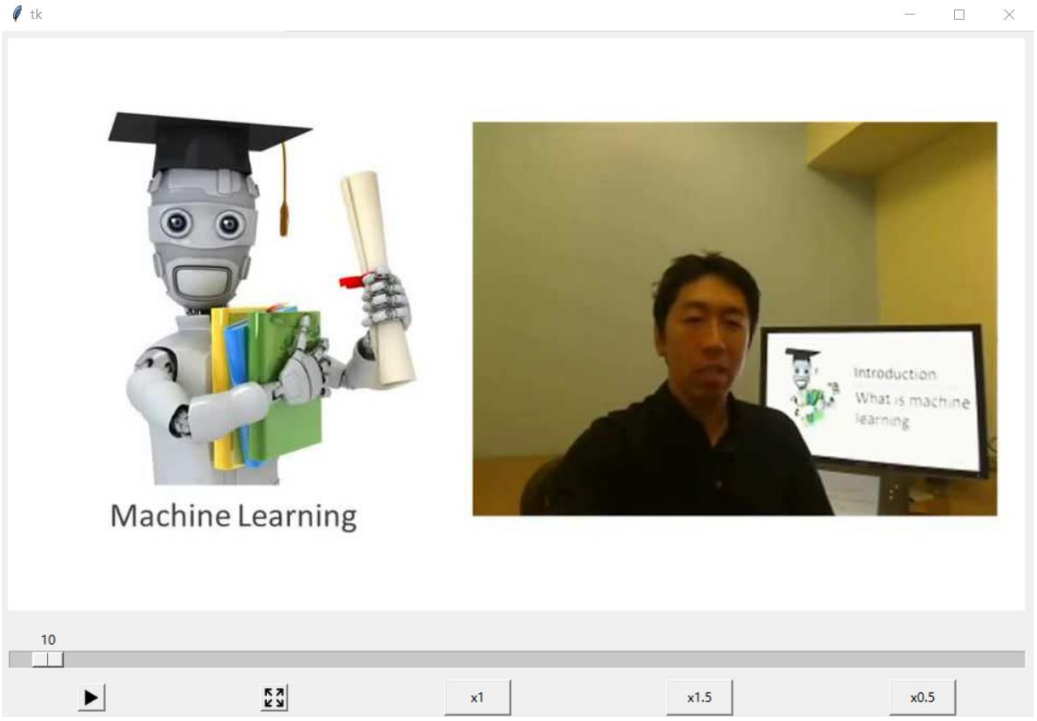
  发生丢包时自动降低画质以缓解网络压力

- **缓存机制**

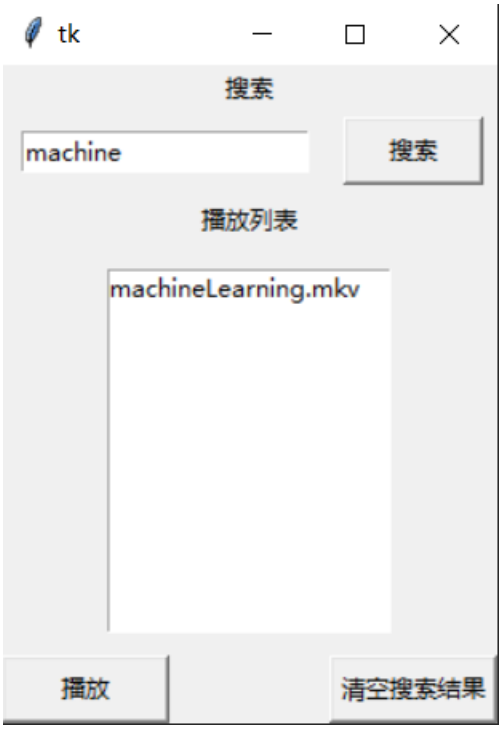
  服务器以略快于播放帧率的速度发送数据，客户端会先缓存起来，播放流畅且内存占用较少

- 基于 UDP 实现了可靠数据传输——**GBN**

四、运行时的截图



（播放 mkv 视频）



（根据关键词搜索）

（字幕）

五、难点与解决方案

1. 按视频原帧率播放

在 task1 中，播放的策略是来一帧播放一帧，这样无法按照视频本来的帧率进行播放，因此采用了两个独立的线程来完成接收 rtp 包和渲染这两个任务。接收线程只负责将收到的 rtp 包解码、以及完成大小和格式的转换再存入 buffer 中；渲染线程只负责按照视频帧率定时访问 buffer 调取需要的帧进行渲染。这样两边互不干扰，也因为有缓存所以播放流畅。

2. 丢包处理

检测到丢包时，将画质降低至当前的 1/2 。画质采用 opencv 的 imencode 编码函数的参数 quality 进行衡量。

3. 字幕

采取的策略是先将 srt 格式解码，按照每一秒存入内存的 buffer 中。在渲染线程中，每隔一秒询问是否当前这一秒有字幕，如有则将内容写入一个 label 上然后定位到视频的底部渲染；如没有视频则将 label 通过 place_forget 方法隐藏。

4. 全屏

tkinter 中不能直接将某个控件设置为全屏，只能将整个窗口全屏。因此首先隐藏除渲染视频的 canvas 外所有控件，将窗口的属性设置为全屏，将 canvas 的大小调整到与屏幕相同，增加 ESC 键盘事件的绑定用于退出。退出时重新渲染控件。

六、总结

这次实验让我熟悉了对多媒体流文件的处理，尤其是 opencv 中视频相关的类和函数的使用；熟悉了多线程编程。第一次接触了 UDP 的 socket 编程，也对包括 UDP 和 TCP 的整个 socket 编程的套路也更熟练了。