基于opencv模块的视频终端状态测试方法研究

姜家文　许荣胜　叶舟宇
中国酒泉卫星发射中心　甘肃酒泉　736200

摘　要：随着网络摄像机价格的平民化，视频监控系统广泛应用于公共和单位内部，区域管理的摄像机数量爆炸式增长，在面向工程过程和设备状态监视的监控系统，视频终端作为重要的图像信息来源，保证其正常工作状态是日常维护的重点，但目前通过传统人工方式对设备工作状态进行目测检查不仅费时费力，无法实时获取设备图像的告警信息。文中基于开源opencv模块，使用B/S架构，采取多线程处理的方式对视频终端进行码流抓取，测试终端状态，省时省力，解决了必须用人去实时监控图像状态的痛点。

关键词：opencv；flask；码流抓取；python

引言

由于视频系统的飞速发展，很多单位的视频系统是经过多年的逐次、逐步建设，终端数量多、类别多是其主要特点，如海康、大华、天地伟业、以及部分厂家私有协议编码器，数量在数百路左右。系统内除了网络摄像机外，其中还有相当部分极为重要的关键设备状态参数通过计算机编码的方式入网，各类企业、园区、单位内部都有类似情况，但大部分仍采用人工维护检查的方式，对于告警实时响应能力欠缺。

对于一般的视频系统，目前值班人员的工作方式仍比较传统，主要靠人工在工作开始前进行逐个切换检查，播放正常表明终端工作正常，无法播放则说明终端出现问题。进入工作时段后由于需要同时开展业务工作，无法再执行人工观察，采取对于某几个关键视频源通过电视墙进行实时人工监控。这种人工方法效率低下，需要手动每一路切换图像，整个过程大概需要1小时。而且人肉眼观察，在终端数量众多的情况下，容易出现视觉疲累而错过问题，即使在电视墙上显示的关键画面，有时人员忙于其他工作，也会存在长时间未发现图像问题的可能，岗位人员劳动强度大且存在长时间的系统监测盲区。为解决人工监测的问题，本文基于opencv开源计算视觉和机器学习软件库对多终端进行码流抓取适配，以此测试终端状态。为方便使用，通过浏览器将需要监控视频终端信息发送到服务器，此程序采用B/S架构，多线程处理查询，快而精准，解决了值班人员实时监控终端状态重复性工作的痛点[1]。

一、设计思路

opencv是一个开源的计算机视觉和机器学习软件库，特别适合对各种视频终端码流的抓取。结合目前工作系统有Linux也有Windows的实际，浏览器则可以两个系统通用，为扩大此软件适用范围，故选用B/S架构。前端使用标准的html、CSS和JavaScript开发，使用浏览器打开网页使用。后端为与Flask框架兼容，采用python版本的opencv进行码流处理[2]。

工作流程如下图1所示，软件运行逻辑为，用户使用浏览器打开页面，选取设备型号，不同型号设备页面发生改变。根据不同型号，码流抓取的方法不同，需要输入参数，而后点击添加按钮，将该视频设备加入一个监控列表。当用户将需要监控的所有设备加入监控列表后，选择间隔测试时间，点击开始监控。页面将该监控列表数据发送到服务器，服务器根据列表中内容对每个视频终端通过opencv进行一次码流测试，测试成功认为该设备工作正常，测试失败该设备出现问题。测试完毕后，将结果写入字典并返回给页面，将测试成功的设备写入日志，测试失败的设备写入报警框，并给出提醒。页面通过JS定时器，按照时间间隔向服务器发起一次轮询，直至用户点击停止按钮后结束。

二、分块解析

根据软件功能和需求，分块解析关键步骤。

（一）opencv的码流获取

opencv中有一个VideoCapture对象可以对正确格式的串流进行播放，通过调用该对象isOpened的方法则可以判断该对象的串流是否可以播放。因此需要获取各

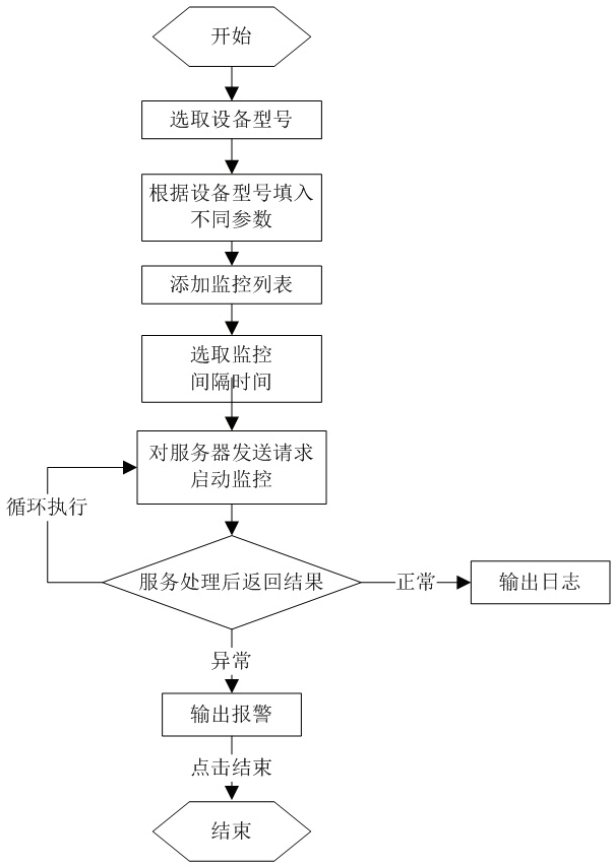

图1　视频终端监控流程图

类型图像终端的串流地址。通过资料收集和测试，获得了以下可用的串流地址。以下每行代表一类设备的串流地址，依次分别为海康、大华、天地伟业和其他类型私有类型的编码器。对于私有类型的编码无法直接获取其串流，但是编码器一般有向某个IP地址和端口号发送UDP码流的功能，开启该功能并向服务端发送，再通过opencv测试UDP码流，就可以对编码器工作状态进行判断。

```
rstp://[username]:[password]@[ip]:[port]/[codec]/[channel]/[subtype]/av_stream
rstp://[username]:[password]@[ip]:[port]/cam/realmonitor?channel=1&subtype=0
rstp://[ip]/MediaInput/h264/stream_1
udp://@1.1.1.1:10001
```

（二）页面设计

根据软件设计思路，前端页面如下图2所示。有两个多选框，分别用于存放可监控编码器的类型和监控时间间隔。随着设备测试码流增多，编码器类型可以逐渐增加。有3个输入框，分别用于输入终端设备的用户名、密码和IP地址。5个按钮，分别用于添加测试列表、清空列表、开始和停止测试、清空测试日志和清空告警日志。为提高辨识程度，告警日志文字被标识为红色。3个可读文本框，分别用于显示测试终端列表、测试日志和告警日志[3]。

图2　视频终端监控前端网页页面

（三）前后端数据交互

前端使用JavaScript中的fetch api发送设备键值信息，选择post作为方法，json作为数据承载载体。后端使用flask作为路由实现，接收fetch api发送的json信息，根据前端设备videoType字段拼接不同的串流字符串，通过opencv进行码流判断。获得的结果保存为字典，格式化为json后返回给前端。前端JavaScript根据服务器返回结果，将状态正常的设备显示在测试日志中，将状态异常的设备放在告警日志中。由于软件主要要用于监控正常的视频终端设备，一般告警日志为空，一旦页面出现红色就代表设备出现异常情况，通知值班人员及时前去处理。

（四）其它优化

（1）前端增加易用性。根据用户选择设备类别的不同，默认用户名自动更改，天地伟业摄像机默认用户名为system，其他设备均为admin；选择私有编码设备则给出弹窗警告，提示用户使用该软件进行监控前，必须先对设备进行配置，给出服务器接收数据的IP地址和端口号。

（2）后端优化串流测试。在给opencv模块测试串流前，先对视频终端进行ping测试，如果设备掉线或掉电，串流ping测试则失败。同时由于ping测试时间远小于网络不通时串流测试时间，增加此操作后，对不在线的设备可大大缩短串流测试时间，从而缩短整体系统的轮巡测试周期。

（3）多线程测试。在python中，将单线程测试改为多线程测试。设备串流测试是并行关系，不存在前后依赖，因此在采用多线程可以大大缩短测试时间。

三、软件实现

（一）opencv串流测试

在python中导入opencv模块，使用该模块进行串流测试。为实现将所有设备进行测试，将设备信息放在dev字典中，再循环调用测试方法，下面的例子为使用opencv对海康摄像机进行测试，使用isOpened方法进行串流测试，结果若为真，设备工作正常，为假设备异常。

```
video_stream_path= “rtsp://”+dev[ “name”]+ “:”+dev[ “password ”]+ “@” +dev[ “ip” ]+ “/h264/1/main/av_stream”
cap=cv2.VideoCapture(video_stream_path)
cap.isOpened()
```

（二）前端页面

使用html和JavaScript共同编写前端页面。为减少工作量，行内CSS仅在告警日志中改变字体颜色使用。为便于更改，将html和JavaScript文件分开编写，使用flask处理独立JavaScript文件时，html中需使用jinja2模板，如下所示。

```
<script src=“{{ url_for(‘static’, filename=‘js/cv.js’) }}” type=“text/javascript”></script>
```

（三）fetch api和flask处理

主流浏览器基本均支持fetch api，且不需要导入类似jQuery、Axios等第三方库，特别适合在内部网络使用[4]。通过该api传送的json字符串可以使用flask中request.json对象获取[5]。下面是前端fetch api示例。

```
fetch(“http://192.168.1.1:5000/test”,{
  method:‘POST’,
  headers:{
    ‘content-type’:‘application/json’
  },
  body: JSON.stringify(Objs)
}).then(response => { ... })
```

前后端处理逻辑为前端提交监控信息，后端接收分析后进行串流测试，因此测试间隔时间是由前端控制的。在前端设置定时器，定时器超时后向后端发送数据，开启测试，直至用户停止循环。定时器的关键代码如下。

```
var intervalId;
document.getElementBy(‘startBtn’).onclick=function(){intervalId=setInterval(..,1000)};
document.getElementBy(‘stopBtn’).onclick=function(){clearInterval(intervalId)};
```

（四）前端优化

通过onchange方法，用于监控终端类型的变动。当多选框终端类别变化时，根据用户选择的不同的设备，同时一并自动填写的其它内容。如果不同设备默认用户名不一样，选择私有编码器则仅需要填写设备IP地址，用户密码应变为不可用。没有填写IP地址时，不允许提交监控设备列表等。对于下面是监控多选变化的关键代码。

```
document.getElementById(‘videoType’).onchange=function(){...}
```

（五）软件运行结果

软件运行时，先添加需要监控设备的列表，点击开始，等待服务器测试并返回结果。其结果如图3所示。

结语

随着视频系统在各个场景应用的迅猛发展，各种视

图3　软件运行结果

频终端设备不仅数量越来越多，种类也呈上升态势。未来设备状态监测必定是向自动化、智能化方向发展。如果仍然墨守成规，使用人工轮询的方式对图像终端进行状态测试的方法难以为继，准确度也不高。针对各品牌视频终端并存，无法统一进行异常情况监视的问题，本文基于opencv开源计算视觉模型和机器学习软件库，通过浏览器将需要监控图像终端信息发送到服务器，使用多线程的方法对不同类型图像终端进行状态测试，不仅简单易行，而且能释放人力资源，降低劳动强度，提高检查周期次数和准确度。

参考文献

[1]明日科技著.Python OpenCV从入门到精通[M].清华大学出版社.2021年9月.

[2]埃里克著.袁国忠译.Python编程：入门到实践[M].人民邮电出版社有限公司.2020年10月

[3]未来科技著.HTML5+CSS3+JavaScript从入门到精通[M].中国水利水电出版社.2017年8月

[4]尼古拉斯·泽卡斯著.JavaScript高级程序设计[M].人民邮电出版社.2012年3月

[5]黄勇著.Flask Web全栈开发实战[M].清华大学出版社.2022年7月