WisView 视频安卓 SDK 移植

用户手册 V2.0

目录

1. 简介

WisView 视频 SDK 主要实现以下功能：

1) 支持本地扫描，获取视频模块的相关信息。
2) 支持 AP 配置将模块配置到路由器。
3) 支持手机本地观看视频和通过数据流量远程观看视频。
4) 支持拍照、录像、对讲、分辨率切换等功能。
5) 支持对视频模块的各种参数设置。
6) 支持 VR 效果，分屏显示视频。
7) 支持软硬件解码。
8) 支持解码 H264 格式和 MJPEG 格式。

2. 移植说明

2.1 本地扫描移植

Scanner 用于本地扫描获取模块的相关信息，依赖于 wisview.sdk.aar。

使用方法如下：

1. 初始化本地扫描接口。

```
Scanner _scanner = new Scanner(AddDeviceStep3.this); //用于调用 Scanner 相关接口
```

2. 设置本地扫描监听。

```
_scanner.setOnScanOverListener(new Scanner.OnScanOverListener() {
        @Override
        public void onResult(Map<InetAddress, String> data, InetAddress gatewayAddress) {
            //监听扫描完成事件，若发现设备即可获得设备信息
            if (data != null) {
                for (Map.Entry<InetAddress, String> entry : data.entrySet()) {
                    String id = entry.getValue(); //扫描到的设备的 id 号
                    String ip = entry.getKey().getHostAddress(); //扫描到的设备的 ip 地址
                }
            }
        }
    });
```

3. 开始本地扫描。

```
_scanner.scanAll(); //开始扫描
```

2.2 AP 配置移植

AP 配置是一种设备建立热点，手机加入设备发送配置信息将设备配置到路由器的配置方式，依赖于 wisview.sdk.aar 和 ParametersConfig.java。

使用方法如下：

1. 初始化模块参数配置接口

```
ParametersConfig _parametersConfig=new ParametersConfig(); //用于调用 ParametersConfig 相关接口
```

2. 设置参数配置接口监听

```
_parametersConfig.setOnResultListener(new ParametersConfig.OnResultListener() {
    @Override
    public void onResult(ParametersConfig.Response result) {
        if(result.statusCode == 200) {
            if (result.type == ParametersConfig.GET_SSID_LIST) {
                //获取设备获取到的网络列表
            }
            else if (result.type == ParametersConfig.JOIN_WIFI) {
                //AP 配置成功
            }
        }
    }
});
```

3. _parametersConfig.getSsidList(); //获取设备获取到的网络列表

4. _parametersConfig.joinWifi(_ssid, _password); //配置设备加入路由器，传入路由器的名称和密码

2.3 视频播放移植

视频播放部分主要是将音视频数据流解码显示的过程，依赖于 wisview.sdk.aar。

使用方法如下：

1.设置对接模块相关参数

```
Module _module = new Module(this);
_module.setLogLevel(Enums.LogLevel.VERBOSE);//设置log打印方式
_module.setUsername("admin");//设置用户名
_module.setPassword("admin");//设置密码
_module.setPlayerPort(554);//设置播放视频的目标端口，本地播放端口为554，远程时为映射端口
_module.setModuleIp(_moduleIp);//设置目标 IP，本地播放 ip 为设备的 ip，远程时为“127.0.0.1”
```

2.设置播放视频相关参数

```
Player _player = _module.getPlayer();
_player.setTimeout(10000);//设置超时时间，单位：ms
_player.setRecordFrameRate(10);//设置录制视频的帧率
_player.setAudioOutput(false);//设置开启或关闭声音
_player.setDisplayView(context, _displayView, _displayView2, _viewType);//设置播放画布解码方式
```

参数描述：

(1) _displayView、_displayView2//显示视频的画布

(2) _viewType//为0：SurfaceView(软解)　1：TextureView0：(软解)　2：SurfaceView(硬解)

使用说明：

*(1) 若使用单屏显示时，将*_displayView、_displayView2其中一个设置为null即可。如：

```
_player.setDisplayView(context, null, _displayView2, 0);
_player.setDisplayView(context, _displayView, null, 0);
_player.setDisplayView(context, null, _displayView2, 1);
_player.setDisplayView(context, _displayView, null, 1);
_player.setDisplayView(context, null, _displayView2, 2);
_player.setDisplayView(context, _displayView, null, 2);
```

(2) 若使用TextureView，则可以通过下面方法获取该TextureView，再做相应的变换。

```
//获取_displayView对应的TextureView
TextureView _textureView=_displayView.getGLTextureView();
if(_textureView!=null){
    _textureView.setRotation(45.0f);//顺时针旋转45°
```

```
        }
    //获取_displayView2对应的TextureView2
        TextureView _textureView2=_displayView2.getGLTextureView2();
        if(_textureView2!=null){
            _textureView2.setRotation(-45.0f);//逆时针旋转45°
        }
```

PS：*SurfaceView的工作方式是创建一个置于应用窗口之后的新窗口，这种方式的效率非常高，因为SurfaceView窗口刷新的时候不需要重绘应用程序的窗口，但是SurfaceView也有一些非常不便的限制。因为SurfaceView的内容不在应用窗口上，所以不能使用变换（平移、缩放、旋转等），也难以放在ListView或者ScrollView中，不能使用UI控件的一些特性比如设置透明度View.setAlpha()。为了解决这个问题 Android 4.0中引入了TextureView。*

SurfaceView(硬解)具有更优的解码效率，需Android 4.1.2以上才支持。

```
_player.getState();//获取视频播放状态
```

状态描述：

(1) Enums.State.IDLE//空闲状态

(2) Enums.State.PLAYING//正在播放

(3) Enums.State.PREPARING//准备播放

(4) Enums.State.STOPPED//已停止播放

```
_player.play(_pipe, Enums.Transport.UDP);//通过UDP获取视频并播放，远程播放时不可用
_player.play(_pipe, Enums.Transport.TCP);//通过TCP获取视频并播放，远程播放时可用
_player.stop();//停止播放视频

boolean _recording = _player.isRecording();//判断是否正在录制视频
_player.beginRecord(String path, String name);//采用mp4v2录制，不占内存，只可录制h264格式
_player.beginRecord0(String path, String name);//采用ffmpeg录制，占内存，可录制h264和mjpeg
_player.endRecord();//结束录制

Bitmap photo = _player.takePhoto();//拍照
_player.setOnVideoSizeChangedListener();//监听播放视频时尺寸变化
_player.setOnStateChangedListener();//监听播放视频时状态变化
_player.setOnRecordStateChangedListener();//监听录制视频时状态变化
```

_player.setOnTimeoutListener();//监听播放视频超时

3.设置默认视频分辨率

修改视频分辨率有两种方式：

1）对于有两路视频的模块，通过选择哪路视频来切换分辨率。

_player.changePipe(_pipe);//设置手机获取视频的分辨率

_pipe 参数描述：

(1) _pipe =Enums.Pipe.H264_PRIMARY//设置手机获取第一路 H264 视频，高清

(2) _pipe =Enums.Pipe.H264_SECONDARY//设置手机获取第二路 H264 视频，标清

(3) _pipe =Enums.Pipe.MJPEG_PRIMARY//设置手机获取第一路 MJPEG 视频，高清

(4) _pipe =Enums.Pipe.MJPEG_SECONDARY//设置手机获取第二路 MJPEG 视频，标清

2）对于只有一路视频的模块，通过参数设置接口设置分辨率。

_parametersConfig.setResolution(int type, int resolution);//设置视频模块的分辨率

resolution参数描述：

0--QVGA(320X240)

1--VGA(640X480)

2--720P(1280X720)

3--1080P(1920X1080)

4.设置播放视频画布

DisplayView _displayView;//设置播放视频画布

_displayView = (DisplayView)findViewById(R.id.*sview*);

_displayView.setFullScreen(true);//设置视频充满画布

布局文件如下：

<com.demo.sdk.DisplayView

android:id="@+id/video_view"

android:layout_width="fill_parent"

android:layout_height="fill_parent"

/>

5.如需对视频做处理，可以通过获取视频 YUV 原始数据，再进行处理

_player.startGetYUVData(true);//使能获取视频 YUV 数据

_player.setOnGetYUVDataListener(new Player.OnGetYUVDataListener() {

@Override

```
        public void onResult(int width, int height, byte[] yData, byte[] uData, byte[] vData) {
            //监听获取视频 YUV 数据
        }
    });
```

2.4 视频参数配置移植

视频参数配置部分主要是获取和配置视频相关参数，依赖于 ParametersConfig.java。

配置接口	功能描述	传入参数	返回值	
updateUsernameAndPassword	更新模块用户名和密码	用户名	成功	{"value": "0"}
		密码	失败	其他
getUsernameAndPassword	获取模块用户名和密码	无	模块的用户名和密码	
getSsidList	获取无线网络列表	无	无线网络列表	
joinWifi	配置模块连接路由器	路由器名称	成功	{"value": "0"}
		路由器密码	失败	其他
getVersion	获取模块版本号	无	模块版本号	
setResolution	设置模块分辨率	类型： 0：本地视频 1：远程视频	成功	{"value": "0"}
		分辨率： 0：320X240 1：640X480 2：1280X720 3：1920X1080	失败	其他
getResolution	获取模块分辨率	类型： 0：本地视频 1：远程视频	320X240	{"value": "0"}
			640X480	{"value": "1"}
			1280X720	{"value": "2"}
			1920X1080	{"value": "3"}
setFps	设置模块帧率	类型： 0：本地视频 1：远程视频	成功	{"value": "0"}
		帧率(1~30)	失败	其他
getFps	获取模块帧率	类型： 0：本地视频 1：远程视频	模块帧率	
setQuality	设置视频质量	类型： 0：本地视频 1：远程视频	成功	{"value": "0"}
		质量（0~139）	失败	其他

getQuality	获取视频质量	类型： 0：本地视频 1：远程视频	视频质量	
setGOP	设置模块的 GOP	gop（0~100）	成功	{"value": "0"}
			失败	其他
getGOP	获取模块的 GOP	无	模块的 GOP	
startSdRecord	开始 SD 卡录像	类型： 0：本地视频 1：远程视频	成功	{"value": "0"}
			繁忙	{"value": "-4"}
			空间不够	{"value": "-22"}
			失败	其他
stopSdRecord	停止 SD 卡录像	类型： 0：本地视频 1：远程视频	成功	{"value": "0"}
			失败	其他
getSdRecordStatus	获取 SD 卡录像状态	类型： 0：本地视频 1：远程视频	空闲	{"value": "0"}
			繁忙	{"value": "1"}
setModuleRtcTime	设置模块 RTC 时间	日期、时、分秒、时区	成功	{"value": "0"}
			失败	其他
getVideoFolderList	获取 SD 卡录像视频文件夹列表	无	SD 卡视频文件夹列表	
getVideoList	获取 SD 卡录像视频文件夹下视频列表	SD 卡视频文件夹路径	SD 卡视频文件列表	
getSignal	获取视频模块信号值	无	模块连接的路由器名称	
			模块的信号值	

NOTE：

由于模块种类和接口种类太多，以上只是部分常用的模块配置接口，所以这部分我们完全开放源码，您可以根据自己的应用需要添加你们想要获取和配置的视频参数。

有任何疑问可以联系：steven.tang@rakwireless.com

2.5 远程 nabto 移植

远程 nabto 部分用于远程通道打通，实现远程播放视频，依赖于 libnabto_client_api_jni.so、com.nabto.api 包中所有文件和 assets 下 nabto 资源文件。

1. RemoteTunnel _remoteTunnel=new RemoteTunnel(getApplicationContext());

2. _remoteTunnel.openTunnel(0,getApplicationContext(), 5555, 554, _deviceId);//5555:映射视频播放端口号; 554:视频默认端口 ; _deviceId:设备id

3. _remoteTunnel.setOnResultListener(new OnResultListener() //监听

```
{
    @Override
    public void onResult(int id, String result)
    {
        // TODO Auto-generated method stub
        if(result.equals("CONNECT_TIMEOUT")||     //远程连接超时
        result.equals("NTCS_CLOSED")||            //远程设备不在线
        result.equals("NTCS_UNKNOWN")||           //远程连接出现未知错误
        result.equals("FAILED"))                  //远程连接失败
        {
        }
        else
        {
            //远程连接成功，映射后的IP为“127.0.0.1”，端口号为“5555”
        }
});
```

4. _remoteTunnel.openTunnel(0,getApplicationContext(), 3333 80, _deviceId);//3333:映射控制端口号; 80:控制默认端口 ; _deviceId:设备id

5._remoteTunnel.closeTunnels();//关闭远程连接

6. 注意：

本地时：目标 ip 为模块的 ip，视频播放端口为 554，控制端口为 80。

远程时：目标 ip 为“127.0.0.1”，视频播放端口为远程连接时对 554 映射后的端口，控制端口为远程连接时对 80 映射后的端口。

2.6 语音对讲移植

语音对讲部分实现模块对讲功能，依赖于 SendAudio.java。

1. 采集 PCM 格式声音数据。

这部分使用安卓自带的的语音采集接口即可。

2. 初始化语音对讲接口。

SendAudio _sendAudio=new SendAudio();

3. 将 PCM 格式的声音数据转换为 PCMU 格式。

byte[] PCMU_Data=_sendAudio.PCMToPCMU(PCM_Data, PCM_Data_Len);

参数说明：

PCM_Data ： PCM 语音数据内容

PCM_Data_Len： PCM 语音数据长度

返回值：

PCMU 声音数据

4. 发送对讲声音数据。

_sendAudio.sendAudio(_deviceIp, _voicePort, buf, len);

参数说明：

_deviceIp ： 模块的 IP 地址

_voicePort： 模块的语音对讲端口

buf： PCMU 语音数据内容

len： PCMU 语音数据长度

5. 关闭语音对讲。

_sendAudio.closeSocket();

注意：

本地时：_ip 为模块的 ip，_voicePort 为 80。

远程时：_ip 为“127.0.0.1”，_voicePort 为远程连接时对 80 映射后的端口。

2.7 视频回放移植

视频回放实现下载播放视频模块录制到 SD 卡中的视频文件。

1.获取 TF 卡中视频文件夹列表

```
ParametersConfig _parametersConfig=new ParametersConfig (_ip+":"+_controlPort,_psk);
_parametersConfig.setOnResultListener(new ParametersConfig .OnResultListener() {
    @Override
    public void onResult(ParametersConfig .Response result) {
        if(result.statusCode==200){
            if(result.type==ParametersConfig .GET_VIDEO_FOLDER_LIST){
                //TF 卡中视频文件夹列表
            }
        }
    }
});
_parametersConfig.Get_Video_Folder_List();
```

2.获取其中一个文件夹中的视频列表

```
ParametersConfig _parametersConfig=new ParametersConfig (_ip+":"+_controlPort,_psk);
_parametersConfig.setOnResultListener(new ParametersConfig .OnResultListener() {
    @Override
    public void onResult(ParametersConfig .Response result) {
        if(result.statusCode==200){
            if(result.type==ParametersConfig .GET_VIDEO_LIST){
                    //TF 卡中其中一个文件夹中的视频列表
                }
            }
        }
    }
});
_parametersConfig.Get_Video_List(path);//path 为 TF 卡中任意文件夹的路径
```

3.根据获取到的视频文件夹和视频路径，播放视频

Mp4Download.playMp4File(url, _psk, savePath, videoHandler);

参数描述：

url：回放视频的路径 ,例如：http://admin:admin@192.168.100.1/link//mnt/rec_folder/video/pipe0/1970Y01M04D15H/NVTDV19700104_150156.mp4

savePath：将会放的视频保存到手机指定的路径。

videoHandler：视频回放时的相关状态返回。

注意：

_psk 为模块密码，默认是 admin。

本地时：_ip 为模块的 ip，controlPort 为 80。

远程时：_ip 为“127.0.0.1”，controlPort 为远程连接时对 80 映射后的端口。

2.8 透传移植

透传部分主要实现手机与模块实时通信的功能。

些模块透传是通过建立 TCP 连接，目标端口号为 80；有些模块是通过建立 UDP 连接，目标端口号为 1008，具体见对应产品的规格书等文档。

1.TCP 透传

（1）创建 TCP 连接

```
Socket _socket = new Socket(_deviceIp, _sendPort);
_socket.setKeepAlive(true);
dataStream =new DataOutputStream(_socket.getOutputStream());
```

（2）TCP 发送数据

```
dataStream.write(message);
```

（3）TCP 接收数据

```
_socket.getInputStream().read(buffer);
```

（4）关闭 TCP 连接

```
_socket.close();
dataStream.close();
```

2.UDP 透传

（1）创建 UDP 连接

```
DatagramSocket udp_socket=new DatagramSocket(25000);
```

（2）UDP 发送数据

```
InetAddress serverAddress = InetAddress.getByName(_deviceIp);
DatagramPacket sendPackage = new DatagramPacket(data , data.length , serverAddress
, _sendPort);
udp_socket.send(sendPackage);
```

（3）UDP 接收数据

```
DatagramPacket recvPackage = new DatagramPacket(buffer , buffer.length);
udp_socket.receive(recvPackage);
```

（4）关闭 UDP 连接

```
udp_socket.close();
```

3.注意：

发送数据均以 0x01 0x55 开头，接收到的数据模块内部会自动添加 0x01 0x55。即：

发送数据时：0x01 0x55 要发送的数据内容

接收数据时：0x01 0x55 要接收的数据内容

本地时：_deviceIp 为模块的 ip，_sendPort 为 80。

远程时：_deviceIp 为“127.0.0.1”，_sendPort 为远程连接时对 80 映射后的端口。

3. 相关权限

WisView SDK需要用到的权限如下：

```
<uses-permission android:name="android.permission.CHANGE_WIFI_MULTICAST_STATE"></uses-permission>
<uses-permission android:name="android.permission.INTERNET"></uses-permission>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"></uses-permission>
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE"></uses-permission>
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE"></uses-permission>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"></uses-permission>
<uses-permission android:name="android.permission.WAKE_LOCK"></uses-permission>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"></uses-permission>
<uses-permission android:name="android.permission.CALL_PHONE"></uses-permission>
<uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS"></uses-permission>
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.ACCESS_LOCATION"></uses-permission>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"></uses-permission>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"></uses-permission>
<uses-permission android:name="android.permission.DISABLE_KEYGUARD"></uses-permission>
<uses-permission android:name="android.permission.RESTART_PACKAGES" />
<uses-permission android:name="android.permission.KILL_BACKGROUND_PROCESSES" />
<uses-permission android:name="android.permission.CHANGE_CONFIGURATION" />
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS" />
<uses-permission android:name="android.permission.GET_TASKS" />
<uses-permission android:name="android.permission.BROADCAST_STICKY" />
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
```

4. 修改记录

版本	作者	时间	修改内容
V1.0	瞿瑾	2016/03/05	创建文档
V1.1	瞿瑾	2016/05/17	1. 更新设置视频画布接口，可以单屏或双屏显示视频，并可以选择 SurfaceView 还是 TextureView 显示。 2. 添加获取视频 YUV 数据接口。
V1.2	瞿瑾	2016/08/05	1. 添加硬件解码。
V1.3	瞿瑾	2016/12/01	1.优化本地扫描。 2.保留 ffmpeg 和 mp4v2 两种录制方式。 3.添加图片处理功能。 4.规避播放 AAC 声音的模块闪退的问题。 5.添加视频回放功能。 6.添加透传功能。
V1.4	瞿瑾	2017/02/24	1.优化视频录制。
V2.0	瞿瑾	2017/04/11	1.整理并开放模块参数配置接口。 2.整理并开放对讲接口。 3.整理 SDK。