WisView 视频 SDK 移植说明 V2.0

1. 简介

1.1 概述

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 libWisView.a, Scanner.h。

使用方法如下：

1. 初始化本地扫描接口。

```
#import "Scanner.h" //引用头文件
Scanner *_scanner =    [[Scanner alloc] init]; //用于调用  Scanner  相关接口
```

2. 设置本地扫描监听。

```
- (void)scanDeviceOver:(Scanner *)result;
{
    if (result.Device_ID_Arr.count > 0) {
    //监听扫描完成事件，若发现设备即可获得设备信息
        [result.Device_ID_Arr enumerateObjectsUsingBlock:^(id obj, NSUInteger idx, BOOL *stop) {
            NSString *deviceIp = [result.Device_IP_Arr objectAtIndex:idx]; //扫描到的设备的 id 号
            NSString *deviceId = [result.Device_ID_Arr objectAtIndex:idx]; //扫描到的设备的 ip 地址
        }
    }
}
```

3. 开始本地扫描。

```
Scanner *result = [_device_Scan ScanDeviceWithTime:2.0f];   //开始扫描，扫描时间 2.0s
[self performSelectorOnMainThread:@selector(scanDeviceOver:) withObject:result waitUntilDone:NO];
```

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 libWisView.a 和 ParametersConfig.h。

使用方法如下：

1. 初始化模块参数配置接口

```
#import "Scanner.h" //引用头文件
<ParametersConfigDelegate> //添加代理，用于接收配置返回数据
//_ip：模块的 ip 地址 _psk：模块密码 初始化用于调用 ParametersConfig 相关接口
ParametersConfig *_parametersConfigStep2AP=[[ParametersConfig alloc]init:self ip:_ip password:_psk];
```

2. 设置参数配置接口监听

```
- (void)setOnResultListener:(int)statusCode :(NSString*)body :(int)type{
    if (type==JOIN_WIFI) {
        if(statusCode==200){ //AP 配置成功
        }
    }
    else if (type==GET_SSID_LIST) {
        if(statusCode==200){ //获取设备获取到的网络列表
        }
    }
}
```

3. [_parametersConfigStep2AP getSsidList]; //获取设备获取到的网络列表

4. [_parametersConfigStep2AP joinWifi:_ssid :_psk]; //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 libWisView.a, WisView.h。

主要接口如下：

1. 初始化视频播放界面

(1)单屏播放

```
_videoView = [[WisView alloc] initWithFrame:CGRectMake(0, 0,w,h)];
[_videoView setView1Frame:frame]; //设置视频播放 view 的 frame
[_videoView set_log_level:4];//设置 log 打印方式
[_videoView delegate:self];//设置代理
[self.view addSubview:_videoView];//添加视频显示的 view
```

(2)双屏播放

```
_videoView = [[WisView alloc] initWithFrame2:frame:frame1:frame2];
// frame1：_videoView 上第一个视频的 frame；frame2： _videoView 上第二个视频的 frame
[_videoView setView1Frame:frame1]; //设置视频播放 view1的 frame1
[_videoView setView2Frame:frame2]; //设置视频播放 view2的 frame2
[_videoView set_log_level:4];//设置 log 打印方式
[_videoView delegate:self];//设置代理
[self.view addSubview:_videoView];//添加视频显示的 view
```

相关接口：

```
- (void)setView1Frame:(CGRect)frame;//设置第一个视频的 frame
- (void)setView2Frame:(CGRect)frame;//设置第二个视频的 frame
- (CGRect)getView1Frame;//获取第一个视频的 frame
- (CGRect)getView2Frame;//获取第二个视频的 frame
- (void)setView1Hidden:(BOOL)isHidden;//设置第一个视频是隐藏
- (void)setView2Hidden:(BOOL)isHidden;//设置第二个视频是隐藏
```

2. 设置视频播放参数

```
NSString *url ;
url = [NSString stringWithFormat:@"rtsp://admin:%@@%@:%d/cam1/%@",_psk,_ip,_port,_pipe];
```

_psk：设备连接密码

_ip：播放视频目标 IP，本地播放 ip 为设备的 ip，远程时为“127.0.0.1”

_port：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

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。

_pipe 参数描述：

(1) _pipe =@"h264"//设置手机获取第一路 H264 视频，高清

(2) _pipe =@"h264-1"//设置手机获取第二路 H264 视频，标清

(3) _pipe =@"mpeg4"//设置手机获取第一路 MJPEG 视频，高清

(4) _pipe =@"mpeg4-1"//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: type :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)

[_videoView sound:NO];//设置开启或关闭声音

[_videoView set_record_frame_rate:10];//设置录制视频的帧率

[_videoView play:url useTcp:NO];//通过 UDP 或 TCP 获取视频并播放

[_videoView stop];//停止播放视频

3. 拍照与录像

[_videoView take_photo];//拍照

[_videoView begin_record:type];//开始录制,type: 0 ffmpeg 录制 1 mp4v2 录制

[_videoView begin_record2:type :path];//开始录制视频到指定的路径，path 为路径

[_videoView end_record];//结束录制

4. 监听视频播放状态

```
- (void)state_changed:(int)state{
    switch (state) {
        case 0: //空闲状态{
            break;
        }
        case 1: //准备播放{
            break;
        }
        case 2: ////正在播放{
            break;
        }
        case 3: //已停止播放{
            break;
        }
        default:
            break;
    }
}
- (void)video_info:(NSString *)codecName codecLongName:(NSString *)codecLongName{
    //监听播放的视频信息
}
- (void)audio_info:(NSString *)codecName codecLongName:(NSString *)codecLongName
                    sampleRate:(int)sampleRate channels:(int)channels{
    //监听播放的音频信息
}
```

5. 获取视频解码后的 YUV 数据

```
- (void)startGetYUVData:(BOOL)start;//使能获取视频解码后的 YUV 数据
- (void)GetYUVData:(int)width :(int)height
                    :(Byte*)yData :(Byte*)uData :(Byte*)vData
                    :(int)ySize :(int)uSize :(int)vSize;//监听视频解码后的 YUV 数据
{//获取视频解码后的 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.m, ParametersConfig.h。

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 common 包和 3rdParty 包中 nabto 包。

1. NabtoLibraryInit();//初始化nabto

2. Async_ConnectDeviceWithTunnel(&videoTunnel ,deviceId,554 ,5555);//videoTunnel：视频通道，5555:映射视频播放端口号; 554:视频默认端口 ; _deviceId:设备id

3. Async_ConnectDeviceWithTunnel(&httpTunnel ,deviceId,554 ,3333);//httpTunnel：控制(透传) 通道，3333:映射控制端口号; 80:控制默认端口 ; _deviceId:设备id

4.int status = CheckConnectStatus(&videoTunnel);//status 为 0 表示远程连接成功，映射后的 IP 为“127.0.0.1”，端口号为“5555”

int status = CheckConnectStatus(&httpTunnel);//status 为 0 表示远程连接成功，映射后的 IP 为“127.0.0.1”，端口号为“3333”

5. CloseTunnel(&videoTunnel);//关闭视频通道

CloseTunnel(&httpTunnel);//关闭控制(透传)通道

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 AudioRecord.m，AudioRecord.h，sendAudio.m，sendAudio.h。

1. 采集 PCM 格式声音数据。

这部分使用 IOS 自带的的语音采集接口即可。

2. 初始化语音对讲接口。

```
AudioRecord* audioRecord = [[AudioRecord alloc]init];
```

3. 录制 PCMU 格式声音数据。

```
[audioRecord StartRecord]; //开始录制 PCMU 语音数据

NSData* PCMUData = [audioRecord StopRecord]; //停止录制 PCMU 语音数据
```

4. 发送对讲声音数据。

```
[sendAudio sendWithIp: _deviceIp port: _voicePort data:PCMUData];
```

参数说明：

_deviceIp ： 模块的 IP 地址

_voicePort： 模块的语音对讲端口

PCMUData： PCMU 语音数据内容

注意：

本地时：_deviceIp 为模块的 ip，_voicePort 为 80。

远程时：_deviceIp 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 TF 卡中的视频文件。

使用方法如下：

1. 初始化模块参数配置接口

```
#import "Scanner.h" //引用头文件
<ParametersConfigDelegate> //添加代理，用于接收配置返回数据
//_ip：模块的 ip 地址 _psk：模块密码  初始化用于调用 ParametersConfig 相关接口
ParametersConfig *_parametersConfig=[[ParametersConfig alloc]init:self ip:_ip password:_psk];
```

2. 设置参数配置接口监听

```
- (void)setOnResultListener:(int)statusCode :(NSString*)body :(int)type{
    if (type==GET_VIDEO_FOLDER_LIST) {
        if(statusCode==200){ //获取 TF 卡中视频文件夹列表
        }
    }
    else if (type==GET_VIDEO_LIST) {
        if(statusCode==200){ //TF 卡中其中一个文件夹中的视频列表
        }
    }
}
```

3. [_parametersConfig getVideoFolderList]; //获取 TF 卡中视频文件夹列表

4. [_parametersConfig getVideoList:folder]; //TF 卡中其中一个文件夹中的视频列表

5. 根据获取到的视频文件夹和视频路径，播放视频

```
MPMoviePlayerController *_moviePlayer=[[MPMoviePlayerController alloc]initWithContentURL:url];
_moviePlayer.view.frame=self.view.bounds;
_moviePlayer.view.autoresizingMask=UIViewAutoresizingFlexibleWidth|UIViewAutoresizingFlexibleHeight;
[self.view addSubview:_moviePlayer.view];
```

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

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。

有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

```
GCDUartSocket = [[GCDAsyncSocket alloc] initWithDelegate:self
delegateQueue:dispatch_get_main_queue()];
[GCDUartSocket connectToHost:_deviceIp onPort:_sendPort error:nil];
```

（2）TCP 发送数据

```
[GCDUartSocket writeData:data withTimeout:1.0 tag:100];
```

（3）TCP 接收数据

```
[GCDUartSocket readDataWithTimeout:-1 tag:0];
-(void)socket:(GCDAsyncSocket *)sock didReadData:(NSData *)data withTag:(long)tag{
    if([sock isEqual:GCDUartSocket]){
        //接收到的数据
        [GCDUartSocket readDataWithTimeout:-1 tag:0];
    }
}
```

（4）关闭 TCP 连接

```
if (GCDUartSocket != nil) {
    [GCDUartSocket disconnect];
    GCDUartSocket = nil;
}
```

2.UDP 透传

（1）创建 UDP 连接

```
GCDUdpSocket = [[GCDAsyncUdpSocket alloc] initWithDelegate:self
delegateQueue:dispatch_get_main_queue()];
[GCDUdpSocket bindToPort :25000 error:nil];
```

（2）UDP 发送数据

```
[GCDUdpSocket sendData:data toHost:_deviceIp port:_sendPort withTimeout:1.0 tag:100];
```

（3）UDP 接收数据

```
[GCDUdpSocket beginReceiving:&err];
- (void)udpSocket:(GCDAsyncUdpSocket *)sock didReceiveData:(NSData *)data
    fromAddress:(NSData *)address withFilterContext:(id)filterContext{
    if([sock isEqual:GCDUdpSocket]){
            //接收到的数据
    }
}
```

（4）关闭 UDP 连接

```
if (GCDUdpSocket != nil) {
    [GCDUdpSocket close];
    GCDUdpSocket = nil;
}
```

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 Frameworks

WisView SDK需要用到的Frameworks：

CoreGraphics.framework

AVFoundation.framework

CoreVideo.framework

Foundation.framework

UIKit.framework

CFNetwork.framework

SystemConfiguration.framework

OpenAL.framework

AssetsLibrary.framework

libbz2.tbd

libbz.tbd

libiconv.tbd

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	2016/07/07	1.添加分屏显示。 2.添加录像到指定路径。 3.添加获取解码后的 YUV 数据。
V1.2	瞿瑾	2016/12/02	1.保留 ffmpeg 和 mp4v2 两种录制方式。 2.规避播放 5275 闪退的问题。 3.添加视频回放功能。 4.添加透传功能。
V1.3	瞿瑾	2017/02/24	1.优化视频录制。
V2.0	瞿瑾	2017/04/17	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。