版本修改记录

版本	责任人	日期	备注
V1.0	**YangBing**	**20181115**	新建
V1.1	**YangBing**	**20181116**	根据建议修改
V1.2	**YangBing**	**20181119**	修改排版及内容
V1.3	**YangBing**	**20190114**	增加客户端事件回调说明
V1.4	**LiuBo**	**20200206**	更新预览图像实例代码

目录

红外网络 SDK 快速开发指南

简介

本文档旨在使开发者能快速接入我司的 SDK，并用 SDK 预览设备图像、测温这两个最常用的功能。

编译环境

该 SDK 基于，

平台：windows 7 64 位专业版 。

编译环境：Visual Studio 2015 专业版 带 update3

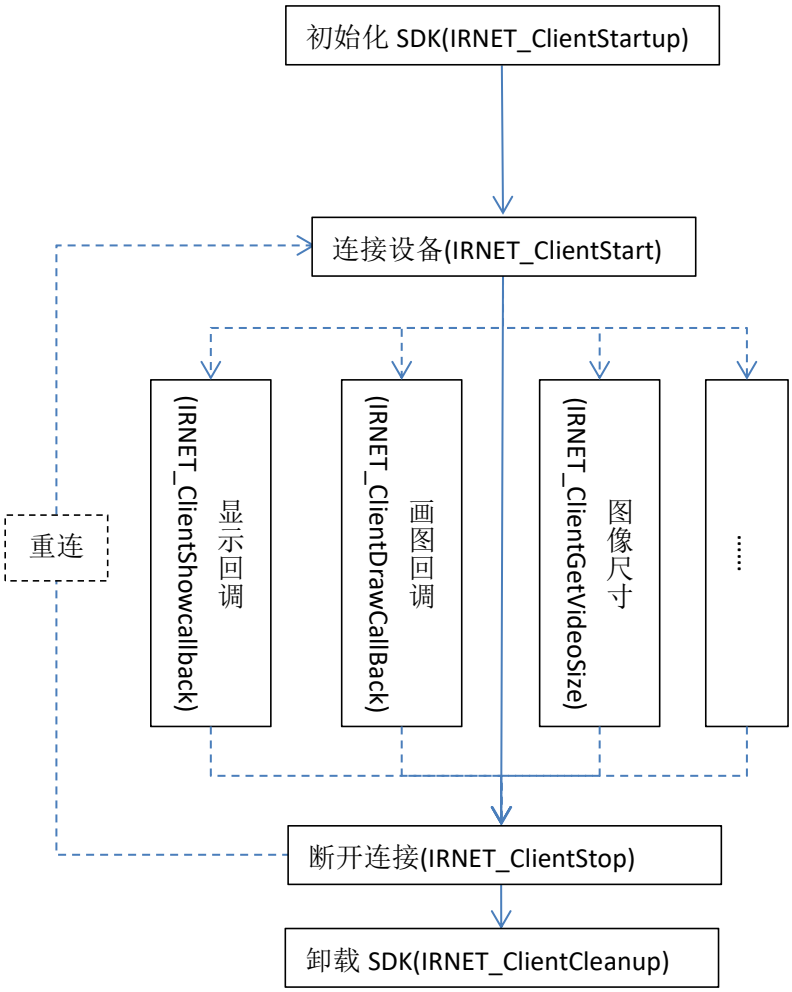
编程指导

注：SDK 内部采用多线程通信方式，所有异常通知都通过回调函数反馈给调用者。不要在回调函数中调用释放资源的接口，避免引起内部死锁或者异常。

预览图像

通过该流程，可达到预览设备图像的效果。

流程示意



- ✧ 初始化 SDK(IRNET_ClientStartup)和卸载 SDK(IRNET_ClientCleanup)整个程序的运行周期(程序额启动到退出)内分别只需要调用一次即可。
- ✧ 建议 IRNET_ClientStartup 的消息回调函数(void (WINAPI*messagecallback)(IRNETHANDLE

handle, WPARAM wparam, LPARAM lparam, void* context)参数传入有效参数，当 SDK 发生一些事件时，会调用该回调接口，比如当设备连接成功(LAUMSG_LINKMSG == wparam &&0 ==lparam)或者连接失败(LAUMSG_LINKMSG == wparam &&0 !=lparam)都会调用该回调。

- ✧ 装填通道信息结构(CHANNEL_CLIENTINFO)后，作为参数传给接口 IRNET_ClientStart 即可获得通道句柄。
- ✧ 可以用 IRNET_ClientStart 返回的句柄调用其他接口，如显示回调(IRNET_ClientShowcallback)、画图回调(IRNET_ClientDrawCallBack)、图像尺寸(IRNET_ClientGetVideoSize)等接口设置或获取特别在意的信息。
- ✧ 连接设备接口(IRNET_ClientStart)被成功调用后一定要调用断开连接接口(IRNET_ClientStop)去释放连接设备接口申请的资源。
- ✧ 如果需要重连，请先调用断开连接接口(IRNET_ClientStop)释放上次的连接。如果需要获得重连事件，请关注初始化接口 IRNET_ClientStartup 的参数 m_messagecallback，该参数为回调函数入口地址，设备断开后，发生的断开事件会通过回调反馈。注：不要在回调函数中调用释放资源的接口。

代码示例

```
#include <windows.h>
#include <stdio.h>
#include "IRNet.h"

void nomalvideo(char *pbuff, int headsize, int datasize, int timetick, int biskeyframe,
void *context)
{
    //.h264 nalu is start from pbuff
    //printf("recv video data headsize:%d datasize:%d timetick:%d iskeframe:%d\n",
headsize, datasize, timetick, biskeyframe);
}

void WINAPI  messagecallback(IRNETHANDLE hHandle,WPARAM wParam,LPARAM  lParam,void *context)
{
    switch (wParam)
    {
    case LAUMSG_LINKMSG:
        if (!lParam)//connection successful
        {
            if (lParam == 0)
            {
                //test .h264 data
                if (-1 != m_hchann)
                {
                    if (!IRNET_ClientStartNomalCap(m_hchann, nomalvideo, NULL,
NULL, NULL))
                    {
                        printf("StartNomalCap start failed!\n");
                    }
                    else
                    {
                        printf("StartNomalCap start success!\n");
                    }
                }
                else printf("connect invalid\n");
            }
        }
        else//connect failed
        {
        }
        break;
```

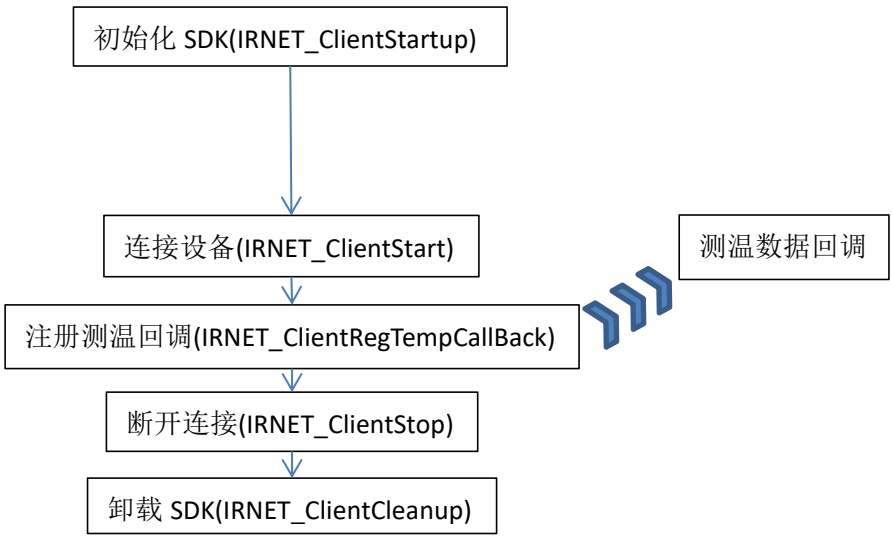
```
    default:break;
    }
}

int main()
{
    int errcode = 0;

    if (!IRNET_ClientStartup(0, NULL, messagecallback, NULL))
    {
        //出错
        errcode = -2;
        goto err;
    }
    bool reconn_flag = false;
    CHANNEL_CLIENTINFO clientInfo;
    IRNETHANDLE m_handleClientStart;
reconnect:
    ZeroMemory(&clientInfo, sizeof(clientInfo));
    clientInfo.m_hChMsgWnd = NULL;
    clientInfo.m_nChmsgid = 0;
    clientInfo.m_sername = "video server";
    clientInfo.m_username = "888888";
    clientInfo.m_password = "888888";
    clientInfo.m_playstart = TRUE;//解码
    clientInfo.m_tranType = 3;//3—tcp方式通信
    clientInfo.m_useoverlay = FALSE;
    clientInfo.nColorKey = RGB(255, 0, 0);
    clientInfo.m_ch = 0;//0通道
    clientInfo.m_buffnum = 20;//缓存帧数目
    clientInfo.m_hVideohWnd = NULL;// m_ctrlPIC.GetSafeHwnd();//视频渲染窗口
    clientInfo.context = NULL;
    clientInfo.m_messagecallback = NULL;

    if ((IRNETHANDLE)(-1) == (m_handleClientStart =
        IRNET_ClientStart("192.168.1.29",
        &clientInfo,
        3000,
        0//码流号 和通道号共同作用，唯一确定设备视频流的编码格式
        )))
    {
        //出错
        errcode = -3;
        goto err;
```

```
    }

    getchar();//输入任意字符按键 然后停止预览、程序退出

    if (!IRNET_ClientStop(m_handleClientStart))
    {
        //停止预览失败
        errcode = -4;
        goto err;
    }
    if (reconn_flag)
    {
        //继续
        goto reconnect;
    }
err:
    if (!IRNET_ClientCleanup())
    {
        errcode = -4;
    }
    return errcode;
}
```

测温

用红外网络 SDK 可实现测温功能，该测温的效果是返回每帧图像的所有像素点的温度值，即为一个温度矩阵。

流程示意



- 调用 SDK 的接口之前需要对 SDK 进行初始化(IRNET_ClientStartup)。初始化接口在整个程序的运行周期(程序的启动到退出)内只需要调用一次。
- SDK 初始化(IRNET_ClientStartup)时可以传入一个消息回调函数(void (WINAPI*messagecallback)(IRNETHANDLE handle, WPARAM wparam, LPARAM lparam, void* context)，SDK 内部发生的事情会通过这个回调函数反馈给调用者。比如当设备连接成功(LAUMSG_LINKMSG == wparam &&0 ==lparam)或者连接失败(LAUMSG_LINKMSG == wparam &&0 !=lparam)都会调用该回调。
- 程序退出之前要卸载 SDK(IRNET_ClientCleanup)。卸载 SDK 在整个程序的运行周期内也只需要调用一次。
- 可以看成 IRNET_ClientStartup 初始化 SDK 资源，IRNET_ClientCleanup 释放 SDK 资源。
- SDK 测温依赖设备的 raw 码流。码流由 IRNET_ClientStart 的参数 m_pChaninof 中的 m_ch 成员(控制通道号)和参数 streamtype(控制码流号)共同控制，raw 码流在 m10 设备中配置为：通道号 0，码流号 2。在 PCB 设备中配置为：通道号 1，码流号 1。
- 连接设备接口(IRNET_ClientStart)被成功调用后一定要调用断开连接接口(IRNET_ClientStop)去释放连接设备接口申请的资源。
- 注：在回调函数中不要调用释放资源相关的接口，应该在回调函数中发送一个异步处理消息，在消息响应函数中释放资源，避免引起回调函数的异常或者 SDK 内部资源的死锁问题。

代码示例

```
#include <windows.h>
#include <stdio.h>
#include "IRNet.h"
//测温回调函数入口
void CALLBACK Temprature(float fTemperature[], UINT uWidth, UINT uHeight, void*context)
{
    //fTemperature可以看做温度数组
    //数组长度为uWidth×uHeight的float数组
    char tm[100] = { 0 };
    sprintf(tm, "%.1f℃[%u,%u]", fTemperature[0], uWidth, uHeight);
}

void WINAPI  messagecallback(IRNETHANDLE hHandle,WPARAM wParam,LPARAM  lParam,void *context)
{
     switch (wParam)
     {
     case LAUMSG_LINKMSG:
          if (!lParam)//connection successful
          {
          }
          else//connect failed
          {
          }
          break;
      default:break;
     }
}
int main()
{
    int errcode = 0;
    IRNETHANDLE m_handleClientStart;
    if (!IRNET_ClientStartup(0, NULL, messagecallback, NULL))
    {
        //出错
        errcode = -2;
        goto err;
    }

    CHANNEL_CLIENTINFO clientInfo;
    ZeroMemory(&clientInfo, sizeof(clientInfo));
    clientInfo.m_hChMsgWnd = NULL;
```

```
    clientInfo.m_nChmsgid = 0;
    clientInfo.m_sername = "video server";
    clientInfo.m_username = "888888";
    clientInfo.m_password = "888888";
    clientInfo.m_playstart = TRUE;//解码
    clientInfo.m_tranType = 3;//3—tcp方式通信
    clientInfo.m_useoverlay = FALSE;
    clientInfo.nColorKey = RGB(255, 0, 0);
    clientInfo.m_ch = 0;//0通道
    clientInfo.m_buffnum = 20;//缓存帧数目
    clientInfo.m_hVideohWnd = NULL;//m_ctrlPIC.GetSafeHwnd();//视频渲染窗口
    clientInfo.context = NULL;
    clientInfo.m_messagecallback = NULL;

    if ((IRNETHANDLE)(-1) == (m_handleClientStart =
        IRNET_ClientStart("192.168.1.29",
        &clientInfo,
        3000,
        2//码流号 和通道号共同作用，唯一确定设备视频流的编码格式
        )))
    {
        //出错
        errcode = -3;
        goto err;
    }

    if (!IRNET_ClientRegTempCallBack(m_handleClientStart, Temprature, NULL))
    {
        //测温回调注册失败
        errcode = -4;
        goto err;
    }
    printf("Press any key to exit");
    getchar();//输入任意字符按键 然后停止预览、程序退出
err:
    if ((IRNETHANDLE)(-1) != m_handleClientStart && !IRNET_ClientStop(m_handleClientStart))
    {
        //停止预览失败
        errcode = -5;
    }
    if (!IRNET_ClientCleanup())
    {
        errcode = -6;
    }
```

```
    return errcode;
}
```

API 接口

通道信息结构

作为 IRNET_ClientStart 的参数使用，保存设备的通道信息及一些基本的配置信息。

CHANNEL_CLIENTINFO

```
/** 客户端通道信息结构*/
typedef struct{
    char *m_sername;                    ///<设备的名称
    char *m_username;                   ///<访问设备的账号
    char *m_password;                   ///<访问设备的密码
    WORD m_tranType;                    ///<传输类型(数据传输类型，多播、
                                        ///<UDP、TCP等；一般传入3-TCP方式)
    WORD m_playstart;                   ///<是否开始预览(分配解码资源)
    BYTE m_ch;                          ///<设备通道号
    HWND m_hVideohWnd;                  ///<视频渲染窗口的句柄(可填NULL)
    HWND m_hChMsgWnd;                   ///<消息窗口句柄(回调消息发送到该窗
口，可填NULL)
    UINT m_nChmsgid;                    ///<消息号(可填0)
    int  m_buffnum;                     ///<帧缓存的数量(10~50)，越大画面越
流畅，但延迟响应的会增加，越小画面越不流畅，但延迟较小，一般填20
    int  m_useoverlay;                  ///<暂时填FALSE
    COLORREF nColorKey;                 ///<暂时无效，可随意填RGB值
    char url[40];                       ///<可不填。char url[40]={0}
    CCICALLBACK m_messagecallback;      ///<消息回调函数，异步消息回调调用
    void *context;                      ///<回调函数回调的自定义数据
}CHANNEL_CLIENTINFO;
```

初始化 SDK

初始化 SDK 的环境，分配 SDK 所需的资源，整个程序运行周期内只需调用一次。

IRNET_ClientStartup

```
/**
 * @brief 初始化客户端SDK
 * @param[in] m_nMessage        应用程序的一个用户自定义消息
 * @param[in] m_hWnd            应用程序中一个窗口句柄
 * @param[in] m_messagecallback 消息回调函数接口
 * @param[in] context           用户上下文
 * @param[in] key               解密字符串，默认为空
 * @return TRUE表示成功，FALSE表示失败
 * @note 服务连接断开或者连接成功等消息可以通过回调函数m_messagecallback异步通知
 * @code
 //回调函数参数说明
 hHandle [IN] 连接句柄，IRNET_ClientStart返回值
 wParam  [IN] 参数1
 lParam  [IN] 参数2
 context [IN] 用户上下文
 * @endcode
*/
BOOL CALLINGCONVEN IRNET_ClientStartup(UINT m_nMessage,HWND m_hWnd,
void (WINAPI *m_messagecallback)(IRNETHANDLE hHandle,WPARAM wParam,LPARAM lParam,void *context) = NULL,
void *context = NULL, char *key = NULL);
```

连接设备

连接设备。该接口成功返回后必须用 IRNET_ClientStop 接口释放内部相关连接资源。

IRNET_ClientStart

```
/**
 * @brief 与服务器建立连接，并实时预览图像
 * @param[in] m_url        服务器的地址或转发服务器的地址
 * @param[in] m_pChaninfo CHANNEL_CLIENTINFO的指针
 * @param[in] wserport     服务器或转发服务器的端口号
 * @param[in] streamtype   连接的码流类型,0:主码流 1:次码流,此项需要设备支持，默认为主码流
 * @return -1表示失败，其他值为连接句柄
*/
 IRNETHANDLE CALLINGCONVEN IRNET_ClientStart(char *m_url,CHANNEL_CLIENTINFO *m_pChaninfo,
WORD wserport = 3000,int streamtype = 0);
```

断开连接

断开与设备的连接，释放相关连接资源。

IRNET_ClientStop

```
/**
 * @brief 停止播放，断开服务器服务器的连接
 * @param hHandle [IN] 连接句柄，IRNET_ClientStart的返回值
 * @return TRUE表示成功，FALSE表示失败
*/
BOOL  CALLINGCONVEN IRNET_ClientStop(IRNETHANDLE hHandle);
```

卸载 SDK

仅释放 SDK 资源，并不会释放连接相关资源，只需程序退出时调用一次。

IRNET_ClientCleanup

```
/**
 * @brief 卸载客户端SDK
 * @param 无
 * @return TRUE表示成功，FALSE表示失败
*/
BOOL  CALLINGCONVEN IRNET_ClientCleanup();
```

测温回调

注册一个回调函数，当相关事件(需要设备支持 raw 码流)发生时调用回调函数，且因为回调函数在每帧图像都会被调用，因此回调函数内部不应做耗时操作，避免影响画面流畅度。

IRNET_ClientRegTempCallBack

```
/**@brief 测温回调函数指针
* @param[in] fTemperature   温度数据地址。大小为图像的宽×高,单位℃
* @param[in] uWidth         图像的宽
* @param[in] uHeight        图像的高
* @param[in] context        与IRNET_ClientRegTempCallBack的context是同一份，此处只是做回传
* @return 无
*/
typedef void(CALLBACK*TEMPCALLBACK)(float fTemperature[], UINT uWidth, UINT uHeight, void*context);
```

```
/**
 * @brief 注册测温回调
 * @param[in] hHandle   连接句柄，IRNET_ClientStart的返回值
 * @param[in] pCallBack 回调函数地址
 * @param[in] context   传NULL或者传入结构体DEV_TEMP_SPAN的地址(调用者分配内存)
 * @return TRUE表示成功，FALSE表示失败
*/
BOOL CALLINGCONVEN IRNET_ClientRegTempCallBack(IRNETHANDLE hHandle, TEMPCALLBACK pCallBack, void*context);
```

显示回调

将要显示的 yuv 数据回调给用户，方便用户对 yuv 数据做处理。

IRNET_ClientShowcallback

```
/**
 * @brief 设置显示的回调函数
 * @param[in] hHandle      IRNET_ClientStart的返回值
 * @param[in] ShowCallBack 回调函数
 * @param[in] context      用户上下文
 * @return TRUE表示成功，FALSE表示失败
 * @par 回调函数参数说明:
 * @code
 m_y      Y数据起始地址
 m_u      U数据起始地址
 m_v      V数据起始地址
 stridey  数据Y的跨度
 strideuv 数据U、V数据的跨度
 width    图像数据的宽度
 height   图像数据的高度
 context  用户传入的上下文
 * @endcode
*/
BOOL CALLINGCONVEN IRNET_ClientShowcallback(IRNETHANDLE hHandle,
void(WINAPI *ShowCallBack)(BYTE *m_y,BYTE *m_u,BYTE *m_v,int stridey,int strideuv,int width,int height,void *context),
void *context);
```

画图回调

在渲染出设备的图像后，回传一个绘图环境给用户，方便用户在设备预览画面之上再灵活的做一些用户想做的工作。

IRNET_ClientDrawCallBack

```
/**
 * @brief 设置画图回调函数
 * @param[in] hHandle      IRNET_ClientStart的返回值
 * @param[in] DrawCallBack 回调函数
 * @param[in] context      用户上下文
 * @return TRUE表示成功，FALSE表示失败
 * @par 回调函数参数说明:
 * @code
 hdc     图元文件
 context 用户传入的上下文
 * @endcode
*/
BOOL CALLINGCONVEN IRNET_ClientDrawCallBack(IRNETHANDLE hHandle,
void(WINAPI *DrawCallBack)(HDC hDC,void *context),
void *context);
```

图像尺寸

获取设备上传的图像的分辨率

IRNET_ClientGetVideoSize

```
/**
 * @brief 获取图像尺寸
 * @param[in]  hHandle   IRNET_ClientStart的返回值
 * @param[out] m_pWidth  返回的宽度
 * @param[out] m_pHeight 返回的高度
 * @return TRUE表示成功，FALSE表示失败
*/
BOOL CALLINGCONVEN IRNET_ClientGetVideoSize(IRNETHANDLE hHandle, DWORD *m_pWidth, DWORD *m_pHeight);
```