红外热像仪软件 SDK 开发包使用说明

目 录

历史版本：

版本	时间	说明
V2.2.5.0	2022/6/1	1、更新自动获取 IP 的方式，不需要另外创建定时器（以前的方式也可以使用）
V2.2.5.1	2022/6/23	1、使程序通过防火墙，程序运行时不需要关闭防火墙 2、修改了 IRSDK_Create 的返回值：创建连接时，判断本地 IP 与相机 IP 是否在同一网段，通过返回值区分

一、 头文件说明

```
#define DEVICE_MAX                  (32)        //最多支持例化32个设备的实例句柄
#define OBJ_MAX                     (32)        //最多支持例化 32 个对象，上层只能少于 32，不能超过

typedef int(*CBF_IR)(void* lData, void* lParam);        //SDK 中回调函数的格式声明（后面有详细说明）

#define CALTEMP(x,y)        ((x-10000)/(float)y)        //Y16 数据与温度数据的转换关系(具体说明见下节)

//文件操作 （后续存储/读取文件时需要用到）
#define OPEN_FILE                   (1)
#define CLOSE_FILE                  (2)
#define WR_FRAME                    (3)

//聚焦参数（只对有电动聚焦功能的设备有效）
#define   STOPFOCUS                 (0)
#define   FARFOCUS                  (1)
#define   NEARFOCUS                 (2)
#define   AUTOFOCUS                 (3)
#define   FOCUSSTATUS               (4)

//帧格式
    typedef struct tagFrame
    {
      unsigned short width;             //图像宽度
      unsigned short height;            //图像高度
      unsigned short u16FpaTemp;        // unused，焦面温度
      unsigned short u16EnvTemp;        // unused,环境温度
      unsigned char   u8TempDiv;        //数据转换成温度时需要除以该数据，文档具体说明
      unsigned char   u8DeviceType;     //unused
      unsigned char   u8SensorType;     //unused
      unsigned char   u8MeasureSel;     //unused
      unsigned char   u8Lens;           //unused
      unsigned char   u8Fps;            //unused
      unsigned char   u8TriggerFrame;   //当有外触发功能时，表示该帧是否是触发帧
      unsigned char   Reversed[17];     //unused
      unsigned short buffer[327680];    //图像数据，按行存储，最大支持 640x512，每个像素是一个 ushort 类型
    } Frame;
//128byte，存储原始视频时，该头信息会保存到视频文件中，读取时，会读出该头信息以供使用
typedef struct tagSAVEHead
{
      unsigned char   Head[32];
      unsigned short width;
      unsigned short height;
```

```
        unsigned int     totalFrames;  //返回视频总帧数
        unsigned short Freq;           //返回文件帧频
        unsigned char    Reserved0;
        unsigned char    Reserved1;
        unsigned int     timelen;              //返回文件时长
        unsigned char    timestamp[32];    //返回时间戳
        unsigned char    Reserved[48];
    } T_SAVE_HEAD;

    typedef struct tagIPADDR
    {
        char IPAddr[32];                //IP,字符串
        unsigned char Reserved[32]; //保留
        unsigned int DataPort;          //Port
        unsigned char isValid;          //是否在线
        unsigned char totalOnline;   //在线个数
        unsigned char Index;            //在列表中的索引
    }T_IPADDR;
//告警
    typedef struct t_alarm
    {
        unsigned char isDraw;                   //是否需要绘制告警的屏显
        unsigned char alarmType;                //告警类型选择（OverHigh, UnderLow, BetweenHL, DeBetween）
        unsigned char reserved2;
        unsigned char reserved3;
        float         HighThresh;               //高门限温度
        float         LowThresh;                //低门限温度
        T_COLOR       colorAlarm;               //告警颜色
    }T_ALARM;
    //矩形以及矩形内温度统计，其它对象类似
    typedef struct stat_rect
    {
        T_RECT sRect;
        STAT_TEMPER sTemp;
        unsigned int   LableEx[32];         //对象名
        unsigned char Lable[32];            //对象名
        float         inputEmiss;           //输入辐射率，一定要给默认值（默认0.98）
        float         inputReflect;         //输入反射温度，一定要给默认值（默认20）
        float         inputDis;             //输入距离，一定要给默认值（默认2）
        float         Area;                 //计算面积
        unsigned char reserved1;
        unsigned char reserved2;
        unsigned char reserved3;
        unsigned char reserved4;
```

```
    T_COLOR     color;              //对象颜色
    T_ALARM     sAlarm;             //告警
}STAT_RECT;

//所有对象，该结构体用来存放所有测温对象，num表示该类型对象的个数，对应的指针，需要指向测温对象，该空间需要在使用前开辟
typedef struct stat_obj
{
    unsigned char numPt;            //点的个数
    unsigned char numLine;          //线的个数
    unsigned char numCircle;        //圆的个数
    unsigned char numRect;          //矩形的个数
    unsigned char numPolygon;       //多边形的个数
    unsigned char Reserved1;        //保留
    unsigned char Reserved2;        //保留
    unsigned char Reserved3;        //保留
    T_ALARM         sGlobalAlarm;       //全局告警信息
    STAT_POINT      sPt[OBJ_MAX];       //点的对象从0索引依次往后面存
    STAT_LINE       sLine[OBJ_MAX];     //线的对象从0索引依次往后面存
    STAT_CIRCLE     sCircle[OBJ_MAX];   //圆的对象从0索引依次往后面存
    STAT_RECT       sRect[OBJ_MAX];     //矩形的对象从0索引依次往后面存
    STAT_POLYGON sPolygon[OBJ_MAX];     //多边形的对象从0索引依次往后面存
    STAT_RADAR      Reserved[1];
}STAT_OBJ;

//参数配置类型
enum T_PARAMTYPE
{
    paramDevice             = 0,        //设备类型
    paramDownSample     = 1,            //降采样
    paramReserved0      = 2,            //保留
    paramReserved1      = 3,            //保留
    paramReserved2      = 4,            //保留
    paramSpaceFilter        = 5,        //空域滤波
    paramReserved4      = 6,            //保留
    paramTempSegSel     = 7,            //温度段选择
    paramReserved6      = 8,            //保留
    paramReserved7      = 9,            //保留
    paramReserved8      = 10,           //保留
    paramCaliSwitch     = 11,           //快门校正开关
    paramTempCorrect    = 12,           //校正温度
    paramReserved11     = 13,           //保留
    paramReserved12     = 14,           //保留
    paramReserved13     = 15,           //保留
```

```
        paramAutoSelTempSelSw    = 16,      //自动选择温度段开关
        paramObjTempFilterSw     = 17,      //测温对象温度滤波开关
        paramImageFlip           = 18,      //图像镜像
    };
```

其它类型及函数说明详见头文件注释

```
#ifndef _T_CTRLPROTOCOL
#define _T_CTRLPROTOCOL
//运动控制类型
enum T_PARAMCTRL
{
    //协议选择
    paramPelcod          = 0,          //pelco-d
    paramUserDef1        = 1,          //自定义协议1（升降杆）
    paramUserDef2        = 2,          //自定义协议2（舵机）
    paramUserDef3        = 3,          //自定义协议3（转台）

    //控制
    paramCtrlUp          = 4,            //上
    paramCtrlDown        = 5,            //下
    paramCtrlLeft        = 6,            //左
    paramCtrlRight       = 7,            //右
    paramCtrlStop        = 8,            //停止
    paramCtrlBaudRate    = 9,            //波特率
};
#endif
```

二、 函数说明

1、设备操作函数

(1) 获取 sdk 版本信息

```
IR_SDK_API int IRSDK_GetVersion(char* version);
```

Function : IRSDK_GetVersion

Description : 获取 sdk 版本信息,格式：v2.2.3.0。

Input : char* version：接收版本的字符串指针，调用前开辟空间，空间大于 16Byte

Return : -1：异常

0: 正常

(2) 初始化 SDK 操作

```
IR_SDK_API int IRSDK_Init(void);
```

Function : IRSDK_Init

Description : 运行软件时，调用初始化函数，才能接收到相机的 IP 信息，只需要调用一次，不要反复调用。

Input : void

Return : 0 失败
1 初始化成功;

(3) 退出 SDK 操作

IR_SDK_API int IRSDK_Quit (void);

Function : IRSDK_Quit

Description : 退出软件时，调用该函数，SDK 释放资源，只需要调用一次，不要反复调用。

Input : void

Return : 0 成功;
-1 失败;

(4) 设置 IP 地址接收空间

IR_SDK_API int IRSDK_SetIPAddrArray(void * pIpInfo);

Function : IRSDK_SetIPAddrArray

Description : 自动获取在线相机的 IP 信息，定义一个 T_IPADDR ipaddr[DEVICE_MAX]数组，将数组地址作为参数传入，数组内容即 IP 地址，数组内容由 SDK 维护。

Input :

void * pIpInfo: T_IPADDR ipaddr[DEVICE_MAX]数组的地址，该数组在调用前需要定义，数组内容中 SDK 维护，SDK 实时更新在线的设备 IP，例如，如果连接了一个设备 192.168.1.10，则该数组第一个元素为该设备的 IP 信息等，其它元素为 0；如果连接了两台设备 192.168.1.10 和 192.168.1.11，则第一个元素为设备 192.168.1.10 的 IP 信息，第二个元素为 192.168.1.11 的 IP 信息，当设备掉线时，该数组也会删除对应的元素，并保证在线的设备 IP 信息顺序放在前面的位置。，

Return : 0 成功;
-1 失败

(5) 创建红外视频连接

IR_SDK_API int IRSDK_Create(int handle, T_IPADDR ipInfo, CBF_IR cbf_stm, CBF_IR cbf_cmd, CBF_IR cbf_comm, void* param = 0);

Function : IRSDK_Create

Description : 创建红外连接，创建后即绑定了红外的视频、通讯、命令端口。

Input :

int handle : 红外句柄， 0- (DEVICE_COUNT-1) 范围内整数。

T_IPADDR inInfo: 查询到的设备 IP 信息。

CBF_IR cbf_stm : 可选参数。 回调函数指针， 用于上传红外视频帧。

CBF_IR cbf_cmd : 可选参数。一般设置为 NULL，回调函数指针，用于与设备命令交互。

CBF_IR cbf_comm : 可选参数，一般设置为 NULL，回调函数指针，用于与通信端口交互。

Void * param : 可选参数。 回调函数的私有数据，在回调调用时传给回调函数。

Return : 1: 创建成功 (没有检测到同网段 IP，可以提示用户修改本机的 IP，linux 版本不检测 IP);
0: 创建成功;
-1: 创建失败，无效句柄;
-2: 创建失败，IP 无效。

(6) 销毁红外视频连接

IR_SDK_API int IRSDK_Destroy(int handle);

Function : IRSDK_Destroy

Description : 销毁红外连接，销毁后即将接收不到对应句柄的红外数据，如果需要再次接收，需要重新创建 IRSDK_Create 和连接 IRSDK_Connect

Input :

int handle : 红外句柄， 0- (DEVICE_COUNT-1)范围内整数。

Return : -1 句柄不存在。

0 销毁成功

(7) 连接红外

```
IR_SDK_API int IRSDK_Connect(int handle);
```

Function : IRSDK_ Connect

Description : 发送连接命令给设备，如果连接成功，回调将会开始调用

Input :

int handle : 红外句柄， 0- (DEVICE_COUNT-1)范围内整数。

Return : -1 句柄不存在。

0 发送成功

(8) 播放

```
IR_SDK_API int IRSDK_Play(int handle);
```

Function : IRSDK_ Play

Description : 在设备 Stop 状态下，发送播放命令给设备，发送该命令可以开始数据传输

Input :

int handle : 红外句柄， 0- (DEVICE_COUNT-1)范围内整数。

Return : -1 句柄不存在。

0 发送成功

(9) 停止

```
IR_SDK_API int IRSDK_Stop(int handle);
```

Function : IRSDK_ Stop

Description : 发送停止命令给设备，设备接收后停止发送图像数据，回调函数不会被执行

Input :

int handle : 红外句柄， 0- (DEVICE_COUNT-1)范围内整数。

Return : -1 句柄不存在。

0 发送成功

(10) 修改 IP

```
IR_SDK_API int IRSDK_SetIP(int handle, char * pIp);
```

Function : IRSDK_SetIP

Description : 发送命令修改设备 IP（只有当设备连接成功后才能修改）

Input :

int handle : 红外句柄， 0- (DEVICE_COUNT-1)范围内整数。

char* pIp：设备 IP，字符串类型指针，如需要修改成 192.168.8.57，则 char* pIp = “192.168.8.57”；

Return : -1 句柄不存在。

0 发送成功

注意：如果修改完后相机 IP 与电脑 IP 不在同一局域网段，需要修改电脑 IP，才能正常连接

(11) 发送命令

IR_SDK_API int IRSDK_Command(int handle, int command, int param);

Function : IRSDK_ Command

Description : 发送命令给设备(该函数用于扩展，一般情况不需要使用)

Input :

int handle : 红外句柄， 0- (DEVICE_COUNT-1)范围内整数。

Int command : 命令；

Int param : 命令参数；

Return : -1 句柄不存在。

0 发送成功

(12) 快门校准

IR_SDK_API int IRSDK_Calibration(int handle);

Function : IRSDK_ Calibration

Description : 发送快门校准命令给设备，设备接收后会进行快门校准动作，一般情况下设备会自动校准，不需要用户操作

Input :

int handle ： 红外句柄， 0- (DEVICE_COUNT-1)范围内整数。

Return : -1 句柄不存在。

0 发送成功

(13) 通信端口发送数据

IR_SDK_API int IRSDK_CommSend(int handle,char *pBuf, int len);

Function : IRSDK_ CommSend

Description : 该命令用于透传 RS485 数据，用户需要时可以使用

Input :

int handle : 红外句柄， 0- (DEVICE_COUNT-1)范围内整数。

char * pBuff : 数据；

Int len : 数据长度；

Return : -1 句柄不存在。

0 发送成功

(14) 获取 IP 地址

（老版本支持，v2.2.5.0 之后的版本不要使用该函数，建议使用 IRSDK_SetIPAddrArray 接收 IP 地址）

IR_SDK_API int IRSDK_InquireIP(T_IPADDR* pIpInfo, int TimerInterval);

Function : IRSDK_ InquireIP

Description : 监控设备状态

Input :

int TimerInterval：监控周期（该函数放在定时器中，参数为触发周期，单位 ms，周期最大 1000ms）

T_IPADDR *：该指针指向一个 T_IPADDR 类型的 IP 列表数组首，数组大小为 DEVICE_COUNT，该数组在调用前需要定义，T_IPADDR IpInfo[DEVICE_COUNT]; SDK 实时更新连接的设备 IP，例如，如果连接了一个设备 192.168.1.10，则该数组第一个元素为该设备的 IP 信息等，其它元素为 0；如果连接了两台设备

192.168.1.10 和 192.168.1.11，则第一个元素为设备 192.168.1.10 的 IP 信息，第二个元素为 192.168.1.11 的 IP 信息，当设备掉线时，该数组也会删除对应的元素，并保证在线的设备 IP 信息顺序放在前面的位置。

Return : 0

注：IRSDK_InquireIP 查询函数必须放在定时器，或者放在线程中间隔 500ms 左右执行一次，该函数第二个参数为间隔的时间，单位 ms

(15) 在线状态

```
IR_SDK_API int IRSDK_IsConnected(int handle);
```

Function : IRSDK_ IsConnected

Description : 上位机发送 IRSDK_ Connect 命令后，通过该函数返回相机状态标志地址，可根据该值判断是否连接正常，如果该值为 0，表示相机连接失败，上位机可以设置一定时间未连接成功就超时失败或者再次发送 IRSDK_Connec 命令。建议两次 IRSDK_Connect 命令间隔不能小于 1s。

Input :

Int handle：句柄

Return :

1：连接成功。

0：未连接成功。

(16) 聚焦函数

```
IR_SDK_API int IRSDK_NearFarFocus(int handle, unsigned int param);
```

Function : IRSDK_ NearFarFocus

Description : 该函数用于控制聚焦电机，近焦远焦时，鼠标按下时发送 IRSDK_NearFarFocus（0，FARFOCUS）或者 IRSDK_NearFarFocus（0，NEARFOCUS），鼠标弹起时发送送 IRSDK_NearFarFocus（0，STOPFOCUS），自动聚焦时，鼠标按下发送 IRSDK_NearFarFocus（0，AUTOFOCUS），弹起不需要操作

Input :

int handle ：红外句柄， 0- (DEVICE_COUNT-1)范围内整数；

unsigned int param :根据需要传入头文件定义的动作：

```
//聚焦参数
#define   STOPFOCUS                (0)
#define   FARFOCUS                 (1)
#define   NEARFOCUS                (2)
#define   AUTOFOCUS                (3)
#define   FOCUSSTATUS               (4)
```

Return :

int : -1 句柄不存在

0 发送成功

（注： 当参数为 FOCUSSTATUS 时，返回 1 表示正在自动聚焦，返回 0 表示未开始或者已完成自动聚焦）

(17) 参数配置

```
IR_SDK_API int IRSDK_ParamCfg(int handle, T_PARAMTYPE mParamType, float f32Param);
```

Function : IRSDK_ParamCfg

Description ：配置相机参数

Description ：该函数用于配置设备参数，paramDevice 需要在发送连接命令前配置，其它参数需要连接成功后配置

Input :

int handle ：红外句柄， 0- (DEVICE_COUNT-1)范围内整数;

T_PARAMTYPE mParamType： 参数类型 （头文件中枚举类型）

float f32Param：配置参数

Return :

int : -1 句柄不存在

0 发送成功

参数说明:

① paramDevice：当设备为人体测温类型时，该参数配置为 1，其它类型设备不需要，该参数用于配置图像是否进行旋转，人体测温类型设备需要高>宽

② paramDownSample: 该参数针对网络带宽不够情况，可以对相机帧频进行降频(sdk 已经实现了自动降频处理，一般情况不需要手动调用)，若要手动降频，则需要进行配置(该参数需要在确定连接正确后发送才有效)。例如：最大帧频为 50Hz，参数配置为 1—50Hz， 2 — 25Hz， 4—12.5Hz，…… ， 32—1.5Hz，该参数最大为 32。(每次发送连接红外命令后，默认开启自动降频，如果想关闭自动降频，每次连接成功后，需要手动配置帧频)

③ paramSpaceFilter：该参数用于降噪，一般情况下该参数不需要调整。(调整范围为 0~100，0 为关闭算法)，参数越大，去噪效果越好，但是图像相对会变模糊。

④ paramTempSegSel: 该参数用于选择温度段（只对有多段温度曲线的设备有效），0 对就常温段，1 对应中温段，2 对应高温段。

⑤ paramCaliSwitch: 该参数用于自动快门的开和关（1 开，0 关，开机默认开启自动快门校正）。

⑥ paramTempCorrect: 该参数用于直接对全屏的温度进行校正（正负表示增加或者减小，精确到 0.01℃）。

⑦ paramAutoSelTempSelSw：该参数用于自动温度段切换功能开关（0：手动切换温度段； 1：自动切换温度段，自动切换需要固件支持）。

⑧ paramObjTempFilterSw：该参数用于开关获取测温对象的温度滤波功能（优化温度波动较大的情况，根据实际测温对象及测温要求等情况配置）（0：关闭滤波功能； 1 开启空域滤波；2 开启时域滤波；3 开启空域时域滤波； 默认为 0）。

⑨ paramImageFlip：该参数用于图像镜像（0：原始图像； 1：左右镜像； 2：上下镜像； 3：对角镜像）。

(18) 设备信息查询

IR_SDK_API int IRSDK_InquireDeviceInfo(int handle, T_DEVICE_INFO* pDevInfo);

Function : IRSDK_ InquireDeviceInfo

Description ：该函数用于查询设备信息

Input :

int handle ：红外句柄， 0- (DEVICE_COUNT-1)范围内整数;

T_DEVICE_INFO* pDevInfo ：传入地址，用于接收设备信息。该空间不需要在上层开辟，只需要传一个接收指针;

Return : 0：查询完成

2、 图像视频相关函数

调光函数几点说明:

①下面 4 个函数均为调光函数，实现从原始数据到灰度数据的转换，手动调光函数功能全，并且通过参数配置，可以实现与自动调光一样的效果;

②调光函数 1 与调光函数 2 相比，少了 DDE 和伽马校正功能，调光函数 2 通过配置 DDE 和伽马校正参数全部为 0 时，与调光函数 1 功能完全相同;

③每个调光函数都对应一个支持多线程的函数，建议使用多线程函数，使用多线程函数，需要额外开辟 1K 的空间传入函数，作为各线程中的函数内部变量存储的空间;

④区间调光通过参数配置可以完全替换自动调光函数，由于需要兼容，所以保留了以前的函数;

⑤调光函数 2 通过参数配置可以完全替换调光函数 1，由于需要兼容，所以保留了以前的函数;

⑥DDE 和时域滤波算法比较耗时，当全部开启时，384x288 设备可以到 50Hz，640x480 设备可以到 35Hz 左右。用户根据需要选择是否开启。

⑦由于需要兼容，输出的灰度图像 pGray 是 unsigned short 类型，实际灰度值范围为 0~255。

⑧用户也可以自己实现调光，替换 sdk 中的调光函数

(19) 自动调光函数 1

IR_SDK_API int IRSDK_FrameConvert(Frame *pFrame, unsigned short *pGray, float f32Constrast, float f32Bright, unsigned int *pGethist, STAT_TEMPER *pFull_temper，unsigned short u16TFilterCoef);

Function : IRSDK_FrameConvert

Description ：将接收到的温度数据进行调光，转成可显示的灰度图像，该函数仅供参考，用户也可以根据自己的特定需求实现调光

Input :

Frame *pFrame ：数据回调函数中接收的帧数据结构指针;

unsigned short *pGray：指向转换后的数据首地址，需要调用前开辟空间;

float f32Constrast：对比度，1~100 可设，默认 50;(对人体测温类型设备，该参数表示显示的最低温)

float f32Bright：亮度，1~100 可设，默认 50;(对人体测温类型设备，该参数表示显示的最高温)

unsigned int *pGethist：获取直方图，该空间需要调用前分配，大小为 65535*4Byte，若不需要，可以设置为 NULL;

STAT_TEMPER *pFull_temper：存储全屏最高最低平均温以及最值坐标，变量在调用前需先定义；若不需要，可以设置为 NULL;(此处获取的温度为测量温度，没有经过辐射率等校正，若需要校正，还需要调用 IRSDK_TempCorrect 进行校正)

unsigned short u16TFilterCoef：时域滤波参数，为 0 时关闭算法，从 1~100 算法效果逐渐减弱，该参数建议设置为 100（注意：1、该算法比较耗时，最大支持 50Hz，对于需要更高帧频的应用，该参数应设置为 0.

2、该算法只适用于实时视频或者回放录像，对于单幅图像的回放和处理，该参数设置为 0，否则可能出现前后两幅图出现重影现象）

Return :

int : 0 完成。

PS: 当时域滤波开启时，该函数不支持多线程调用，如果开启时域滤波，需要多线程调用时，则调用下面的函数：

IR_SDK_API int IRSDK_FrameConvert_m(Frame *pFrame, unsigned short *pGray, float f32Constrast, float f32Bright, unsigned int *pGethist, STAT_TEMPER *pFull_temper，unsigned short u16TFilterCoef, unsigned char *pThreadbuf);

相比上面的函数，多了一个参数：

unsigned char * pThreadbuf: input，传入一个指针，开辟不小于 w*h*4Byte 的空间，指针对应空间在调用函数前开辟，注意在一个线程中该空间地址不能变（该空间主要是为了支持多线程，存储函数内部变量）。

(20) 自动调光函数 2

IR_SDK_API int IRSDK_FrameConvertDDE(Frame *pFrame, unsigned short *pGray, float f32Constrast, float f32Bright, unsigned int *pGethist, STAT_TEMPER *pFull_temper，unsigned short u16TFilterCoef, unsigned char u8DDEcoef, unsigned char u8Gamma);

Function : IRSDK_FrameConvertDDE

Description : 相比 IRSDK_FrameConvert，多了两个参数，具体描述如下

Input :

unsigned char u8DDEcoef：细节增强参数，0~100，0 为关闭，效果同 IRSDK_FrameConvert，1~100 细节增强，同时噪声也会有一定程度增加，建议根据实际场景及需求来进行配置；

unsigned char u8Gamma：伽马校正参数，0~10，0 为关闭， 1~10 表示不同等级伽马校正，建议根据实际场景及需求来进行配置；

其它参数与 IRSDK_FrameConvert 函数完全相同。同样的，也对应一个支持多线程的函数，

PS: 当时域滤波开启时，该函数不支持多线程调用，如果开启时域滤波，需要多线程调用时，则调用下面的函数：

IR_SDK_API int IRSDK_FrameConvertDDE_m(Frame *pFrame, unsigned short *pGray, float f32Constrast, float f32Bright, unsigned int *pGethist, STAT_TEMPER *pFull_temper，unsigned short u16TFilterCoef, unsigned char u8DDEcoef, unsigned char u8Gamma, unsigned char *pThreadbuf);

相比上面的函数，多了一个参数：

unsigned char * pThreadbuf: input，传入一个指针，开辟不小于 w*h*4Byte 的空间，指针对应空间在调用函数前开辟，注意在一个线程中该空间地址不能变（该空间主要是为了支持多线程，存储函数内部变量）。

(21) 区间调光函数 1

IR_SDK_API int IRSDK_FrameAgc(Frame *pFrame, unsigned short *pGray, float f32Constrast, float f32Bright, float f32MinT, float f32MaxT, STAT_RECT *pRect, unsigned char u8Method， unsigned int *pGethist, STAT_TEMPER *pFull_temper，unsigned short u16TFilterCoef);

Function : IRSDK_FrameAgc

Description : 将接收到的温度数据进行调光，转成可显示的灰度图像，该函数仅供参考，用户也可以根据自己的特定需求实现调光；与 IRSDK_FrameConvert 函数相比，该函数支持设定调光范围。

Input :

float f32MinT：调光范围最低温阈值；

float f32MaxT：调光范围最高温阈值（当 f32MinT 与 f32MaxT 相等时，该函数为自动调光）；

STAT_RECT *pRect：根据提供的矩形区域的最低最高温设定调光范围，当使用这个时，f32MinT 和 f32MaxT 设置无效，不需要使用时设置为 NULL

unsigned char u8Method：调光方法，设置为 0；

其它参数与 IRSDK_FrameConvert 相同。

Return :

int : 0 完成。

PS: 当该函数参数 f32MinT 与 f32MaxT 相等，pRect 为 NULL 时，该函数与 IRSDK_FrameConvert 功能相同。

该函数对应一个支持多线程的函数，

IR_SDK_API int IRSDK_FrameAgcDDE_m(Frame *pFrame, unsigned short *pGray, float f32Constrast, float f32Bright, float f32MinT, float f32MaxT, STAT_RECT *pRect, unsigned char u8Method, unsigned int *pGethist, STAT_TEMPER *pFull_temper, unsigned short u16TFilterCoef, unsigned char *pThreadbuf);

相比上面的函数，多了一个参数：

unsigned char * pThreadbuf: input，传入一个指针，开辟不小于 w*h*4Byte 的空间，指针对应空间在调用函数前开辟，注意在一个线程中该空间地址不能变（该空间主要是为了支持多线程，存储函数内部变量）。

(22) 区间调光函数 2

IR_SDK_API int IRSDK_FrameAgcDDE(Frame *pFrame, unsigned short *pGray, float f32Constrast, float f32Bright, float f32MinT, float f32MaxT, STAT_RECT *pRect, unsigned char u8Method, unsigned int *pGethist, STAT_TEMPER *pFull_temper, unsigned short u16TFilterCoef, unsigned char u8DDECoef, unsigned char u8Gamma);

Function : IRSDK_FrameAgcDDE

Description : 与 IRSDK_FrameAgc 相比，增加了两个参数，具体描述如下。

Input :

unsigned char u8DDEcoef：细节增强参数，0~100，0 为关闭，效果同 IRSDK_FrameConvert，1~100 细节增强，同时噪声也会有一定程度增加，建议根据实际场景及需求来进行配置；

unsigned char u8Gamma：伽马校正参数，0~10，0 为关闭， 1~10 表示不同等级伽马校正，建议根据实际场景及需求来进行配置；

其它参数与 IRSDK_FrameAgc 函数完全相同。同样的，也对应一个支持多线程的函数，

IR_SDK_API int IRSDK_FrameAgcDDE_m(Frame *pFrame, unsigned short *pGray, float f32Constrast, float f32Bright, float f32MinT, float f32MaxT, STAT_RECT *pRect, unsigned char u8Method, unsigned int *pGethist, STAT_TEMPER *pFull_temper, unsigned short u16TFilterCoef, unsigned char u8DDEcoef, unsigned char u8Gamma, unsigned char *pThreadbuf);

相比上面的函数，多了一个参数：

unsigned char * pThreadbuf: input，传入一个指针，开辟不小于 w*h*4Byte 的空间，指针对应空间在调用函数前开辟，注意在一个线程中该空间地址不能变（该空间主要是为了支持多线程，存储函数内部变量）。

(23) 灰度转成 RGB

IR_SDK_API int IRSDK_Gray2Rgb(unsigned short* pGray, unsigned char* pRgb, unsigned short Width, unsigned short Height, int PalType int Pal);

unsigned short* pGray : input，输入灰度值

unsigned char* pRgb： output，输出 RGB 值，该空间需要调用前开辟，空间大小为(w * h * 3)Byte

unsigned short Width ： input,图像宽

unsigned short Height：input,图像高

int PalType: 选择调色板类型， 一般设置为 0。0 为一般工业调色板，1 为医用调色板。

int Pal : input，选择对应调色板，当 PalType 为 0 时，该参数范围 0~8，当 PalType 为 1 时，该参数范围 0~6

return: -1：异常

0：正常

(24) 获取调色板 JPEG 图像

IR_SDK_API int IRSDK_GetPaletteJPEG(unsigned char* pPaletteJPG, unsigned int *pJPGLen, unsigned char Method, int PalType, int pal);

unsigned char* pPaletteJPG： output，输出 pal 对应的调色板 JPEG 图像，输出图像为 1x256 分辨率，获取后可以按照需要放大，该空间需要调用前开辟，建议不小于(w * h * 3)Byte。

unsigned int *pJPGLen: output，输出上述调色板 JPEG 图像的文件大小，该变量在调用前定义

unsigned char Method: input，配置为 0

int PalType: 选择调色板类型， 一般设置为 0。0 为一般工业调色板，1 为医用调色板。

int Pal : input，选择对应调色板，当 PalType 为 0 时，该参数范围 0~8，当 PalType 为 1 时，该参数范围 0~6

return: -1：异常

0：正常

(25) 获取调色板 Bmp 图像

IR_SDK_API int IRSDK_GetPaletteBmp (unsigned char* pPaletteBmp, unsigned int * pBmpLen, unsigned char Method, int PalType, int pal);

unsigned char* pPaletteBmp： output，输出 pal 对应的调色板 Bmp 图像，输出图像为 1x256 分辨率，获取后可以按照需要放大，该空间需要调用前开辟，建议不小于(54+4*256)Byte。

unsigned int * pBmpLen: output，输出上述调色板 Bmp 图像的文件大小，该变量在调用前定义

unsigned char Method: input，配置为 0

int PalType: 选择调色板类型，一般设置为 0。0 为一般工业调色板，1 为医用调色板。

int pal : input，选择对应调色板，当 PalType 为 0 时，该参数范围 0~8，当 PalType 为 1 时，该参数范围 0~6

return: -1：异常

0：正常

(26) RGB 转 BMP 图像文件

IR_SDK_API int IRSDK_Rgb2Bmp(unsigned char * pBmpData, unsigned int *pLen, unsigned char* pRgb, unsigned short Width, unsigned short Height);

unsigned char* pBmpData： output，输出转换后对应的 BMP 图像缓存，该空间需要调用前开辟，建议不小于(54+w * h * 3) Byte 空间。

unsigned int *pLen: output，输出上述 BMP 图像的文件大小，该变量在调用前定义
unsigned char* pRgb：input，输入 RGB 值，空间大小为(w * h * 3)Byte
unsigned short Width ： input,图像宽
unsigned short Height：input,图像高
return: -1：异常
0：正常

(27) RGB 转 JPG 图像文件

IR_SDK_API int IRSDK_Rgb2Jpeg(unsigned char * pJpegout, unsigned int *pLen, int quality, unsigned char * pRgb, unsigned short Width, unsigned short Height);

unsigned char* pJpegout： output，输出转换后对应的 Jpg 图像缓存，该空间需要调用前开辟，建议不小于(w*h*3)Byte 空间。
unsigned int *pLen: output，输出上述 Jpg 图像的文件大小，该变量在调用前定义
int quality: input，压缩效率，1~100，越大效果越好，文件也越大，建议配置为 100
unsigned char* pRgb：input，输入 RGB 值，空间大小为(w*h*3)Byte
unsigned short Width ： input,图像宽
unsigned short Height：input,图像高
return: -1：异常
0：正常

(28) 温度数据转 JPEG 图像文件

IR_SDK_API int IRSDK_SaveFrame2Jpeg(char *pFile, Frame *pFrame, unsigned char* pRgb, unsigned char isSaveObj, STAT_OBJ *pObj);

char *pFile：input,用于传入文件的路径和文件名，以字符串形式输入
Frame *pFrame: input，温度数据帧，合并到 JPEG 文件中
unsigned char* pRgb：input，输入 RGB 值，用于生成 JPEG 图像
unsigned char isSaveObj ： input,是否将测温对象存入到 JPEG 图像
STAT_OBJ *pObj：input,测温对象信息
return: -1：异常
0：正常

(29) 读取 JPEG 图像文件解出温度数据

IR_SDK_API int IRSDK_ReadJpeg2Frame(char* pFile, Frame *pFrame, unsigned char isSaveObj, STAT_OBJ *pObj);
char *pFile：input，用于传入文件的路径和文件名，以字符串形式输入
Frame *pFrame: output，解出的温度数据帧，需要在调用前定义并分配空间
unsigned char isSaveObj ： input,是否加载测温对象
STAT_OBJ *pObj：output,加载测温对象信息，为空时不保存对象信息
return: -1：异常
0：正常

(30) 存储原始温度数据文件

IR_SDK_API int IRSDK_SaveFrame2Video(char *pFile, Frame *pFrame, unsigned char Op, unsigned char isSaveObj,

STAT_OBJ *pObj, unsigned char * pThreadbuf);

char *pFile：input,用于保存文件的路径和文件名，以字符串形式输入

Frame * pFrame: input, 需要写入的帧数据

unsigned char Op：写第一帧时，Op = OPEN_FILE

写最后一帧时，Op = CLOSE_FILE

其它情况，Op = WR_FRAME

OPEN_FILE/CLOSE_FILE/WR_FRAME 在头文件中已定义

unsigned char isSaveObj ： input,是否将测温对象存入到 JPEG 图像

STAT_OBJ *pObj：input, 测温对象信息，，为空时不保存对象信息

unsigned char * pThreadbuf: input, 传入一个指针，开辟不小于 1Kbyte 的空间，指针对应空间在调用函数前开辟，注意在一个线程中该空间地址不能变（该空间主要是为了支持多线程，存储函数内部变量）。

return: -1：异常

0：正常

(31) 读取解析原始温度数据录像文件

IR_SDK_API int IRSDK_ReadVideo2Frame(char *pFile, Frame *pFrame, unsigned int Index, unsigned char Op,

T_SAVE_HEAD *pVideoHead, STAT_OBJ *pObj, unsigned char * pThreadbuf);

char *pFile：input,用于传入文件的路径和文件名，以字符串形式输入

Frame *pFrame: output, 温度数据帧，调用前开辟空间

unsigned int Index：input,读取帧索引

unsigned char Op：读第一帧时，Op = OPEN_FILE

读最后一帧时，Op = CLOSE_FILE (文件读完时会自动关闭文件，也可以在文件没读完时主动关闭文件)

其它情况，Op = WR_FRAME

OPEN_FILE/CLOSE_FILE/WR_FRAME 在头文件中已定义

T_SAVE_HEAD *pVideoHead ： output, 读取录像文件头信息，该空间调用前分配，会返回每帧图像的时间戳

STAT_OBJ *pObj：output, 读取测温对象信息，该空间调用前分配

unsigned char * pThreadbuf: input, 传入一个指针，开辟不小于 1Kbyte 的空间，指针对应空间在调用函数前开辟，注意在一个线程中该空间地址不能变（该空间主要是为了支持多线程，存储函数内部变量）。

return: -1：文件异常

-2：格式错误

-3：文件结束（同时将关闭文件，此时不需要手动关闭文件）

0：正常

(32) 将 Rgb 图像保存为 avi 视频文件

IR_SDK_API int IRSDK_SaveRgb2AVI(char* pFile, unsigned char *pRgb, unsigned short Width, unsigned short Height,

unsigned char Op, int quality, unsigned char * pThreadbuf);

char *pFile：input,用于保存文件的路径和文件名，以字符串形式输入

unsigned char * pRgb: input, rgb 数据，

unsigned short Width,：图像宽

unsigned short Height,：图像高

unsigned char Op：写第一帧时，Op = OPEN_FILE

写最后一帧时，Op = CLOSE_FILE

其它情况，Op = WR_FRAME

OPEN_FILE/CLOSE_FILE/WR_FRAME 在头文件中已定义

int quality：input, rgb 需要压缩成 jpg 格式后，再转 avi，该参数定义压缩质量，0~100，越大图像质量越好，压缩比越小，因此，该参数不宜过大或者过小，建议配置为 50

unsigned char * pThreadbuf: input，传入一个指针，开辟不小于 w*h*4Byte 的空间，指针对应空间在调用函数前开辟，注意在一个线程中该空间地址不能变（该空间主要是为了支持多线程，存储函数内部变量）。

return: -1：异常
0：正常

3、 温度相关函数

(33) 温度数据转 CSV 文件

IR_SDK_API int IRSDK_SaveFrame2CSV(char * pFile, Frame *pFrame, float emiss, float dis, float reflect);

char * pFile： input，用于传入文件的路径和文件名，以字符串形式输入

Frame *pFrame: input，温度数据帧，写入到 CSV 文件中

float emiss: input，辐射率，范围 0.01~1.0，默认设置为 0.98

float reflect: input，反射温度（单位摄氏度），即环境温度，默认设置为 20

float dis: input，距离（单位为 m），范围 0~1000，默认设置为 2

return: -1：异常
0：正常

IR_SDK_API int IRSDK_SaveLine2CSV(char* pFile, Frame *pFrame, STAT_LINE sLine, unsigned char u8Format);

char * pFile： input，用于传入文件的路径和文件名，以字符串形式输入

Frame *pFrame: input，温度数据帧，写入到 CSV 文件中

STAT_LINE sLine：input，线对象（注意里面校正系数要赋值）

unsigned char u8Format：存储的格式，只有线支持两种(参数为 0 时，按所在位置方式存储， 为 1 时，按列表的方式存储，具体区别可以通过实验对比)

return: -1：异常
0：正常

其它对象保存 CSV 函数类似：

IR_SDK_API int IRSDK_SaveRect2CSV(char* pFile, Frame *pFrame, STAT_RECT sRect);

IR_SDK_API int IRSDK_SaveCircle2CSV(char* pFile, Frame *pFrame, STAT_CIRCLE sCircle);

IR_SDK_API int IRSDK_SavePolygon2CSV(char* pFile, Frame *pFrame, STAT_POLYGON sPolygon);

(34) 存储温度曲线数据 CSV

IR_SDK_API int IRSDK_SaveObj2CSV(char *pFile, unsigned char Op, STAT_OBJ *pObj, STAT_TEMPER *pGlobalTemper, unsigned char * pThreadbuf);

char *pFile：input,用于保存文件的路径和文件名，以字符串形式输入

unsigned char Op：写第一帧时，Op = OPEN_FILE
写最后一帧时，Op = CLOSE_FILE
其它情况，Op = WR_FRAME
OPEN_FILE/CLOSE_FILE/WR_FRAME 在头文件中已定义

STAT_OBJ *pObj：input，测温对象信息

STAT_TEMPER *pGlobalTemper: input 全局温度信息

unsigned char * pThreadbuf: input，传入一个指针，开辟不小于 1Kbyte 的空间，指针对应空间在调用函数前开辟，注意在一个线程中该空间地址不能变（该空间主要是为了支持多线程，存储函数内部变量）。

return: -1：异常

0：正常

(35) 获取所有测温对象统计

IR_SDK_API int IRSDK_GetObjTemp(Frame *pFrame, STAT_OBJ *pObjStat);

Frame *pFrame：input，输入的帧格式数据

STAT_OBJ *pObjStat：output，传入的结构体中，需要指明对象的个数和坐标信息，然后函数返回所有测温对象的测温信息

return: -1：异常

0：正常

注意: 传入的测温对象结构体中，辐射率、距离、反射温度需要给值，默认可以分别配置为 0.98、2、20.否则可能会出现温度异常

(36) 分别获取测温对象统计

IR_SDK_API int IRSDK_GetPointTemp(Frame *pFrame, STAT_POINT *pPointStat，unsigned char index);

IR_SDK_API int IRSDK_GetLineTemp(Frame *pFrame, STAT_LINE *pLineStat，unsigned char index);

IR_SDK_API int IRSDK_GetCircleTemp(Frame *pFrame, STAT_CIRCLE *pCircleStat，unsigned char index);

IR_SDK_API int IRSDK_GetRectTemp(Frame *pFrame, STAT_RECT *pRectStat，unsigned char index);

IR_SDK_API int IRSDK_GetPolygonTemp(Frame *pFrame, STAT_POLYGON *pPolygonStat，unsigned char index);

Frame *pFrame：input，输入的帧格式数据

STAT_CIRCLE *pCircleStat：output，传入的结构体中，需要指明对象的坐标信息，然后函数返回测温对象的测温信息

unsigned char index : input，用于不同的圆形对象的索引，不能重复。（索引值为 0~31）。（注：当测温时域滤波关闭时，此索引无意义，可以设置为 0，测温滤波默认为关闭，可通过 paramObjTempFilterSw 配置）

return: -1：异常

0：正常

获取其它对象温度（点、线、矩形、多边形）的函数使用方法一样。

注意: 传入的测温对象结构体中，辐射率、距离、反射温度需要给值，默认可以分别配置为 0.98、2、20，否则可能会出现温度异常

(37) 全局温度校正

IR_SDK_API int IRSDK_TempCorrect(float f32Emiss, float f32Reflect, float f32Dis, STAT_TEMPER *pFull_temper);

该函数用于对全局温度进行辐射率、反射温度、距离修正。（该函数只修正全局最高最低平均温度值）。

float f32Emiss: input，全局辐射率，范围 0.01~1.0

float f32Reflect: input，全局反射温度（单位摄氏度），即环境温度

float f32Dis: input，全局距离（单位为 m），范围 2.0~1000，距离小于 2m 时，仅辐射率起作用

STAT_TEMPER *pFull_temper：input/output，全局温度

return: -1：异常

0：正常

(38) 点温度校正

IR_SDK_API float IRSDK_TempPntCorrect(float f32Emiss, float f32Reflect, float f32Dis, float temp);

该函数用于对某点温度进行辐射率、反射温度、距离修正。（该函数只修一个点的温度，比上面全局温度校正函数更为通用，全局温度也可以使用该函数校正）。

float f32Emiss: input，辐射率，范围 0.01~1.0，默认设置为 0.98

float f32Reflect: input，反射温度（单位摄氏度），即环境温度，默认设置为 20

float f32Dis: input，距离（单位为 m），范围 0~1000，默认设置为 2

float temp：input，校正前温度

return: float ：校正后的温度值

4、 运动控制相关函数

(39) 运动控制函数

IR_SDK_API int IRSDK_MoveCtrl(int handle, T_PARAMCTRL mProtocol, T_PARAMCTRL mType, unsigned int u32Param);

该函数用于通过 RS485 协议控制云台、轨道、转台等运动部件。T_PARAMCTRL 类型在头文件中已定义。

int handle：红外句柄， 0- (DEVICE_COUNT-1)范围内整数。

T_PARAMCTRL mProtocol： input，目前支持 4 种协议。 paramPelcod(pelco-d)， paramUserDef1（升降杆），paramUserDef2（舵机），paramUserDef3（转台），后面三种都是针对特定设备的协议，需要时可以具体咨询。

T_PARAMCTRL mType：input，控制类型，支持上下左右停止，波特率配置（目前支持 9600 和 19200，默认为 9600）。

unsigned int u32Param：input，当选择 pelco-D 协议时，该参数为速度，高 16 位为垂直方向速度，低 16 位为水平方向速度；当选择其它协议时，该参数保留。

三、 其它说明

1、 图像数据与温度转换

回调函数中接收到的 pFrame->buffer[MAX_COUNT]中的数据与实际温度的对应关系为

实际温度 = （数据 – 10000）/ x (保留两位小数)

该公式的 x 为 pFrame->u8TempDiv，该数据会随图像数据通过回调函数一起上传。默认情况下 x=100.0。

可以直接调用头文件中的宏定义进行转换 CALTEMP(x, pFrame->u8TempDiv)

2、 回调函数

回调函数有数据回调，命令回调，通信回调，根据需要设置回调函数。不需要的回调函数设置为空。一般只需要使用数据回调，如：

IRSDK_Create (0, IpInfo[0], FrameCallBack, NULL, NULL，long(this));

long(this)：该参数传给回调函数，调用回调时使用，如果不需要传递参数给回调，则可以设置为 NULL.

数据回调声明如下（为了兼容 32 位与 64 位系统，建议回调函数参数用 void * 类型定义）：

```
int FrameCallBack(void* lData, void* lParam);
```

回调函数中尽量只接收数据，数据的处理可以另起线程，防止回调阻塞导致图像卡顿。

3、 相机连接说明

推荐相机连接过程如下：

① SDK 初始化 IRSDK_Init，该函数在程序打开后调用（只需要调用一次，不要反复初始化，与 IRSDK_Quit 对应），调用后才能查询到相机 IP 信息

② 设置 IP 地址接收空间，IRSDK_SetIPAddrArray，该函数与初始化一起，调用后才能查询到相机 IP 信息

注意：上面两步只需要执行一次，一般在打开程序时执行。

③ 获取到 IP 后，调用函数创建 IRSDK_Create，创建红外视频连接

④ 连接 IRSDK_Connect

⑤ 发送连接命令后，通过 IRSDK_IsConnected 返回值确定是否连接上，若返回值为 0，则继续发送 IRSDK_Connect 命令（间隔 500ms 发送一次），直到返回为 1，停止发送。后续要实时监控该状态，发现返回值为 0 时，需要再次发送 IRSDK_Connect 命令（或者一段时间连接失败后，进行超时连接失败的处理）

⑥ 配置 IRSDK_ParamCfg（一般不需要配置，特殊机型（如人体测温类）根据需要配置，有些配置需要在判断连接成功后，才能配置生效，具体参考“参数配置函数”）

示例伪代码如下，详细函数说明见上一节：

```
CBF_IR pCBFframe = &FrameCallBack;   //回调函数
T_IPADDR IpInfo[DEVICE_MAX];         //定义接收 ip 信息的数组
Frame sFrame;                        //接收回调的数据
//界面打开时需要初如化，并设置 IP 地址接收空间，这样才能搜索到 IP 地址，只需要执行一次
IRSDK_Init();
memset(IpInfo,0,sizeof(IpInfo));     //清零
IRSDK_SetIPAddrArray(IpInfo);        //设置接收 IP 的空间
……
//创建，IpInfo 为查询到的 IP 信息，IpInfo[0]为第一个相机，注意先判断一下 IP 是否有效
IRSDK_Create (0, IpInfo[0], pCBFframe, NULL, NULL，long(this));
//配置设备参数(一般不需要配置，特殊设备根据需要配置)
IRSDK_ParamCfg (0, paramDevice，0);
//连接，连接成功后回调就可以接收到数据
IRSDK_Connect(0);
……
//监控相机连接状态，建议 1s 执行一次，实时监测相机连接状态和在线状态，如果没有连接上，则一直发送连接命令
 int Monitor()
 {
      if (0 == IRSDK_IsConnect(0))
      {
           IRSDK_Connect(0);        //连接
      }
```

```
}

//回调函数，IData 为指向帧结构的指针，即获取到的原始数据，IParam 为回调函数的私有参数，即
//IRSDK_Create 时传入的最后一个参数，根据用户需要使用，若不需要，则 Create 时最后一个参数可以置
//为 NULL
  int FrameCallBack(void* IData, void* IParam)
  {
      Frame *pFrame;
      pFrame = (Frame *)IData;       //回调返回的指针在回调函数执行完成后 SDK 会释放掉，要注意
      memcpy(&sFrame, pFrame , sizeof(Frame )); //复制获取到的数据
      //建议数据处理放到另外一个线程中，这样当处理比较耗时或者其它进程阻塞时，不会阻塞回调，
      //当处理不复杂时，可以直接放到回调函数处理
      ......
      return 0;
  }
```

4、 相机断开说明

① 如果只是需要停止数据传输，调用 IRSDK_Stop，该函数使相机停止发送图像数据，发送该命令后，回调不会被执行，再发送 IRSDK_Play 即恢复正常模式，继续发送图像数据。

② 如果需要销毁红外句柄，注销红外连接，需要调用 IRSDK_Destroy，销毁后，需要重新 IRSDK_Create，才可以重新获取到新的句柄，进行后续操作。

③ 退出程序时，需要调用 IRSDK_Quit, SDK 释放资源（只需要调用一次，与 IRSDK_Init 对应）。

5、 修改 IP 说明

修改 IP 前需要保证机器开机并且正常工作连接，才能生效，修改 IP 后，如果修改后的 IP 与电脑 IP 不在同一网段，需要修改电脑的 IP，然后重新连接。

6、 温度校正说明

SDK 中提供了几种进行温度校正的途径：

① 对于点、线、框、圆以及多边形，在调用 sdk 获取这些测温对象的温度时，可以通过各对象结构体中的辐射率、距离、反射温度等参数校正；

② 对于获取到的全局温度（通过调光函数可以获取全局极值温度），可以通过函数（36）进行校正。

③ 函数（37）是对某一个温度值进行校正，用该函数可以对任意像素的温度进行校正，是最灵活的一种方式；

④ 通过 paramTempCorrect 参数进行全屏温度修正， 正负表示增加或者减小，精确到 0.01℃。

注：采用①②③方法进行校正时，只影响获取到的校正温度值，不影响原始数据，从回调获取到的数据不会变化；
采用方法④进行校正时，直接校正全屏的原始数据，从回调获取到的数据会随校正而变化。

7、 调光说明（温度数据转成可显示的 bmp 或者 jpg）

调光函数用于将温度数据转换成可显示的灰度图像，如果要显示伪彩，再利用对应调色板进行调色，调光示例代码如下（以 384*288 为例）：

```
unsigned short pGray [384*288];  (注意：为了版本维护及扩展，这里面用的是16bit数据)
```

```
unsigned char pRgb [384*288*3];
unsigned int gethist[65536];
unsigned char pBmpData[384*288*3+54]; //开辟空间可以大，不能小
 unsigned int BmpLen;
float f32Min = 10;
float f32Max = 100;

//自动调光
IRSDK_FrameConvert(&sFrame, pGray, f32Constrast, f32Bright, gethist, &Full_temper);
//区间调光
//IRSDK_FrameAgc(&sFrame, pGray, f32Constrast, f32Bright, f32Min, f32Max, NULL, 0, gethist,
&Full_temper);
IRSDK_Gray2Rgb(pGray, pRgb, sFrame.width, sFrame.height, PalType, Pal);
IRSDK_Rgb2Bmp(pBmpData, & BmpLen, pRgb, sFrame.width, sFrame.height); //如需 jpg，调用 IRSDK_Rgb2Jpg
```

执行上述函数后，pGray 存储的是灰度图像，pRgb 存储的是 Rgb 数据，pBmpData 存储的是 Bmp 文件数据，BmpLen 存储的是文件长度。

FrameConvert 函数中，可以进行对比度亮度调节，默认情况下，这两个参数配置成 50。(FrameAgc 为自动调光，二者选一)

Gethist 中存放的是对应的直方图，该空间需要在调用前开辟，如果不需要直方图，则将 gethist 设置为 NULL.

对于人体测温类型的机器， f32Constrast 代表显示的最低温，f32Bright 代表显示最高温。对于其它型号机器，这两个分别表示对比度和亮度。

Gray2Rgb 函数中，PalType 表示调色板类型，一般配置为 0，Pal 为不同的调色板选择，具体见函数说明。

8、 人体测温类型设备配置

针对人体测温类型，下面几点需要注意：

① 设备类型配置为 1：

IRSDK_ParamCfg(0, paramDevice, 1);

配置为 1 后，回调得到的图像高>宽。

② 调光函数 IRSDK_FrameConvert

对于人体测温类型的机器， 该函数参数 f32Constrast 代表显示的调光的最低温，f32Bright 代表显示调光的最高温，在范围之外的显示固定的颜色，在范围之类的会进行拉伸，用来显示更丰富的细节信息。

9、 外触发功能

相机支持多种触发模式，具体详见《红外热像仪外触发使用说明》。

部分型号产品可以支持外触发功能，外触发信号采用 RS485 电平，通过相机尾部“D-D+”接口接入。

上电后，如果未检测到上降沿，相机会一直处于等待触发状态，无视频数据输出；

当检测到第一次下降沿时，相机开始同步输出视频流；

当检测到上升沿时，会将对应帧数据的 u8TriggerFrame 标记置 1，用于标记该帧是触发帧。上位机可以根据具体需求对触发帧进行处理。

```
//帧格式
    typedef struct tagFrame
    {
      unsigned short width;          //图像宽度
      unsigned short height;         //图像高度
      unsigned short u16FpaTemp;     // unused，焦面温度
      unsigned short u16EnvTemp;     // unused,环境温度
      unsigned char  u8TempDiv;      //数据转换成温度时需要除以该数据，文档具体说明
      unsigned char  u8DeviceType;   //unused
      unsigned char  u8SensorType;   //unused
      unsigned char  u8MeasureSel;   //unused
      unsigned char  u8Lens;         //unused
      unsigned char  u8Fps;          //unused
      unsigned char  u8TriggerFrame; //当有外触发功能时，表示该帧是否是触发帧
      unsigned char  Reversed[17];   //unused

      unsigned short buffer[327680]; //图像数据，按行存储，最大支持 640x512，每个像素是一个 ushort 类型
    } Frame;
```

四、 异常处理

现象	可能原因	解决
电脑的网络连接显示红叉	1、 网线连接问题 2、 电脑网口故障	1、 更换网线 2、 检查电脑网口
正常创建句柄后，发送 IRSDK_Connect 命令后，回调获取不到数据	1、 电脑 IP 与设备 IP 不在同一个网段 2、 操作系统防火墙阻止了设备连接 3、 网卡设置问题	1、 将电脑 IP 网段修改成与设备 IP 网段一致（IP 地址不能相同） 2、 关闭操作系统防火墙 3、 对于 100Hz 帧频的产品，需要网卡开启巨帧功能，并且要与电脑直连（如果接交换机，请确保交换机也支持巨帧）
修改 IP 没有生效	1、当前电脑 IP 与设备 IP 不在同一个网段	1、将电脑 IP 网段修改成与设备 IP 网段一致（IP 地址不能相同）
帧频异常	1、 网卡速度不匹配 2、 电脑配置过低	1、 检查网卡配置(建议千兆网卡) 2、 升级电脑配置（建议 9 代 i7 主频 2.6G 以上，8G 内存以上）

以上，如有未表达清楚之处，请及时沟通，谢谢！