**网络测温机芯SDK用户使用手册**

目录

[1. 概述 4](#_Toc27680)

[2. 函数详解 4](#_Toc11896)

[2.1 SDK初始化 4](#_Toc31577)

[2.2 SDK释放 4](#_Toc16941)

[2.3 设备搜索 4](#_Toc27070)

[2.4 设备登入 5](#_Toc12691)

[2.5 设备登出 6](#_Toc18023)

[2.6 打开数据流 6](#_Toc2847)

[2.7 关闭数据流 7](#_Toc6861)

[2.8 获取设备类型 7](#_Toc23536)

[2.9 查询SDK支持功能 7](#_Toc27380)

[2.10 获取设备信息 10](#_Toc7636)

[2.11 快门 11](#_Toc8160)

[2.12 拍照 13](#_Toc23142)

[2.13 Y16录像 16](#_Toc30404)

[2.14 AVI录像 16](#_Toc21513)

[2.15 设置色带 17](#_Toc8840)

[2.16 获取色带数据 17](#_Toc19409)

[2.17 设置Y16帧率 18](#_Toc31384)

[2.18 调焦 19](#_Toc29588)

[2.19 获取SDK版本信息 21](#_Toc10523)

[2.20 修改机芯设备IP 22](#_Toc21954)

[2.21 获取测温参数 23](#_Toc7178)

[2.22 设置测温参数 26](#_Toc2223)

[2.23 保存测温参数 30](#_Toc22452)

[2.24 获取实时点温 30](#_Toc29044)

[2.25 获取实时全图温 31](#_Toc32440)

[2.26 获取实时矩形温 32](#_Toc14996)

[2.27 获取图片温 33](#_Toc17019)

[2.28 添加矩形分析对象 35](#_Toc1282)

[2.29 视频流叠加字符串 38](#_Toc18596)

[2.30 电子变焦 41](#_Toc20042)

[2.31 调光模式 42](#_Toc13607)

[2.32 亮度对比度调光 42](#_Toc7384)

[2.33 手动温度调光 44](#_Toc17883)

[2.34 设置机芯设备温度报警 45](#_Toc19390)

[3. 其它 47](#_Toc5057)

[3.1 计算机配置性能建议 47](#_Toc22887)

[3.2 SDK使用注意事项 47](#_Toc28501)

[4. SDK更新 47](#_Toc30997)

[5. SDK版本号 50](#_Toc14506)

# 概述

本SDK在Windows7系统下开发，使用VS2015开发工具。SDK适用于Windows系统下调用。

本SDK用于测温机芯系列。使用本SDK可以从设备获取H264码流、RGB数据、温度数据、支持拍照、录像、IP可配置、调焦以及从图片获取任意点温度、全图温度和区域温度信息等功能。

本SDK使用IPv4协议。

# 函数详解

## SDK初始化

int32\_t GD\_MTC\_SDK\_Init();

**功能:**

SDK初始化.此SDK中大部分接口调用时要求SDK已经初始化, 用户在程序初始化时调用此函数初始化SDK.

**参数:**

无.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## SDK释放

int32\_t GD\_MTC\_SDK\_Release();

**功能:**

SDK释放.用户在程序退出前调用此函数释放SDK.

**参数:**

无.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 设备搜索

int32\_t GD\_MTC\_SDK\_StartSearchDevice(PROCESS\_SEARCH\_DEVICE\_CB pSearchDeviceCallBack, int \*pUser);

**功能:**

在网络中搜索机芯设备.

**参数:**

pSearchDeviceCallBack:

搜索结果回调函数. 具体参照PROCESS\_SEARCH\_DEVICE\_CB说明.

pUser:

用户指针.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int (CALLBACK\* PROCESS\_SEARCH\_DEVICE\_CB)(const GD\_MTC\_IRDeviceInfo \* pDeviceInfo, int nDeviceInfoSize, void \*pUser);

**功能:**

搜索机芯设备回调函数.

**参数:**

pDeviceInfo:

搜索到的设备信息的地址.

nDeviceInfoSize:

搜索到的设备信息的条数.

pUser:

用户指针.

**返回值:**

返回1.

## 设备登入

int32\_t GD\_MTC\_SDK\_Login(GD\_MTC\_IRDeviceInfo DeviceInfo, int \*pUser);

**功能:**

设备登入.获取数据流前需要成功登入.

**参数:**

DeviceInfo:

设备信息, 只需向此结构中填入IpAddress字段和DevType字段再调用此函数即可.**当传入的表示设备类型的DevType字段的值为DEVICE\_TYPE\_UNKNOWN时, 此函数实现中会自动识别设备类型, 否则函数认为传入的设备类型就是将要连接的设备类型.可通过**GD\_MTC\_SDK\_GetDeviceType**接口在登入前获取设备类型.**

pUser:

传0值即可.

**返回值:**

登入ID值. 非ERROR\_LOGIN\_FAIL值表示调用成功, ERROR\_LOGIN\_FAIL值表示调用失败.

## 设备登出

int32\_t GD\_MTC\_SDK\_Logout(const int iLoginID);

**功能:**

设备登出.

**参数:**

iLoginID:

调用设备登入函数成功时得到的登入ID.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 打开数据流

int32\_t GD\_MTC\_SDK\_OpenStream(const int iLoginID, GD\_MTC\_IRLinkInfo \*pIRLinkInfo, int iPaletteIndex, int iFps, int \*pUser);

**功能:**

打开数据流.获取测温及图像数据前需要打开数据流.

**参数:**

iLoginID:

调用设备登入函数成功时得到的登入ID.

pIRLinkInfo:

连接信息对象地址.在此参数传入对应的数据回调函数及获取数据流的方式.

iPaletteIndex:

图像的初始色带.值的范围为0-7.

iFps:

Y16数据的初始帧率.值的范围为1-25.

pUser:

用户指针.传入0值即可.

**返回值:**

数据流ID. 非ERROR\_OPENSTREAM\_FAIL值表示调用成功, ERROR\_OPENSTREAM\_FAIL值表示调用失败.

## 关闭数据流

int32\_t GD\_MTC\_SDK\_CloseStream(const int iSessionID);

**功能:**

关闭数据流.

**参数:**

iSessionID:

调用打开数据流函数成功时得到的数据流ID.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 获取设备类型

bool GD\_MTC\_SDK\_GetDeviceType(const char \* strIpAddress, GD\_MTC\_IRDeviceType & nDeviceType);

**功能:**

通过IP地址获取设备类型.

**参数:**

strIpAddress:

机芯设备的IP地址.

nDeviceType:

获取到的机芯的设备类型.

**返回值:**

如果成功获取取设备类型返回true, 否则返回false.

## 查询SDK支持功能

int32\_t GD\_MTC\_SDK\_SupportFunction(GD\_MTC\_IRDeviceType nDeviceType, PULL\_DATA\_TYPE nPullDataType, DEVICE\_FUNCTION\_SUPPORT nFunctionType);

**功能:**

查询SDK对特定机型数据流模式是否支持及对特定机型的指定SDK功能接口是否支持.

例如, 以下函数调用分别查询OR2设备是否支持Y8数据流, Y16数据流及Y8+Y16数据流模式:

GD\_MTC\_SDK\_SupportFunction(OR2, PULL\_DATA\_TYPE\_H264\_ONLY, DFS\_Y8)

GD\_MTC\_SDK\_SupportFunction(OR2, PULL\_DATA\_TYPE\_Y16\_ONLY, DFS\_Y16)

GD\_MTC\_SDK\_SupportFunction(OR2, PULL\_DATA\_TYPE\_H264\_AND\_Y16, DFS\_Y8\_AND\_Y16)

以下函数调用分别查询NET机型在Y8模式是否支持Y16录像, NET800x600机型在Y16模式是否支持自动调焦:

GD\_MTC\_SDK\_SupportFunction(NET, PULL\_DATA\_TYPE\_H264\_ONLY, DFS\_Y16\_RECORD)

GD\_MTC\_SDK\_SupportFunction(NET800x600, PULL\_DATA\_TYPE\_Y16\_ONLY, DFS\_AUTO\_FOCUS)

**参数:**

nDeviceType:

设备型号.

nPullDataType:

获取数据流类型.

nFunctionType:

功能项.

**返回值:**

1表示支持此项功能, 0表示不支持.

说明:

以下是SDK各函数对应的功能支持枚举项

|  |  |
| --- | --- |
| 枚举项 | 函数名 |
| DFS\_Y8 | GD\_MTC\_SDK\_OpenStream |
| DFS\_Y16 | GD\_MTC\_SDK\_OpenStream |
| DFS\_Y8\_AND\_Y16 | GD\_MTC\_SDK\_OpenStream |
| DFS\_SHUTTER | GD\_MTC\_SDK\_Shutter |
| DFS\_SHUTTER\_CLOSE | GD\_MTC\_SDK\_ShutterClose |
| DFS\_SHUTTER\_OPEN | GD\_MTC\_SDK\_ShutterOpen |
| DFS\_SNAP | GD\_MTC\_SDK\_SnapPicture |
| DFS\_SNAP\_NO\_CB | GD\_MTC\_SDK\_SnapPictureNoCb |
| DFS\_SNAP\_WITHOUT\_TEMP\_NO\_CB | GD\_MTC\_SDK\_SnapPictureWithoutTempNoCb |
| DFS\_SNAP\_WITHOUT\_IMG\_NO\_CB | GD\_MTC\_SDK\_SnapPictureWithoutImgNoCb |
| DFS\_SNAP\_UNIFORM | GD\_MTC\_SDK\_SnapPictureUniform |
| DFS\_AVI\_RECORD | GD\_MTC\_SDK\_StartRecordAvi  GD\_MTC\_SDK\_StopRecordAvi |
| DFS\_Y16\_RECORD | GD\_MTC\_SDK\_StartRecord  GD\_MTC\_SDK\_StopRecord |
| DFS\_SET\_PALETTE | GD\_MTC\_SDK\_SetPalette |
| DFS\_GET\_PALETTE\_DATA\_SIZE | GD\_MTC\_SDK\_GetPaletteDataSize |
| DFS\_GET\_PALETTE\_DATA | GD\_MTC\_SDK\_GetPaletteData |
| DFS\_SET\_FRAME\_RATE | GD\_MTC\_SDK\_setY16FramePerSecond |
|  |  |
| DFS\_NEAR\_FOCUS | GD\_MTC\_SDK\_FocusNear  GD\_MTC\_SDK\_FocusStop |
| DFS\_FAR\_FOCUS | GD\_MTC\_SDK\_FocusFar  GD\_MTC\_SDK\_FocusStop |
| DFS\_AUTO\_FOCUS | GD\_MTC\_SDK\_FocusAuto |
| DFS\_NEAR\_FOCUS\_FINE | GD\_MTC\_SDK\_FocusNearFine |
| DFS\_FAR\_FOCUS\_FINE | GD\_MTC\_SDK\_FocusFarFine |
| DFS\_GET\_MOTOR\_POS | GD\_MTC\_SDK\_getMotorPostition |
| DFS\_SET\_MOTOR\_POS | GD\_MTC\_SDK\_setMotorPostition |
|  |  |
| DFS\_SET\_UART\_CHECK\_ADDRESS | GD\_MTC\_SDK\_SetUartCheckAddress |
| DFS\_SET\_UART\_RATE | GD\_MTC\_SDK\_SetUartRate |
|  |  |
| DFS\_DISABLE\_AUTO\_SHUTTER | GD\_MTC\_SDK\_DisableShutter |
| DFS\_ENABLE\_AUTO\_SHUTTER | GD\_MTC\_SDK\_EnableShutter |
| DFS\_SET\_AUTO\_SHUTTER\_TIME | GD\_MTC\_SDK\_SetShutterTime |
|  |  |
| DFS\_ADD\_RECT\_ANALYSER | GD\_MTC\_SDK\_AddRectAnalyser |
| DFS\_DELETE\_RECT\_ANALYSER | GD\_MTC\_SDK\_DeleteRectAnalyser |
| DFS\_MAX\_MIN\_AVG\_TYPE | GD\_MTC\_SDK\_ChangeAnalyserTemperatureShowType |
| DFS\_ENABLE\_DIGITAL\_TEMP\_WITH\_CURSOR | GD\_MTC\_SDK\_EnableRectAnalyserDigitalTempWithCursor  GD\_MTC\_SDK\_DisableRectAnalyserDigitalTempWithCursor |
|  |  |
| DFS\_ADD\_STRING | GD\_MTC\_SDK\_AddStringToStream |
| DFS\_ENABLE\_ADD\_STRING | GD\_MTC\_SDK\_EnableAddStringToStream |
| DFS\_DISABLE\_ADD\_STRING | GD\_MTC\_SDK\_DisableAddStringToStream |
|  |  |
| DFS\_GET\_DEVICE\_SERIAL\_NUMBER | GD\_MTC\_SDK\_GetDeviceSerialNumber |
| DFS\_GET\_ARM\_VERSION | GD\_MTC\_SDK\_GetArmVersion |
| DFS\_GET\_FPGA\_VERSION | GD\_MTC\_SDK\_GetFpgaVersion |
| DFS\_GET\_SDK\_VERSION | GD\_MTC\_SDK\_GetSDKVersion |
|  |  |
| DFS\_CHANGE\_IP | GD\_MTC\_SDK\_ChangeDeviceAddress |
| DFS\_CHANGE\_IP\_NO\_CONNECT | GD\_MTC\_SDK\_ChangeDeviceAddressWithoutConnection |
|  |  |
| DFS\_GET\_EMISS | GD\_MTC\_SDK\_getEmiss |
| DFS\_SET\_EMISS | GD\_MTC\_SDK\_setEmiss |
| DFS\_GET\_DISTANCE | GD\_MTC\_SDK\_getDistance |
| DFS\_SET\_DISTANCE | GD\_MTC\_SDK\_setDistance |
| DFS\_GET\_HUMIDITY | GD\_MTC\_SDK\_getHumidity |
| DFS\_SET\_HUMIDITY | GD\_MTC\_SDK\_setHumidity |
| DFS\_GET\_BACKGROUND\_TEMP | GD\_MTC\_SDK\_getBackgroundTemp |
| DFS\_SET\_BACKGROUND\_TEMP | GD\_MTC\_SDK\_setBackgroundTemp |
| DFS\_BACKGROUND\_TEMP\_SWITCH\_OPEN | GD\_MTC\_SDK\_getBackgroundTempSwitch |
| DFS\_BACKGROUND\_TEMP\_SWITCH\_CLOSED | GD\_MTC\_SDK\_setBackgroundTempSwitch |
| DFS\_GET\_B2 | GD\_MTC\_SDK\_getB2 |
| DFS\_SET\_B2 | GD\_MTC\_SDK\_setB2 |
| DFS\_GET\_KF | GD\_MTC\_SDK\_GetGearKf |
| DFS\_SET\_KF | GD\_MTC\_SDK\_SetGearKf |
| DFS\_SET\_VSK\_GEAR | GD\_MTC\_SDK\_SetVskGear |
| DFS\_SAVE\_ALL\_PARM | GD\_MTC\_SDK\_saveCalcParm |
|  |  |
| DFS\_GET\_BRIGHTNESS | GD\_MTC\_SDK\_GetBrightness |
| DFS\_SET\_BRIGHTNESS | GD\_MTC\_SDK\_SetBrightness |
| DFS\_GET\_CONTRAST | GD\_MTC\_SDK\_GetContrast |
| DFS\_SET\_CONTRAST | GD\_MTC\_SDK\_SetContrast |
|  |  |
| DFS\_GET\_Y16\_POINT\_TEMP | GD\_MTC\_SDK\_GetPointTemp |
| DFS\_GET\_Y16\_MATRIX\_TEMP | GD\_MTC\_SDK\_GetTempMatrix |
| DFS\_GET\_Y16\_MAX\_TEMP | GD\_MTC\_SDK\_GetHighTemp |
| DFS\_GET\_Y16\_MIN\_TEMP | GD\_MTC\_SDK\_GetLowTemp |
| DFS\_GET\_Y16\_AVG\_TEMP | GD\_MTC\_SDK\_GetAvegTemp |
| DFS\_GET\_Y16\_RECT\_MAX\_TEMP | GD\_MTC\_SDK\_GetRectHighTemp |
| DFS\_GET\_Y16\_RECT\_MIN\_TEMP | GD\_MTC\_SDK\_GetRectLowTemp |
| DFS\_GET\_Y16\_RECT\_AVG\_TEMP | GD\_MTC\_SDK\_GetRectAvegTemp |
|  |  |
| DFS\_GET\_PIC\_POINT\_TEMP | GD\_MTC\_SDK\_GetPointTemperatureFromPicture  GD\_MTC\_SDK\_GetTempPictureDemension |
| DFS\_GET\_PIC\_MATRIX\_TEMP | GD\_MTC\_SDK\_GetGlobalTemperatureFromPicture |
| DFS\_GET\_PIC\_MAX\_MIN\_AVG\_TEMP | GD\_MTC\_SDK\_GetRectTemperatureFromPicture |
|  |  |
| DFS\_GET\_DEVICE\_ALARM\_STATE | GD\_MTC\_SDK\_GetDeviceAlarmState |
| DFS\_SET\_DEVICE\_ALARM\_STATE | GD\_MTC\_SDK\_SetDeviceAlarmState |
| DFS\_GET\_DEVICE\_ALARM\_MAX\_TEMP | GD\_MTC\_SDK\_GetDeviceAlarmMax |
| DFS\_SET\_DEVICE\_ALARM\_MAX\_TEMP | GD\_MTC\_SDK\_SetDeviceAlarmMax |
| DFS\_GET\_DEVICE\_ALARM\_MIN\_TEMP | GD\_MTC\_SDK\_GetDeviceAlarmMin |
| DFS\_SET\_DEVICE\_ALARM\_MIN\_TEMP | GD\_MTC\_SDK\_SetDeviceAlarmMin |
|  |  |
| DFS\_GET\_RECT\_ANALYSER\_INFO | GD\_MTC\_SDK\_GetRectAnalyserInfo |
| DFS\_GET\_ADD\_STRING\_INFO | GD\_MTC\_SDK\_GetAddStringInfo |
|  |  |
| DFS\_SET\_EX\_LEVEL | GD\_MTC\_SDK\_SetExLevel  GD\_MTC\_SDK\_GetExLevel |
|  |  |

## 获取设备信息

int32\_t GD\_MTC\_SDK\_GetDeviceInfo(const int iLoginID, GD\_MTC\_IRDeviceInfo \*pIRDevice, int \*pUser);

**功能:**

获取机芯设备信息.目前支持的字段包括: 设备IP, 设备类型, 设备版本号, 图像宽, 图像高.GD\_MTC\_IRDeviceInfo结构体中DeviceVersion字段表示设备获取到的设备版本号, 此字符串以一个空格为分隔, 空格前为ARM版本号, 空格后为FPGA版本号.

**参数:**

iLoginID:

调用打开数据流成功时得到的数据流ID.

pIRDevice:

设备信息对象地址.如果调用成功, 通过此参数读取设备信息.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 快门

int32\_t GD\_MTC\_SDK\_Shutter(const int iLoginID, int \*pUser);

**功能:**

快门. 注意, 在成功调用GD\_MTC\_SDK\_FocusAuto接口且机芯在调焦过程动作结束前调用此接口会失败.

**参数:**

iLoginID:

登入ID.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_ShutterClose(const int iLoginID, int \*pUser);

**功能:**

关闭快门.注意, 在成功调用GD\_MTC\_SDK\_FocusAuto接口且机芯在调焦过程动作结束前调用此接口会失败.**注意,如果需要在让快门一直保持闭合状态,在调用此接口前可调用GD\_MTC\_SDK\_DisableShutter接口禁用自动快门.**

**参数:**

iLoginID:

登入ID.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_ShutterOpen(const int iLoginID, int \*pUser);

**功能:**

弹开快门.注意, 在成功调用GD\_MTC\_SDK\_FocusAuto接口且机芯在调焦过程动作结束前调用此接口会失败.

**参数:**

iLoginID:

登入ID.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_EnableShutter(const int iLoginID, int \*pUser);

**功能:**

启用机芯自动快门.

**参数:**

iLoginID:

登入ID.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_DisableShutter(const int iLoginID, int \*pUser);

**功能:**

禁用机芯自动快门. 禁用机芯自动快门后机芯设备不会自动打快门, 但程序仍可调用GD\_MTC\_SDK\_Shutter接口打快门.

**参数:**

iLoginID:

登入ID.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_SetShutterTime(const int iLoginID, int nMin, int \*pUser);

**功能:**

设置机芯自动快门的时间间隔.

**参数:**

iLoginID:

登入ID.

nMin:

机芯自动快门的时间间隔, 单位分钟, 值的范围为1~255.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 拍照

int32\_t GD\_MTC\_SDK\_SnapPictureUniform(const int iSessionID, const char \* strFilePath, bool bFullPath, bool bImagData, bool bTempData, PROCESS\_EDIT\_SNAP\_PIC\_CB pSnapPicCb = 0, int \* pUser = 0);

**功能:**

通用版本的拍照接口函数. 可指定保存图片的文件夹名称或文件全路径, 可指定是否包含图像信息, 是否包含温度信息, 可在保存图片前修改或自定义图像数据.

**参数:**

iSessionID:

数据流ID.

strFilePath:

保存拍照图片的文件夹路径或图片的全路径.当参数bFullPath为false时, strFilePath参数为保存图片的文件夹路径, 当bFullPath参数为true时, strFilePath参数为保存图片的全路径.

bFullPath:

指示strFilePath参数是否是保存的全路径.

bImagData:

当此参数为true时表示保存的图片中包含图像数据, 否则不包含.

bTempData:

当此参数为true时表示保存的图片中包含温度数据, 否则不包含.

pSnapPicCb :

在保存图像前修改或定义图像数据的回调函数.具体参照的PROCESS\_EDIT\_SNAP\_PIC\_CB说明.

pUser :

用户指针.SDK内部在调用pSnapPicCb 回调函数时反馈给调用者.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

typedef int (CALLBACK\* PROCESS\_EDIT\_SNAP\_PIC\_CB)(unsigned char \* pRgbData, int nRgbDataLen, int nRgbWidth, int nRgbHeight, void \* pUser);

**功能:**

修改或自定义拍照图像数据回调函数. 通过自定义此函数实现在保存图片前修改或指定图片的图像部分数据.

**参数:**

pRgbData:

图像的RGB数据.

nRgbDataLen:

RGB数据的字节长度.

nRgbWidth:

图像的宽度.

nRgbHeight:

图像的高度.

pUser:

用户指针.

**返回值:**

返回1.

int32\_t GD\_MTC\_SDK\_SnapPicture(void \* pOpque, const char \* strFilePath);

**功能:**

回调形式的含温度信息拍照.在自定义的Y16回调中调用此函数.使用此函数得到的图片文件中同时包含图像数据和温度数据.

**参数:**

pOpque:

Y16回调函数传入的对应值.

strFilePath:

保存拍照图片的路径.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_SnapPictureNoCb(const int iSessionID, const char \* strFilePath);

**功能:**

非回调用形式的含温度信息拍照.使用此函数得到的图片文件中同时包含图像数据和温度数据.

**参数:**

iSessionID:

数据流ID.

保存拍照图片的路径.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_SnapPictureWithoutTempNoCb(const int iSessionID, const char \* strFilePath);

**功能:**

非回调形式的不含温度信息拍照.使用此函数得到的图片文件中仅含图像数据.

**参数:**

iSessionID:

数据流ID.

strFilePath:

保存拍照图片的路径.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_SnapPictureWithoutImgNoCb(const int iSessionID, const char \* strFilePath);

**功能:**

非回调形式的不含图像信息拍照.使用此函数得到的图片文件中仅含温度数据.

**参数:**

iSessionID:

数据流ID.

strFilePath:

保存拍照图片的路径.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## Y16录像

int32\_t GD\_MTC\_SDK\_StartRecord(const int iSessionID, const char \* strFilePath);

**功能:**

开始Y16录像.调用此函数生成的包含温度信息的Y16视频文件, 文件格式是irgd, 此类型文件需要用专门的红外分析工具查看. 注意, 在关闭数据流前如果正在进行Y16录像, 请调用GD\_MTC\_SDK\_StopRecord函数停止录像.

**参数:**

iSessionID:

数据流ID.

strFilePath:

保存录像文件的路径.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_StopRecord(const int iSessionID);

**功能:**

停止Y16录像. 注意, 在关闭数据流前如果正在进行Y16录像, 请调用此函数停止录像.

**参数:**

iSessionID:

数据流ID.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## AVI录像

int32\_t GD\_MTC\_SDK\_StartRecordAvi(const int iSessionID, const char \* strFilePath);

**功能:**

开始AVI录像.调用此函数生成的AVI文件可用支持此格式的视频播放器播放. 注意, 在关闭数据流前如果正在进行AVI录像, 请调用GD\_MTC\_SDK\_StopRecordAvi函数停止录像.

**参数:**

iSessionID:

数据流ID.

strFilePath:

保存录像文件的路径.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_StopRecordAvi(const int iSessionID);

**功能:**

停止AVI录像. 注意, 在关闭数据流前如果正在进行AVI录像, 请调用此函数停止录像.

**参数:**

iSessionID:

数据流ID.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 设置色带

int32\_t GD\_MTC\_SDK\_SetPalette(const int iSessionID, int iPaletteIndex, int \*pUser);

**功能:**

设置图像的色带.

**参数:**

iSessionID:

数据流ID.

iPaletteIndex:

要设置的色带号.值的范围为0-7.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 获取色带数据

int32\_t GD\_MTC\_SDK\_GetPaletteDataSize(const int iSessionID, int \* pValue);

**功能:**

获取色带数据的字节长度.

**参数:**

iSessionID:

数据流ID.

pValue:

接收数据长度的地址.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetPaletteData(const int iSessionID, unsigned char \* pPaletteData, int \* pDemension0, int \* pDemension1, int \* pDemension2, char \* strDataFormat);

**功能:**

获取色带数据.

**参数:**

iSessionID:

数据流ID.

pPaletteData:

色带数据, 一个三维数组.

pDemension0:

色带数据第一个维度的大小.即色带的序号.

pDemension1:

色带数据第二个维度的大小.即一条色带中包含的颜色点的个数.

pDemension2:

色带数据第三个维度的大小.即一个颜色点的数据.

strDataFormat:

色带中单个颜色点的格式, 例如”bgra”表示4个字节, 依次是蓝,绿,红,透明.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 设置Y16帧率

int32\_t GD\_MTC\_SDK\_setY16FramePerSecond(const int iLoginID, int iFps, int \*pUser);

**功能:**

设置Y16数据的帧率.

**参数:**

iLoginID:

登入ID.

iFps:

新的Y16数据帧率.帧率的范围为1~25.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 调焦

int32\_t GD\_MTC\_SDK\_FocusNear(const int iSessionID, int \*pUser);

**功能:**

调近焦. 调用此接口后机芯设备会持续将镜头拉近, 直至最近或者调用GD\_MTC\_SDK\_FocusStop停止调焦. 注意, 在成功调用GD\_MTC\_SDK\_Shutter接口且机芯正在执行快门动作期间调用, 调用此接口会失败.

**参数:**

iSessionID:

数据流ID.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_FocusFar(const int iSessionID, int \*pUser);

**功能:**

调远焦. 调用此接口后机芯设备会持续将镜头拉远, 直至最远或者调用GD\_MTC\_SDK\_FocusStop停止调焦. 注意, 在成功调用GD\_MTC\_SDK\_Shutter接口且机芯正在执行快门动作期间调用, 调用此接口会失败.

**参数:**

iSessionID:

数据流ID.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_FocusStop(const int iSessionID, int \*pUser);

**功能:**

停止调焦.注意, 在成功调用GD\_MTC\_SDK\_Shutter操作且机芯正在执行快门动作期间调用, 调用此接口会失败.

**参数:**

iSessionID:

数据流ID.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_FocusAuto(const int iSessionID, int \*pUser);

**功能:**

自动调焦. 调用此函数后机芯设备自动动作调焦至最合适状态.注意, 在成功调用GD\_MTC\_SDK\_Shutter接口且机芯正在执行快门动作期间调用, 调用此接口会失败.

**参数:**

iSessionID:

数据流ID.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_FocusFarFine(const int iSessionID, int \*pUser);

**功能:**

远焦微调. 注意, 在成功调用GD\_MTC\_SDK\_Shutter接口且机芯正在执行快门动作期间调用, 调用此接口会失败.

**参数:**

iSessionID:

数据流ID.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_FocusNearFine(const int iSessionID, int \*pUser);

**功能:**

近焦微调. 注意, 在成功调用GD\_MTC\_SDK\_Shutter接口且机芯正在执行快门动作期间调用, 调用此接口会失败.

**参数:**

iSessionID:

数据流ID.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_setMotorPostition(const int iLoginID, int iMotorPos, int \*pUser);

**功能:**

设置调焦电机位置.注意, 在成功调用GD\_MTC\_SDK\_Shutter接口且机芯正在执行快门动作期间调用, 调用此接口会失败.

**参数:**

iLoginID:

登入ID.

iMotorPos:

电机将要移动到的位置.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_getMotorPostition(const int iLoginID, int \* pMotorPos, int \*pUser);

**功能:**

获取调焦电机位置.

**参数:**

iLoginID:

登入ID.

pMotorPos:

接收当前电机位置的地址.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 获取SDK版本信息

int32\_t GD\_MTC\_SDK\_GetSDKVersion(const int iLoginID, char \*SDKVersion, int \*pUser);

**功能:**

获取SDK版本号.

**参数:**

iLoginID:

传入0值即可.

SDKVersion:

函数向此地址处写入SDK版本号.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

获取机芯序列号

int32\_t GD\_MTC\_SDK\_GetDeviceSerialNumber(const int iLoginID, char\* serialNum, int \*pUser);

**功能:**

获取机芯设备的序列号.

**参数:**

iLoginID:

登入ID.

serialNum:

函数向此地址处写入序列号.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 修改机芯设备IP

int GD\_MTC\_SDK\_ChangeDeviceAddressWithoutConnection(const char\* sNewIP, const char\* sNetMask, const char\* sNewGateWay, const char\* sNewDNS, const char\* sOriIP);

**功能:**

修改设备的IP地址.调用函数不要求已经登入和已打开数据流, 如果已经打开数据流, 请先关闭之. 此函数与GD\_MTC\_SDK\_ChangeDeviceAddress函数相比, 建议优先使用此函数.在设备是未登入状态时调用此函数.

**参数:**

sNewIP:

新的IP地址.

sNetMask:

子网掩码.

sNewGateWay:

新的网关.

sNewDNS:

新的DNS.

sOriIP:

原来的IP.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int GD\_MTC\_SDK\_ChangeDeviceAddress(const int iSessionID, const char\* sNewIP, const char\* sNetMask, const char\* sNewGateWay, const char\* sNewDNS, const char\* sOriIP);

**功能:**

修改设备的IP地址.调用函数要求已经登入和已打开数据流.此函数与GD\_MTC\_SDK\_ChangeDeviceAddressWithoutConnection函数相比, 建议优先使用后者.

**参数:**

iSessionID:

数据流ID.

sNewIP:

新的IP地址.

sNetMask:

子网掩码.

sNewGateWay:

新的网关.

sNewDNS:

新的DNS.

sOriIP:

原来的IP.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 获取测温参数

int32\_t GD\_MTC\_SDK\_getEmiss(const int iLoginID, float \* pEmiss, int \*pUser);

**功能:**

获取发射率.

**参数:**

iLoginID:

登入ID.

pEmiss:

接收当前发射率的地址.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_getDistance(const int iLoginID, float \* pDistance, int \*pUser);

**功能:**

获取距离.

**参数:**

iLoginID:

登入ID.

pDistance:

接收当前距离的地址.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_getHumidity(const int iLoginID, int \* pHumidity, int \*pUser);

**功能:**

获取湿度.

**参数:**

iLoginID:

登入ID.

pHumidity:

接收当前湿度的地址.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_getBackgroundTemp(const int iLoginID, float \* pBgTemp, int \*pUser);

**功能:**

获取背景温度.

**参数:**

iLoginID:

登入ID.

pBgTemp:

接收当前背景温度的地址. 背景温度的单位是摄氏度.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_getBackgroundTempSwitch(const int iLoginID, int \* pEnable, int \*pUser);

**功能:**

获取背景温度开关.

**参数:**

iLoginID:

登入ID.

pEnable:

接收当前背景温度开关的地址. 1表示开状态, 0表示关状态.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_getB2(const int iLoginID, float \* pB2, int \*pUser);

**功能:**

获取B2参数.

**参数:**

iLoginID:

登入ID.

pB2:

接收当前B2参数的地址.B2参数的单位是摄氏度.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetGearKf(const int iSessionID, int \* pKf);

**功能:**

获取KF参数.

**参数:**

iSessionID:

数据流ID.

pKf:

接收当前KF参数的地址.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 设置测温参数

int32\_t GD\_MTC\_SDK\_setEmiss(const int iLoginID, float fEmiss, int \*pUser);

**功能:**

设置发射率.注意修改此参数后只对设备此次开机有效, 设备断电重启后此设置无效, 如需此修改在设备断电重启后仍然有效, 请调用GD\_MTC\_SDK\_saveCalcParm函数保存修改到设备.

**参数:**

iLoginID:

登入ID.

fEmiss:

新的发射率. 值的范围为0到1.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_setDistance(const int iLoginID, float fDistance, int \*pUser);

**功能:**

设置距离.注意修改此参数后只对设备此次开机有效, 设备断电重启后此设置无效, 如需此修改在设备断电重启后仍然有效, 请调用GD\_MTC\_SDK\_saveCalcParm函数保存修改到设备.

**参数:**

iLoginID:

登入ID.

fDistance:

距离, 单位米, 值的范围为1~10.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_setHumidity(const int iLoginID, int iHumidity, int \*pUser);

**功能:**

设置湿度.注意修改此参数后只对设备此次开机有效, 设备断电重启后此设置无效, 如需此修改在设备断电重启后仍然有效, 请调用GD\_MTC\_SDK\_saveCalcParm函数保存修改到设备.

**参数:**

iLoginID:

登入ID.

iHumidity:

湿度, 值的范围为0~100.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_setBackgroundTemp(const int iLoginID, float fBgTemp, int \*pUser);

**功能:**

设置背景温度.注意修改此参数后只对设备此次开机有效, 设备断电重启后此设置无效, 如需此修改在设备断电重启后仍然有效, 请调用GD\_MTC\_SDK\_saveCalcParm函数保存修改到设备.

**参数:**

iLoginID:

登入ID.

flfBgTempt:

背景温度. 单位摄氏度

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_BackgroundTempSwitch(const int iLoginID, int nEnable, int \*pUser);

**功能:**

背景温度开关.注意修改此参数后只对设备此次开机有效, 设备断电重启后此设置无效, 如需此修改在设备断电重启后仍然有效, 请调用GD\_MTC\_SDK\_saveCalcParm函数保存修改到设备.

**参数:**

iLoginID:

登入ID.

nEnable:

传入1表示开, 0表示关.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_setB2(const int iLoginID, float fB2, int \*pUser);

**功能:**

设置B2参数. 可通过设置B2参数以线性调整SDK输出的温度值, 例如将此参数增大1, SDK输出的温度值会增加1摄氏度.一般先通过GD\_MTC\_SDK\_getB2获取当前的B2数值, 再根据需要调整B2参数数值, 再调用此函数将新的B2参数设置下去.注意修改此参数后只对设备此次开机有效, 设备断电重启后此设置无效, 如需此修改在设备断电重启后仍然有效, 请调用GD\_MTC\_SDK\_saveCalcParm函数保存修改到设备.

**参数:**

iLoginID:

登入ID.

fB2:

B2参数的值.单位摄氏度.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_SetGearKf(const int iSessionID, int nKf);

**功能:**

设置KF参数. 注意修改此参数后只对设备此次开机有效, 设备断电重启后此设置无效, 如需此修改在设备断电重启后仍然有效, 请调用GD\_MTC\_SDK\_saveCalcParm函数保存修改到设备.

**参数:**

iSessionID:

数据流ID.

nKf:

KF参数的值.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_SetVskGear(const int iSessionID, int nGear);

**功能:**

设置当前的测温档位.

**参数:**

iSessionID:

数据流ID.

nGear:

当前的档位值, 目前支持1, 2.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_SetUartCheckAddress(const int iLoginID, int iValue);

**功能:**

设置UART校验地址.

**参数:**

iLoginID:

登入ID.

iValue:

UART校验地址.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_SetUartRate(const int iLoginID, int iValue);

**功能:**

设置UART传输速率.

**参数:**

iSessionID:

数据流ID.

iValue:

UART传输速率.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 保存测温参数

int32\_t GD\_MTC\_SDK\_saveCalcParm(const int iLoginID, int \*pUser);

**功能:**

将测温参数的修改保存到机芯设备. 测温相关参数成功保存到机芯设备后, 设备断电重启后相关参数的修改仍然有效. 保存过程需要2分钟, 请在调用此函数后的2分钟之内不要操作设备, 保持设备与电脑的连接状态同时不再调用SDK中的接口函数. 测温参数包括: 发射率, 距离, 湿度, 背景温度, 背景温度开关, B2参数, KF参数.

**参数:**

iLoginID:

登入ID.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_saveB2(const int iLoginID, int \*pUser);

**功能:**

保存B2参数. 将B2参数的修改保存到机芯设备.B2参数成功保存到机芯设备后, 设备断电重启后B2参数的修改仍然有效. 保存过程需要2分钟, 请在调用此函数后的2分钟之内不要操作设备, 保持设备与电脑的连接状态同时不再调用SDK中的接口函数.

**参数:**

iLoginID:

登入ID.

pUser:

用户指针.传入0值即可.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 获取实时点温

int32\_t GD\_MTC\_SDK\_GetPointTemp(void \* pOpque, GD\_MTC\_TempPointInfo \*pTempPoint);

**功能:**

获取点温.在自定义的Y16回调函数中调用此函数.

**参数:**

pOpque:

Y16回调函数中传入的对应值.

pTempPoint:

表示点和温度的对象地址.向此对象传入要获取点的坐标值,函数调用成功后读取此点的温度值.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 获取实时全图温

int32\_t GD\_MTC\_SDK\_GetTempMatrix(void \* pOpque, float \*pTempMatrix);

**功能:**

获取全图温.在自定义的Y16回调函数中调用此函数.

**参数:**

pOpque:

Y16回调函数中传入的对应值.

pTempMatrix:

传入用于接收温度数据的缓冲区地址,函数调用成功后读取对应的温度值.缓冲区中能够容纳的元素个数为图像的宽与高的乘积.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetHighTemp(void \* pOpque, GD\_MTC\_TempPointInfo \*pTempPoint);

**功能:**

获取全图最高温.在自定义的Y16回调函数中调用此函数.

**参数:**

pOpque:

Y16回调函数中传入的对应值.

pTempPoint:

表示点和温度的对象地址.函数调用成功后读取对应的坐标和温度值.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetLowTemp(void \* pOpque, GD\_MTC\_TempPointInfo \*pTempPoint);

**功能:**

获取全图最低温.在自定义的Y16回调函数中调用此函数.

**参数:**

pOpque:

Y16回调函数中传入的对应值.

pTempPoint:

表示点和温度的对象地址.函数调用成功后读取对应的坐标和温度值.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetAvegTemp(void \* pOpque, GD\_MTC\_TempPointInfo \*pTempPoint);

**功能:**

获取全图平均温.在自定义的Y16回调函数中调用此函数.

**参数:**

pOpque:

Y16回调函数中传入的对应值.

pTempPoint:

表示点和温度的对象地址.函数调用成功后读取对应的温度值.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 获取实时矩形温

int32\_t GD\_MTC\_SDK\_GetRectHighTemp(void \* pOpque, GD\_MTC\_RectangleInfo RecInfo, GD\_MTC\_TempPointInfo \*pTempPoint);

**功能:**

获取指定矩形区域的最高温.在自定义的Y16回调函数中调用此函数.

**参数:**

pOpque:

Y16回调函数中传入的对应值.

RecInfo:

表示一个矩形区域. 调用此函数前设置好此结构体中的各成员.

pTempPoint:

表示点和温度的对象地址.函数调用成功后读取对应的坐标和温度值.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetRectLowTemp(void \* pOpque, GD\_MTC\_RectangleInfo RecInfo, GD\_MTC\_TempPointInfo \*pTempPoint);

**功能:**

获取指定矩形区域的最低温.在自定义的Y16回调函数中调用此函数.

**参数:**

pOpque:

Y16回调函数中传入的对应值.

RecInfo:

表示一个矩形区域. 调用此函数前设置好此结构体中的各成员.

pTempPoint:

表示点和温度的对象地址.函数调用成功后读取对应的坐标和温度值.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetRectAvegTemp(void \* pOpque, GD\_MTC\_RectangleInfo RecInfo, GD\_MTC\_TempPointInfo \*pTempPoint);

**功能:**

获取指定矩形区域的平均温.在自定义的Y16回调函数中调用此函数.

**参数:**

pOpque:

Y16回调函数中传入的对应值.

RecInfo:

表示一个矩形区域. 调用此函数前设置好此结构体中的各成员.

pTempPoint:

表示点和温度的对象地址.函数调用成功后读取对应的温度值.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 获取图片温

int32\_t GD\_MTC\_SDK\_GetTempPictureDemension(const char\* filename, int \* pWidth, int \* pHeight);

**功能:**

获取温度图片的宽和高.

**参数:**

filename:

温度图片的路径.

pWidth:

传入用于接收温度图片的宽度的地址.

pHeight:

传入用于接收温度图片的高度的地址.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetPointTemperatureFromPicture(const char\* filename, int x, int y, float\* pointT);

**功能:**

获取温度图片中任意点的温度.

**参数:**

filename:

温度图片的路径.

x:

图片中的点相对图片左上角的水平坐标.

y:

图片中的点相对图片左上角的垂直坐标.

pointT:

传入用于接收温度图片中指定点温度值的地址.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetGlobalTemperatureFromPicture(const char\* filename, float\* imageT);

**功能:**

获取温度图片的温度矩阵.在调用此函数前可调用GD\_MTC\_SDK\_GetTempPictureDemension接口来获取温度图片的宽和高.

**参数:**

filename:

温度图片的路径.

imageT:

传入用于接收温度矩阵的地址.温度矩阵的字节长度是宽乘高个float的长度.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetRectTemperatureFromPicture(const char\* filename, int x, int y, int width, int height, POINT\_INFO\* maxInfo, POINT\_INFO\* minInfo, float\* avgT);

**功能:**

获取温度图片中指定矩形区域的最高温及坐标, 最低温及坐标, 平均温. 在调用此函数前可调用GD\_MTC\_SDK\_GetTempPictureDemension接口来获取温度图片的宽高.

**参数:**

filename:

温度图片的路径.

x:

指定矩形区域左上角相对图片左上角的水平坐标.

y:

指定矩形区域左上角相对图片左上角的垂直坐标.

width:

指定矩形区域的宽度.

height:

指定矩形区域的高度.

maxInfo:

传入用于接收指定矩形区域中最高温信息的结构体地址. 如果此函数调用成功,此结构体中包含对应点的坐标及温度值.

minInfo:

传入用于接收指定矩形区域中最低温信息的结构体地址. 如果此函数调用成功,此结构体中包含对应点的坐标及温度值.

avgT:

传入用于接收指定矩形区域平均温的地址..

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 添加矩形分析对象

int32\_t GD\_MTC\_SDK\_AddRectAnalyser(const int iLoginID, int x, int y, int width, int height, int type);

**功能:**

在视频流中添加矩形温度分析区域. 根据函数中的标记类型, 会标记显示矩形域内的最高温及最低温点. 目前仅部分机型在Y8数据流模式支持.

**参数:**

iLoginID:

登入ID.

x:

以视频图像左上角顶点为起始点, 矩形左上角顶点相对于此点的x值.

y:

以视频图像左上角顶点为起始点, 矩形左上角顶点相对于此点的y值.

width:

矩形的宽.

height:

矩形的高.

type:

标记类型. 如果标记类型中包含最高温或最低温, 则会在指定的矩形内用十字光标标记最高温或最低温点. 标记类型数值的具体含义如下:

0表示最高温,

1表示最低温,

2表示平均温,

3表示最高温+最低温,

4表示最高温+平均温,

5表示最低温+平均温,

6表示最高温+最低温+平均温,

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_DeleteRectAnalyser(const int iLoginID);

**功能:**

删除视频流中的矩形温度分析区域.

**参数:**

iLoginID:

登入ID.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_ChangeAnalyserTemperatureShowType(const int iLoginID, int type);

**功能:**

在视频流中添加矩形温度分析区域. 根据矩形分析区域的标记类型, 会标记显示域内的最高温及最低温点. 目前仅部分机型在Y8数据流模式支持.

**参数:**

iLoginID:

登入ID.

type:

标记类型. 如果标记类型中包含最高温或最低温或平均温, 则会在视频流的左下方显示最高温或最低温或平均温的温度数值. 标记类型数值的具体含义如下:

0表示最高温,

1表示最低温,

2表示平均温,

3表示最高温+最低温,

4表示最高温+平均温,

5表示最低温+平均温,

6表示最高温+最低温+平均温,

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_EnableRectAnalyserDigitalTempWithCursor(const int iLoginID);

**功能:**

在矩形分析对象中的光标旁边显示数字温度.

**参数:**

iLoginID:

登入ID.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_DisableRectAnalyserDigitalTempWithCursor(const int iLoginID);

**功能:**

在矩形分析对象中的光标旁边不显示数字温度.

**参数:**

iLoginID:

登入ID.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetRectAnalyserInfo(const int iLoginID, GD\_MTC\_RECT\_ANALYSER\_INFO \* pRectAnalyser)

**功能:**

获取矩形温度分析区域信息.

**参数:**

iLoginID:

登入ID.

pRectAnalyser:

矩形温度分析区域信息, 具体查看GD\_MTC\_RECT\_ANALYSER\_INFO说明.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

GD\_MTC\_RECT\_ANALYSER\_INFO结构体说明:

struct GD\_MTC\_RECT\_ANALYSER\_INFO

{

int nAnalyseMode = 0;

int nCursorMode = 0;

int x = 0;

int y = 0;

int nWidth = 0;

int nHeight = 0;

int nEnableDigitalTempWithCursor = 0;

};

**成员:**

nAnalyseMode :

分析模式.

0表示关闭;

1表示点分析;

2表示区域分析

nCursorMode :

光标模式.

0表示最高温;

1表示最低温;

2表示平均温;

3表示最高温和最低温;

4表示最高温和平均温;

5表示最低温和平均温;

6表示最高温和最低温和平均温

x :

矩形区域左上角水平坐标.

y :

矩形区域左上角垂直坐标.

nWidth :

矩形区域宽度.

nHeight :

矩形区域高度.

nEnableDigitalTempWithCursor :

是否在矩形分析对象中光标旁边显示数字温度.

## 视频流叠加字符串

int32\_t GD\_MTC\_SDK\_AddStringToStream(const int iLoginID, int x, int y, const void\* pStrData, int nStrByteLen, const char \* strEncodeName);

**功能:**

在指定的坐标处叠加指定的字符串到视频流中. 目前仅部分机型在Y8数据流模式支持.

**参数:**

iLoginID:

登入ID.

x:

以视频图像左上角顶点为起始点, 字符串矩形区域左上角顶点相对于此点的x值.

y:

以视频图像左上角顶点为起始点, 字符串矩形区域左上角顶点相对于此点的y值.

pStrData:

要叠加到视频流中的字符串数据的地址.

nStrByteLen:

字符串数据的字节长度.

strEncodeName:

字符串数据的编码方式, 在调用此接口时传入”utf16”. 目前此接口仅支持”utf16”方式编码的字符串.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

**备注：**

**接口变更(2018.06.18), 接口参数变化, 添加字符串原有接口不再支持**.

int32\_t GD\_MTC\_SDK\_EnableAddStringToStream(const int iLoginID);

**功能:**

启用字符串叠加功能. 目前仅部分机型在Y8数据流模式支持.

**参数:**

iLoginID:

登入ID.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_DisableAddStringToStream(const int iLoginID);

**功能:**

禁用字符串叠加功能. 目前仅部分机型在Y8数据流模式支持.

**参数:**

iLoginID:

登入ID.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetAddStringInfo(const int iLoginID, GD\_MTC\_ADD\_STRING\_INFO \* pAddStringInfo)

**功能:**

获取字符串叠加信息. 目前仅部分机型在Y8数据流模式支持.

**参数:**

iLoginID:

登入ID.

pAddStringInfo:

字符串叠加信息, 具体查看GD\_MTC\_ADD\_STRING\_INFO说明.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

GD\_MTC\_ADD\_STRING\_INFO结构体说明:

struct GD\_MTC\_ADD\_STRING\_INFO

{

int nEnabled;

int x;

int y;

unsigned char strData[128];

int nStrByteLen;

GD\_MTC\_ADD\_STRING\_INFO()

{

nEnabled = 0;

x = 0;

y = 0;

*memset*(strData, 0, sizeof(strData));

nStrByteLen = 0;

}

};

**成员:**

nEnabled:

当前是否启用字符叠加状态, 1表示开启, 0表示关闭.

x :

叠加字符串区域左上角的水平坐标.

y :

叠加字符串区域左上角的垂直坐标.

strData:

叠加的字符串数据, 数据是”utf16”编码.

nStrByteLen:

叠加的字符串数据字节长度.

## 电子变焦

int32\_t GD\_MTC\_SDK\_SetExLevel(const int iSessionID, *GD\_MTC\_EX* nExLevel, int nCenterX, int nCenterY);

**功能:**

设置温度原始数据电子变焦.

**参数:**

iSessionID:

数据流ID.

nExLevel:

电子变焦放大倍率.

GD\_MTC\_EX\_1表示原始图像;

GD\_MTC\_EX\_2表示2倍放大;

GD\_MTC\_EX\_4表示4倍放大;

nCenterX:

电子变焦中心点水平坐标.

nCenterY:

电子变焦中心点垂直坐标.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetExLevel(const int iSessionID, GD\_MTC\_EX\_INFO \* pExInfo)

**功能:**

获取电子变焦状态.

**参数:**

iSessionID:

数据流ID.

pExInfo:

电子变焦状态, 具体查看GD\_MTC\_EX\_INFO说明.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

GD\_MTC\_EX\_INFO说明

struct GD\_MTC\_EX\_INFO

{

GD\_MTC\_EX nExLevel;

int nCenterX;

int nCenterY;

};

**成员:**

nExLevel:

电子变焦放大倍率.

GD\_MTC\_EX\_1表示原始图像;

GD\_MTC\_EX\_2表示2倍放大;

GD\_MTC\_EX\_4表示4倍放大;

nCenterX:

电子变焦中心点水平坐标.

nCenterY:

电子变焦中心点垂直坐标.

## 调光模式

int32\_t GD\_MTC\_SDK\_SetAdjustLightMode(const int iSessionID, int nMode);

**功能:**

设置调光模式.只有部分数据流模式支持, 具体查看功能支持表.

**参数:**

iSessionID:

数据流ID.

nMode:

调光模式.目前调光模式有三种, 亮度对比度调光, 自动温度调光, 手动温度调光, 分别对应ADJUST\_LIGHT\_MODE\_BRIGHTNESS\_CONTRAST

ADJUST\_LIGHT\_MODE\_TEMPERATURE\_AUTO

ADJUST\_LIGHT\_MODE\_TEMPERATURE\_MANUAL

当为亮度对比度调光时调用GD\_MTC\_SDK\_SetBrightness和GD\_MTC\_SDK\_SetContrast接口设置亮度对比度改变调光效果.

当为自动温度调光时SDK自动根据当前温度改变调光效果,无需要手动设置其它参数.

当为手动温度温度调光时调用GD\_MTC\_SDK\_SetSpanMin和GD\_MTC\_SDK\_SetSpanMax接口指定温度高值和温度低值来改变调光效果.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 亮度对比度调光

int32\_t GD\_MTC\_SDK\_SetBrightness(const int iSessionID, int nValue);

**功能:**

设置图像的亮度. 只有部分数据流模式支持, 具体查看功能支持表.

**参数:**

iSessionID:

数据流ID.

nValue:

图像的亮度, 范围为1到100.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetBrightness(const int iSessionID, int \* pValue);

**功能:**

获取图像的亮度. 只有部分数据流模式支持, 具体查看功能支持表.

**参数:**

iSessionID:

数据流ID.

pValue:

接收图像亮度的地址. 亮度值的范围为1到100.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_SetContrast(const int iSessionID, int nValue);

**功能:**

设置图像的对比度. 只有部分数据流模式支持, 具体查看功能支持表.

**参数:**

iSessionID:

数据流ID.

nValue:

图像的对比度, 范围为1到100.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetContrast(const int iSessionID, int \* pValue);

**功能:**

获取图像的对比度. 只有部分数据流模式支持, 具体查看功能支持表.

**参数:**

iSessionID:

数据流ID.

pValue:

接收图像对比度的地址. 对比度值的范围为1到100.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 手动温度调光

int32\_t GD\_MTC\_SDK\_SetSpanMin(const int iSessionID, float nValue);

**功能:**

设置手动调光温度低值. 只有部分数据流模式支持, 具体查看功能支持表.

**参数:**

iSessionID:

数据流ID.

nValue:

手动调光温度低值.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetSpanMin(const int iSessionID, float \* pValue);

**功能:**

获取手动调光温度低值. 只有部分数据流模式支持, 具体查看功能支持表.

**参数:**

iSessionID:

数据流ID.

pValue:

接收温度低值的地址.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_SetSpanMax(const int iSessionID, float nValue);

**功能:**

设置手动调光温度高值. 只有部分数据流模式支持, 具体查看功能支持表.

**参数:**

iSessionID:

数据流ID.

nValue:

手动调光温度高值.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetSpanMax(const int iSessionID, float \* pValue);

**功能:**

获取手动调光温度高值. 只有部分数据流模式支持, 具体查看功能支持表.

**参数:**

iSessionID:

数据流ID.

pValue:

接收温度高值的地址.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

## 设置机芯设备温度报警

int32\_t GD\_MTC\_SDK\_GetDeviceAlarmState(const int iSessionID, int \* pState);

**功能:**

获取机芯设备温度报警的当前开关状态. 如果触发机芯设备温度报警, 不管是高温报警还是低温报警, 机芯设备后端对应针角会输出报警电平.

**参数:**

iSessionID:

数据流ID.

pState:

接收开关状态的地址. 值的含义具体如下:

0表示高温报警关, 低温报警关;

1表示高温报警开, 低温报警关;

2表示高温报警关, 低温报警开;

3表示高温报警开, 低温报警开;

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_SetDeviceAlarmState(const int iSessionID, int nState);

**功能:**

设置机芯设备温度报警的当前开关状态.

**参数:**

iSessionID:

数据流ID.

nState:

要设置的温度报警开关状态. 值的含义具体如下:

0表示高温报警关, 低温报警关;

1表示高温报警开, 低温报警关;

2表示高温报警关, 低温报警开;

3表示高温报警开, 低温报警开;

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetDeviceAlarmMax(const int iSessionID, float \* pTemp);

**功能:**

获取机芯设备温度报警的高温报警阈值.

**参数:**

iSessionID:

数据流ID.

pTemp:

接收高温报警阈值的地址.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_SetDeviceAlarmMax(const int iSessionID, float fTemp);

**功能:**

设置机芯设备温度报警的高温报警阈值.

**参数:**

iSessionID:

数据流ID.

fTemp:

要设置的高温报警阈值.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_GetDeviceAlarmMin(const int iSessionID, float \* pTemp);

**功能:**

获取机芯设备温度报警的低温报警阈值.

**参数:**

iSessionID:

数据流ID.

pTemp:

接收低温报警阈值的地址.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

int32\_t GD\_MTC\_SDK\_SetDeviceAlarmMin(const int iSessionID, float fTemp);

**功能:**

设置机芯设备温度报警的低温报警阈值.

**参数:**

iSessionID:

数据流ID.

fTemp:

要设置的低温报警阈值.

**返回值:**

NO\_ERROR值表示调用成功, 其它值表示调用失败.

# 其它

## 计算机配置性能建议

由于SDK在特定模式时需要对大量的温度信息进行运算, 因此对计算机的硬件性能有一定要求, 建议计算机达到或超过如下配置:

1. CPU: Intel i5, 第五代, 双核, 3.2GHz
2. 内存: 2G
3. 硬盘: 1T
4. 显卡: GTX960, 显存2G

## SDK使用注意事项

1. 目前SDK保存图片及视频时存储路径中不能包含中文字符, 路径中只能包含英文符号及数字.

# SDK更新

|  |  |  |
| --- | --- | --- |
| 更新内容 | SDK版本 | 更新时间及实施人 |
| 1.修复了海思平台切换到2档时Y16无图像的问题;  2.修复了平均温显示坐标的问题;  3.添加了修改IP时新IP和网关必须在同一网段的限制;  4.添加了修改IP时网关的二进制表示必须是前部分为1后部分为0的限制;  5.修复了停止AVI录像接口返回值异常的问题;  6.添加了对SDK不支持的功能直接返回失败的处理;  7.对超出范围的电机位置不作下发处理并记录错误日志; | 1.0.0.64 | 2018-08-16  付壮杰 |
| 1.IRGD录像文件大小限制修改为4G.  2.添加了修改IP时新IP不能与网关和DNS相同的限制.  3.SDK使用说明文档中添加了调用SDK的电脑建议配置.  4.修复了SDK生成的测温图片用红外分析软件无法生成报表的问题.  5.修复了先删除矩形分析对象再点添加时不显示最高温最低温的问题. | 1.0.0.67 | 2018-08-23  付壮杰 |
| 1.添加了微调焦接口.  2.添加了设置UART校验地址和传输速率的接口.  3.添加了对海思机型Y16帧有效性检查的处理.  4.添加了调焦和打快门互斥的处理.  5. OR417测温模型更新. | 1.0.0.68 | 2018-08-31  付壮杰 |
| 1.更新了SDK中ZU01A和ZU02A的Y16调光.  2.修复了Demo程序鼠标按下从视频区域向下拖出时程序崩溃的BUG.  3.修复了切换档次时Y16温度与Y8温度不一致的BUG.  4.添加了机芯刚上电时SDK去连接机芯获取设备信息失败的处理. | 1.0.0.69 | 2018-09-04  付壮杰 |
| 1.Linux64平台下的SDK从使用BOOST动态库更新为使用BOOST静态库. | 1.0.0.70 | 2018-09-07  付壮杰 |
| 1.ZU01A和ZU02A更新了获取设置B2参数接口.  2.ZU01A和ZU02A测温模型由OR417更新为IPT384,IPT640. | 1.0.0.71 | 2018-09-12  付壮杰 |
| 1.更新了NET800x600HS和NET640x480HS机芯解析曲线和参数行的逻辑, 保持与NETHS一致. | 1.0.0.72 | 2018-09-13  付壮杰 |
| 1.ZU01A,ZU02A,ZU01B温度计算更新. | 1.0.0.73 | 2018-09-14  付壮杰 |
| 1.Y16模式SDK绘制矩形,最高温最低温十字光标及数字温度完成.  2.Y16模式SDK内部可叠加色带功能. | 1.0.0.75 | 2018-09-27  付壮杰 |
| 1.ZU02A机型同时支持高温版和低温版.  2.修复了删除矩形分析对象时Y16中心温不正确的问题.  3.添加了获取UART地址和速率的接口.  4.添加了对ORHS机型的支持. | 1.0.0.77 | 2018-10-16  付壮杰 |
| 1.添加了irgd录像过程中不能设置帧率和色带的限制.  2.Demo界面上控件状态”近焦微调”和”远焦微调”状态根据机型功能支持自动更新.  3.添加了对静态修改IP时网关4段不能为0的限制.  4.修复了重复调用初始化和登入导致程序闪退的问题. | 1.0.0.78 | 2018-10-17  付壮杰 |
| 1.添加了对NET384x288HAOBOHS机型的支持.  2.更新了ORHS机型的测温模型. | 1.0.0.79 | 2018-10-26  付壮杰 |
| 1.修复了ORHS机型快门中点击关闭快门, 快门闭合后再点击关闭快门无效的问题.  2.修复了ORHS机型在Linux64平台下Y8模式连续拍照实时画面卡的问题.  3.对Demo中参数块中的参数添加了范围限制.  4.修复了”Demo中输入超长数字字符判定为有效”的BUG. | 1.0.0.80 | 2018-10-30  付壮杰 |
| 1.NET800x600机型参数行部分参数更新以缩小温度变化范围. | 1.0.0.81 | 2018-11-05  付壮杰 |
| 1.修复了NET800x600机型实时画面对动态场景反应慢的问题.  2.修复了NET800x600机型在Linux平台下不能断网/断电重连的问题.  3.修复了Linux32平台下设置发射率为0.01~0.04之间时读取到的值比下发的值小0.01的问题.  4.修复了Demo中手动温度调光时高值可比低值小的问题.  5.修复了ORHS机型切换色带后机芯断电再上电色带恢复为默认的问题. | 1.0.0.82 | 2018-11-8  付壮杰 |
| 1.修复了ORHS机型用MFCDemo切换色带后机芯断电再上电色带恢复为默认的问题.  2.修复了ORHS机型添加矩形分析对象机芯断电后没有保存的问题.  3.修复了ORHS机型设置自动快门时间间隔机芯断电后没有保存的问题. | 1.0.0.83 | 2018-11-12  付壮杰 |
| 1.添加了对ZU07A项目对应机芯的支持. | 1.0.0.84 | 2018-11-15  付壮杰 |
| 1.添加了Linux平台下亿嘉和SDK接口对应的Demo程序. | 1.0.0.85 | 2018-11-22  付壮杰 |
| 1.修复了Win32平台Demo程序和Linux平台Demo程序多次点击打开设备的异常. | 1.0.0.86 | 2018-11-23  付壮杰 |
| 1.修复了ZU02A海思机芯Y8与Y16温度相关较大的问题. | 1.0.0.87 | 2018-11-26  付壮杰 |
| 1.修复了在Linux平台下连OR机芯添加矩形分析对象失败的问题. | 1.0.0.88 | 2018-11-26  付壮杰 |
| 1.添加了SDK对ZU02C384x288HS机型的支持. | 1.0.0.90 | 2018-12-06  付壮杰 |
| 1.添加了对ZS02B640x480HS机型的支持. | 1.0.0.92 | 2018-12-07  付壮杰 |
| 1.添加了对NET640x480HAOBOHS机型的支持.  2.添加了控制显示光标跟随数字温度显示开关的接口, 仅部分机型支持. | 1.0.0.93 | 2018-12-14  付壮杰 |
| 1.NET640x480HAOBOHS机芯开启调焦功能.  2.NET640x480HAOBOHS机芯开启电子变倍功能. | 1.0.0.94 | 2018-12-15  付壮杰 |
| 1.ZU02A机型测温模型变更为IPT384B. | 1.0.0.95 | 2018-12-15  付壮杰 |

# SDK版本号

1.0.0.95