



TUCAM-API 开发指南



鑫图光电有限公司

保留所有的权利

目录

1. 使用前阅读	4
2. 简介	4
3. 概述	5
3.1 层结构	5
3.2 SDK 开发包文件夹简介	6
安装完开发包后文件夹如下内容:	6
3.3 原理	6
3.4 规则	7
3.4.1 句柄相关的内容	7
3.4.2 函数相关内容	7
3.4.3 性能相关内容	8
3.4.4 属性相关的内容	9
3.4.5 触发相关的内容	9
4. 编程指引	12
4.1 搭建编程环境	12
4.1.1 VS2013 开发环境配置	12
4.2 快速上手	15
4.2.1 起始和终止	16
4.2.2 属性获取和设置	18
4.2.3 属性获取和设置	20
4.2.4 内存管理	22
4.2.5 捕获控制	25
4.2.6 文件管理	26
4.2.7 扩展控制	30
4.2.8 其它	31
5. 参考	35
5.1 类型和常量	35
5.1.1 TUCAMRET 错误代码	35
5.1.2 TUCAM_IDINFO 产品信息代码	37

5.1.3 TUCAM_IDCAPA 性能代码	38
5.1.4 TUCAM_IDPROP 属性代码	41
5.1.5 TUCAM_IDPPROP 图像拼接代码	42
5.1.6 TUCAM_IDCROI 区域计算代码	42
5.1.7 TUCAM_CAPTURE_MODES 捕获模式代码	43
5.1.8 TUIMG_FORMATS 图像格式代码	43
5.1.9 TUREG_TYPE 寄存器类型代码	43
5.1.10 TUCAM_TRIGGER_EXP 触发曝光时间模式代码	44
5.1.11 TUCAM_TRIGGER_EDGE 激发边沿类型代码	44
5.1.12 TUCAM_TRIGGER_READOUTDIRRESET 触发读取输出复位代码	44
5.1.13 TUCAM_TRIGGER_READOUTDIR 触发输出方向代码	44
5.1.14 TUFRM_FORMATS 帧格式代码	45
5.1.15 TUGAIN_MODE 增益模式代码	45
5.1.16 TUCHN_SELECT 通道选择代码	45
5.1.17 TUCAM_OUTPUTTRG_PORT 触发输出端口代码	45
5.1.18 TUCAM_OUTPUTTRG_KIND 触发输出种类代码	45
5.1.19 TUCAM_OUTPUTTRG_EDGE 触发输出边沿代码	46
5.1.20 TUDRAW_MODE 图像绘制模式代码	46
5.1.21 TUREC_MODE 录制文件关键帧模式代码	46
5.1.22 TUPROC_TYPE 图像处理模式代码	46
5.1.23 TUSTITCH_MODE 图像处理拼接模式代码	46
5.1.24 TUFOCUS_STATUS 对焦状态代码	47
5.1.25 TUCAM_IDVPROP 厂商属性（仅限厂商使用）	47
5.2 结构体	48
5.2.1 初始化	48
5.2.2 打开相机	48
5.2.3 打开图像	49
5.2.4 相机信息	49
5.2.5 性能的属性	50
5.2.6 属性的属性	50
5.2.7 性能 / 属性值文本	51
5.2.8 厂商属性的属性	51

5.2.9 处理图像属性	52
5.2.10 ROI 的属性	52
5.2.11 区域计算属性	53
5.2.12 触发的属性	54
5.2.13 触发输出的属性	54
5.2.14 任意 Bin 属性	55
5.2.15 帧结构	55
5.2.16 文件保存	57
5.2.17 录像保存	57
5.2.18 寄存器读写	58
5.2.19 图像绘制初始化	58
5.2.20 图像绘制参数（仅 Windows）	59
5.3 函数	60
5.3.1 API 初始化 / 反初始化	60
5.3.2 打开、关闭相机	60
5.3.3 性能获取和设置	63
5.3.4 属性获取和设置	65
5.3.5 内存管理	68
5.3.6 捕获控制	73
5.3.7 文件管理	80
5.3.8 扩展控制	82
6. 常见问题解答（FAQ）	83
6.1 问题排查思路	83
6.2 常见问题解决方法	84

1. 使用前阅读

这份文档和软件示例代码是 TUCSEN 的内部文件和公布内容，以使用户能够创建使用 TUCSEN 数字相机中的应用。本文档和软件示例代码只针对上述目的而公开的，并且不构成所有者的许可、转让或任何其他权力。使用软件文档的所有风险和结果仍然取决于用户。

使用软件文档的所有风险和结果仍然取决于用户。本文档可能包括技术错误或印刷错误。并且不能保证这样的错误或文本所产生的任何损害。TUCSEN 不承诺更新或保持当前的这个文档中所包含的信息。

所有品牌和产品名称都是其各自所有者的商标或注册商标。TUCSEN 对文档的版权保留所有权利。在没有 TUCSEN 的事先书面许可下，文档的任何部分不得被复制、传递、转录，存储在检索系统或翻译成任何语言或计算机语言，以任何的形式，或以任何方式，任何的手段如：电子、机械、电磁、光学、化学手动或其他。

2. 简介

TUCAM-API

TUCAM-API 是用于控制 TUCSEN 制造的相机的应用软件程序接口。TUCAM-API 目前支持连接 USB 、 CameraLink 、 CoaXPress 、 Gige 数据接口，在初始化时自动进行匹配工作。另外 API 设计特别容易理解。出于这个原因，函数接口的数量限制到最少，并且函数的调用格式采用 C 语言的写法。

SDK

TUCAM-API 软件开发套件（以下简称“SDK”）为开发者制作主机软件程序提供资源。 SDK 由头文件、库文件、 TUCAM-API 函数参考文件、系列相机性能属性参考文件和示例代码组成。例如部分修改源代码和创建完全独立的程序，和/或将主机软件或插件模块提供给所有人。

说明

部分扩展的函数是某些特定数字相机可以使用的附加功能。不同数字相机的数值可能不同，这取决于捕捉图像所使用的数字相机型号。数值应该简单地视为指南，而不是精确值。具体可以参考系列相机性能、属性文件。

3. 概述

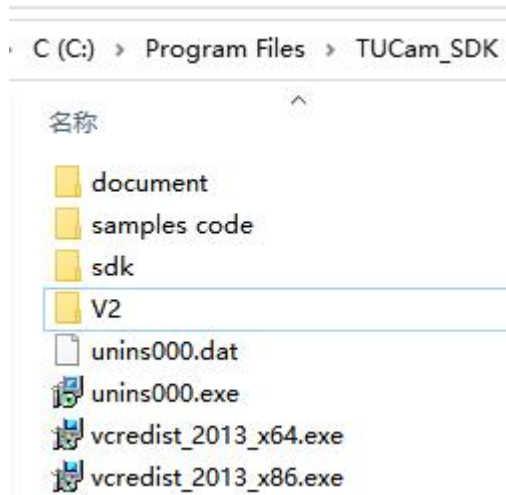
3.1 层结构



TUCSEN 数字相机通过 SDK 连接不同的操作系统的数字相机驱动来达到控制数字相机和采集图像数据的作用。

3.2 SDK 开发包文件夹简介

安装完开发包后文件夹如下内容：



document 文件夹是 SDK 接口开发文档说明书(/document/CH/TUCAM_API 开发指南.pdf)和属性列表手册(/document/CH/xxxx 系列属性性能说明.pdf)帮助客户了解相机二次开发和相机功能。

samples code 文件夹是示例代码包含 C#、C++、Python(/samples code/console/), 还包含带 GUI 界面示例代码 C#、MFC、Qt 帮助客户提升开发效率(/samples code/win/).

sdk 文件夹包含了可被调用二次开发的头文件(/sdk/inc/)和连接的动态库帮助客户及时找到开发库的文件(/sdk/lib)。其中 All_x64 文件夹里面 TUCam.dll 是支持所有协议相机, x64 和 x86 文件夹里面 TUCam.dll 仅支持 USB 和 Cameralink 相机。

V2 文件夹是升级补充了 C#、Python 开发文档说明书(/v2/Document/)和包含 C#、C++、Python、Qt(/V2/CodeSamples/Console/)示例代码, 以及带 GUI 界面示例代码有 MFC、Qt、C#(/V2/CodeSamples/GUI/)帮助客户有更多更丰富的示例做参考。

.Net 支持版本: .Net Framework3.5。

Linux 支持版本: Ubuntu 18.04LT 内核版本: 5.4.0-150-generic。

3.3 原理

数字相机的具体总线接口和库通过 TUCAM-API 封装。您只需要访问 TUCAM-API 层。模块层提供更高級的 TUCAM-API 集成。模块可以不断更新访问新相机和提供新接口技术, 而无需重新编译您的软件。

TUCAM-API 不包含用于显示图像的程序。因为一些显示图像的方法难以预测，这取决于应用程序，它是不可能支持所有这些通用模块的。当调用显示程序时，检测图像是否更新，并且在更新后绘制图像。对于更详细的信息，请参阅示例源代码。

3.4 规则

3.4.1 句柄相关的内容

名称	描述
HDTUCAM	是 API 上指定目标相机的句柄，几乎所有的 API 都需要这个句柄作为参数。TUCAM_Dev_Open 函数提供这个句柄，利用 TUCAM_Dev_Release 函数释放句柄，终止设备使用。
HDTUIMG	是 API 指定打开图片的句柄。TUIMG_File_Open 函数提供这个句柄，利用 TUCAM_File_Close 关闭句柄，释放图片资源。

3.4.2 函数相关内容

函数名称

所有 TUCAM-API 函数都以大写字母“TUCAM_”作为前缀。TUCAM-API 所有函数按照性能被归类，同组的函数有相同前缀。以下是函数前缀列表：

函数前缀	函数名称	说明
TUCAM_Api_	Init, Uninit	初始化、反初始化 API
TUCAM_Dev_	Open, Close, GetInfo, GetInfoEx	打开、关闭相机、获取信息
TUCAM_Capa_	GetAttr, GetValue, SetValue, GetValueText	性能获取和设置
TUCAM_Prop_	GetAttr, GetValue, SetValue, GetValueText	属性获取和设置
TUCAM_Buf_	Alloc, Release, AbortWait, WaitForFrame, CopyFrame	内存管理相关控制
TUCAM_Cap_	SetROI, GetROI, SetTrigge, GetTrigger, DoSoftwareTrigger, SetTriggerOut, GetTriggerOut, Start, Stop	图像数据捕获相关控制
TUCAM_File_	SaveImage, LoadProfiles, SaveProfiles, Open, Close	图片文件、配置文件相关操作
TUCAM_Rec_	Start, AppendFrame, Stop	视频文件保存相关接口
TUCAM_Reg_	Read, Write	相机相关配置寄存器读/写
TUCAM_Draw_	Init, Frame, UnInit	绘制图像接口（Windows 系统）
TUCAM_Calc_	SetROI, GetROI	相关功能 ROI 计算的区域读/写

TUCAM_Vendor_	Config, ConfigEx, Update, Prop_GetAttr Prop_SetValue, Prop_GetValue, Prop_GetValueText, ResetIndexFrame, WaitForIndexFrame, Buf_Attach, Buf_Detach	厂商控制的相关接口
---------------	--	-----------

函数参数

除了 TUCAM_Api_Init 初始化、TUCAM_Api_Uninit 反初始化、TUCAM_Dev_Open 打开相机等函数，几乎所有的 TUCAM-API 都需要 HDTUCAM 句柄作为第一个参数。一些函数还需要一个指向结构体的指针作为参数，在结构体参数中有 [in] 标识变量，意味着应用程序必须在调用之前设置值，有 [out] 标识变量，意味着函数返回时填充一个值。

函数返回

TUCAM-API 所有的函数执行后都有返回一个 TUCAMRET 值。所有的返回结果都被定义成 TUCAMRET_ 开头的。如果函数执行成功返回 TUCAMRET_SUCCESS，在开发中建议您在执行完函数后，判断返回结果值为 TUCAMRET_SUCCESS 再进行下一步操作，否则给出错误提醒，方便问题的查找。

3.4.3 性能相关内容

在 TUCAM 中什么是性能？

“性能”是指设备支持能力。TUCAM-API 中定义了 TUCAM_IDCAPA 性能常量，每个性能都有个唯一 ID 以 TUIDC_ 为前缀。用户通过带有 TUCAM_CAPA_ 前缀的函数查询设备是否支持某项能力，对性能值进行获取、设置。每个性能值都被处理成整型，每个性能都有自己 ID，最小值，最大值，默认值。

多性能值

一些性能可能有多个值，它们的值有关联的文本，我们称这样的文本为值文本。通过 TUCAM_Capa_GetValueText 函数获取值文本。

3.4.4 属性相关的内容

在 TUCAM 中什么是属性？

“属性”是指设备参数。TUCAM-API 中定义了 TUCAM_IDPROP 常量，每个属性都有个唯一 ID 以 TUIDP_ 为前缀。用户通过 TUCAM_Prop_ 前缀的函数获取、设置、查询设备参数。每个属性值都被处理成双精度浮点型，每个属性都有自己 ID, 最小值，最大值，默认值。

多属性值

一些属性可能有多个值，它们的值也有关联的文本，我们称这样的文本为值文本。通过 TUCAM_Prop_GetValueText 函数获取。

3.4.5 触发相关的内容

捕获模式：

相机的捕获模式分为以下 2 类：

- 1) 序列模式（流模式）：用来捕获连续的图像数据。
- 2) 触发模式：相机通过内、外部信号来捕获图像。我们称这个选项为“触发模式”，您可以调用 TUCAM_Cap_SetTrigger 来配置此选项。我们也把外部信号称为“外部触发”。

图像单元：

通常情况下是二维的，具有垂直和水平方向。

帧：是一个用于图像数据的单位。对于一帧，一个像素的数据是从左到右和从上到下对齐的。这是一系列的图像数据单位。

触发属性：

触发属性设置，参考 TUCAM_TRIGGER_ATTR 结构体。包括触发模式、曝光模式、激发电平类型、触发延迟、输出帧数。不同触发模式支持的参数不一样，例如软件触发，就不支持设置边沿激发类型，触发曝光时间由软件设定（软触发没有上升沿、下降沿）。

模式	触发曝光模式 (TUCAM_TRIGGER_EXP)	边沿激发类型 (TUCAM_TRIGGER_EDGE)	延时	帧数
标准触发 (Overlap/Non-Overlap)	支持	支持	支持	支持
同步触发	支持	支持	不支持	不支持
全局触发	不支持	支持	不支持	不支持
软件触发	不支持	不支持	不支持	不支持

触发模式

- 1) 标准模式 (Standard/Non-Overlap)：当相机接收到电平信号后（由激活边沿决定）开始进行一帧或多帧的图像捕获，捕获帧数由配置参数决定。
- 2) 同步模式 (Synchronization)：当相机接收到电平信号后（由激活边沿决定）开始进行曝光，当收到相反的电平信号后，结束曝光、并且进行图像数据的捕获。即实现每一帧的曝光与读出，均与外触发信号完全同步。
- 3) 全局模式 (Global)：在相机未触发前进行预触发。当相机接收到电平信号后（由激活边沿决定）或者为软件设定的曝光时间时，结束当前正在进行的重置操作，待曝光结束时捕获图像数据，并重新开始预触发。该方式用于控制卷帘曝光模式的相机实现全局曝光模式。
- 4) 软件触发 (Software)：通过软件下发命令模拟触发信号。

曝光模式

- 1) 曝光时间：接收到触发信号后，由 TUIDP_EXPOSURETM 设置的曝光时间决定。
- 2) 电平宽度：接收到触发信号后，曝光时间由电平的宽度所决定。

注：标准模式 (Standard) 和全局模式 (Global) 可配置这两个选项。同步模式 (Synchronization) 只能是电平宽度。软件触发只能由软件设置的曝光时间决定。

激发电平类型

- 1) 上升电平 (Rising Edge)：接收到的触发电平处于上升沿时开始曝光。

- 2) 下降电平（Falling Edge）：接收到的触发电平处于下降沿时开始曝光。

触发延迟

- 1) 输入：表示接收到一个触发信号后，可以设置多长的延迟时间才使相机进行触发曝光。
- 2) 输出：表示接收到一个输出信号后，可以设置多长的延迟时间才使相机电平信号输出。

触发输出端口

触发输出信号有 3 个端口 TRIG.OUT1、TRIG.OUT2、TRIG.OUT3，对应 Port1、Port2、Port3。

三种输出信号默认是始终开启状态，相机将电平信号输出给第三方设备作为其输入信号。三种信号可以独立工作、同时输出到不同设备。

触发输出硬件接口有 3 个管脚，可以对 3 个端口进行配置，并且 3 个端口互不干涉。

触发输出电平

- 1) High：始终以高电平信号输出。
- 2) Low：始终以低电平信号输出。
- 3) Trigger Ready：以相机处于开流状态且可以即时响应外触发信号时，输出高电平。
- 4) Exposure Start：以第一行开始曝光到最后一行开始曝光的指示电平为信号输出。
- 5) Readout End：以第一行开始读出到最后一行开始读出的电平信号输出。
- 6) Global Exposure：以最后一行开始曝光至第一行结束曝光（所有行处于曝光状态）的指示电平信号输出。

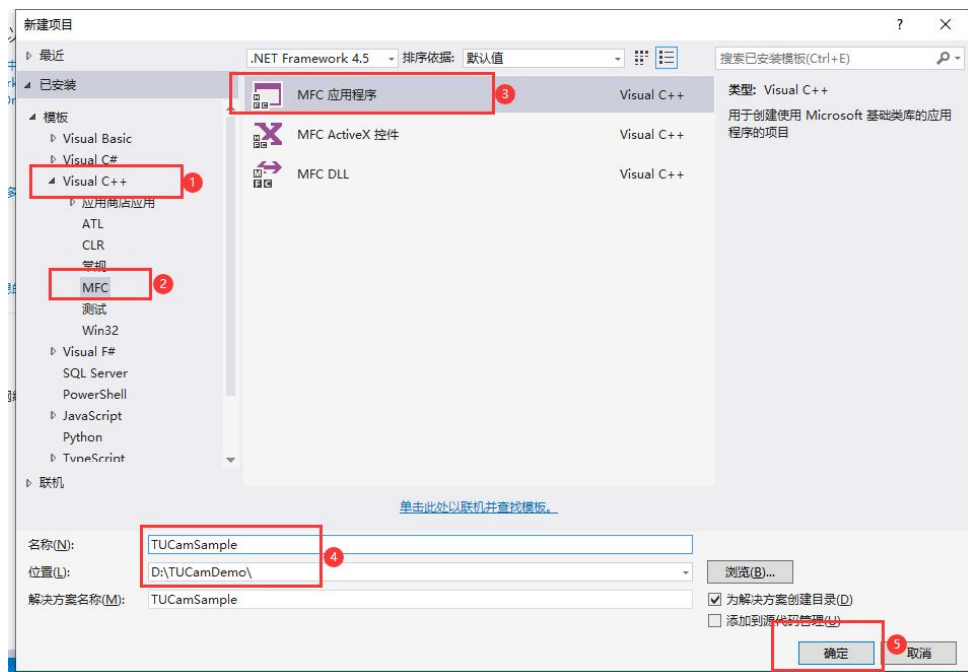
4. 编程指引

4.1 搭建编程环境

4.1.1 VS2013 开发环境配置

1) 新建工程

新建工程选择 MFC，点击“确定”后，对话框中点击“下一步”：





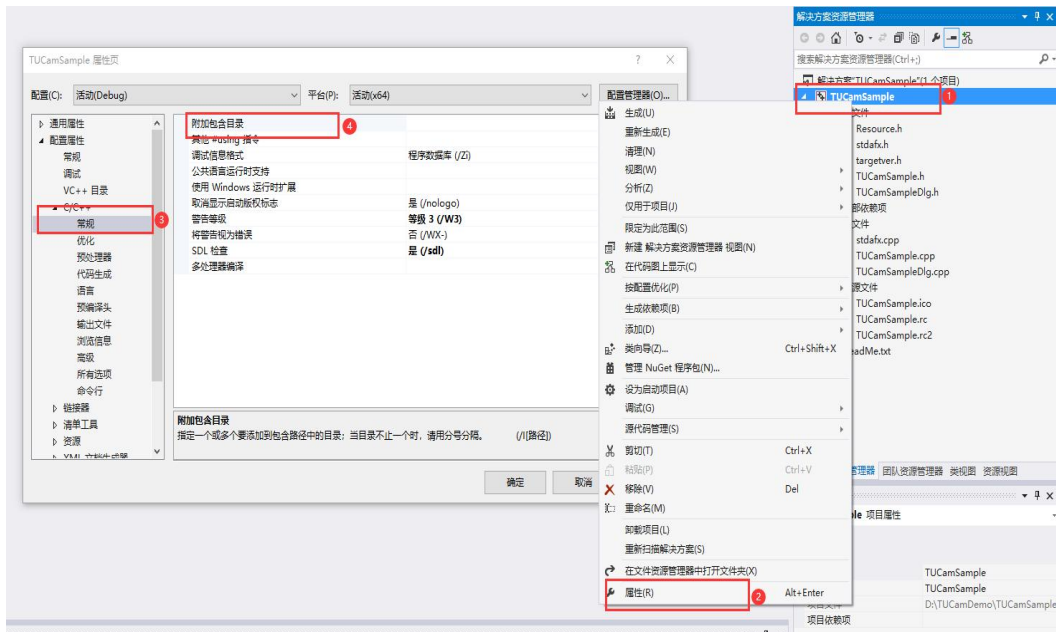
2) 配置引用

在弹出 MFC 应用程序向导对话框中选择“基于对话框”, 点击完成。就在指定的目录中创建了 MFC 程序工程。



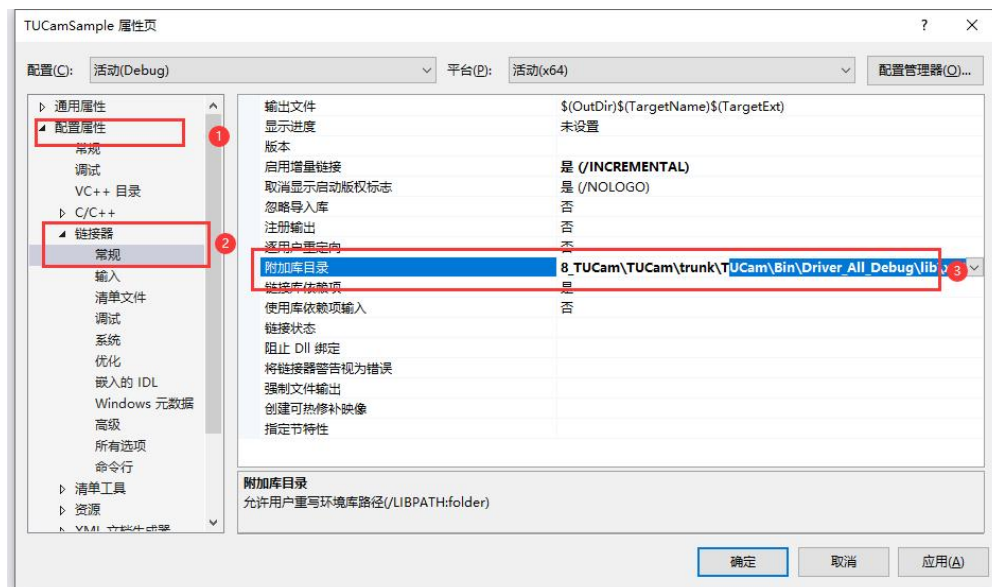
3) 配置引用头文件

在解决方案资源管理器窗口选中刚创建的工程，右键选中菜单中的“属性”选项，弹出属性窗口。选择 配置属性->C/C++->常规 在“附加包含目录”中填写 TUCamApi.h 所在目录路径地址（依用户安装目录为准），如图所示：

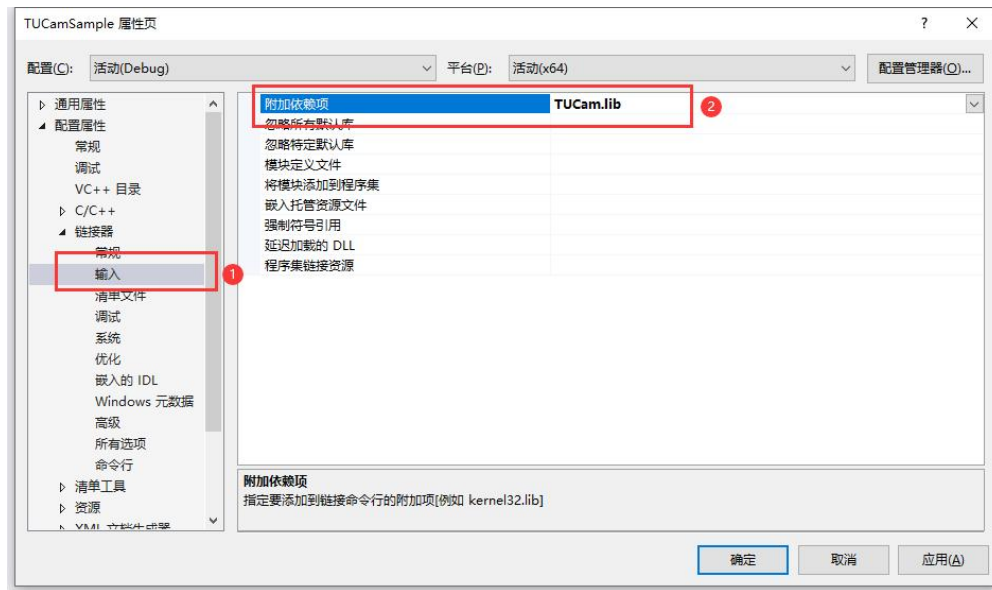


4) 配置 lib 文件

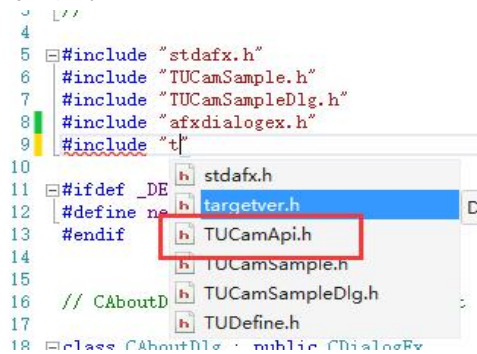
然后选择 配置属性->链接器->常规，在 Additional Library Directories 中 填写 TUCam.lib 所在目录路径地址（依用户安装目录为准），如图所示：



然后选择配置属性->链接器->输入在“附加依赖项”中填写 TUCam.lib，如图所示：



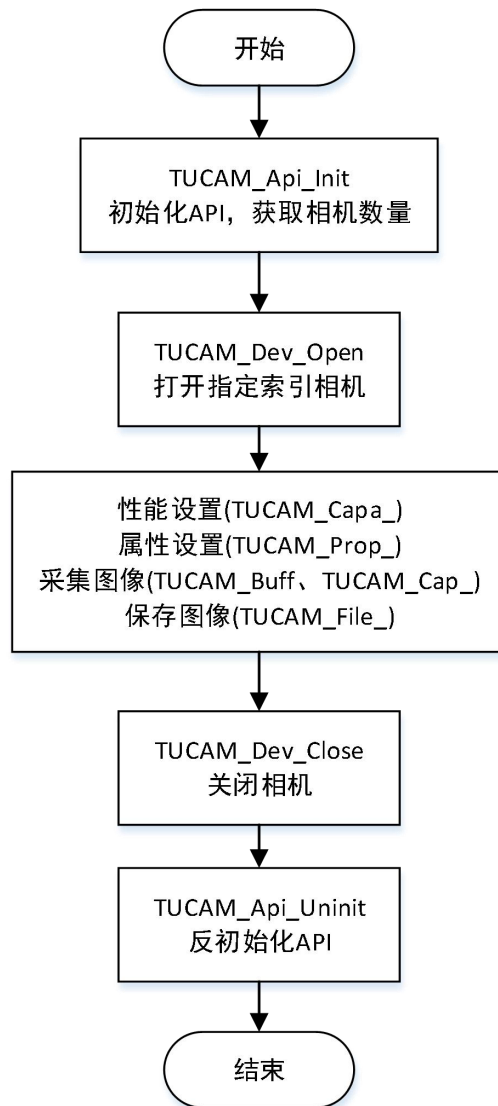
引用头部文件，有如下图提示：



注意：根据用户应用选择适合（x86、x64）引用。

4.2 快速上手

TUCAM-API 工作调用流程：



4.2.1 起始和终止

4.2.1.1 调用顺序

首先，驱动程序初始化。当应用程序执行设备句柄的初始化已经成功完成，获取可以控制的相机数量。

其次，当应用程序启动时，调用 SDK 的 API 初始化函数执行初始化操作。

再次，当初始化函数调用成功后，其他函数才能正常被调用执行。

相机终止函数用于程序的关闭。当一个相机被挂起，或资源被释放而不再控制相机时执行的函数（例如，当应用程序退出）。当终止函数被调用时，其他的功能函数调用将不被执行，直到初始化函数再次调用之后。

4.2.1.2 SDK 的 API 初始化

API 使用 TUCAM_Api_Init 函数进行初始操作。该函数初始化帧采集和控制相机。
[接口参考 5.3.1](#)

4.2.1.3 相机初始化

相机初始化使用 TUCAM_Dev_Open 函数。该函数获取必要的相机句柄 HDTUCAM 来作为其他函数的输入参数。[接口参考 5.3.2](#)

4.2.1.4 相机产品信息

当调用 TUCAM_Dev_Open 函数打开相机之后，可以通过相机句柄获取相机的产品信息。例如，相机型号、固件版本等。[接口参考 5.3.2](#)，产品信息参考 [TUCAM_IDINFO](#)

4.2.1.5 终止程序

终止相机程序使用 TUCAM_Dev_Close 函数。调用这个函数释放被用于相机帧获取的端口及资源。这个函数被调用后，相机将不再被控制。

4.2.1.6 示例代码

SDK 的 API 初始化参考以下代码，或者参考 Samples 目录 init_open 工程，相机产品信息参考 Samples 目录 get_info/get_infoEx 工程。

```
1. int main (int argc, char** argv)
2. {
3.     TUCAM_INIT  itApi; // 初始化 SDK 环境参数
4.     TUCMA_OPEN  opCam; // 打开相机参数
5.
6.     itApi.pstrConfigPath = NULL;
7.     itApi.uiCamCount = 0;
8.     if (TUCAMRET_SUCCESS != TUCAM_Api_Init(&itApi))
9.     {
10.        // 初始化 SDK API 环境失败
11.        return 0;
12.    }
13.
```

```
14.     if (0 == itApi.uiCamCount)
15.     {
16.         // 没有相机
17.         return 0;
18.     }
19.
20.     opCam.hIdxTUCam = 0;
21.     opCam.uiIdxOpen = 0;
22.
23.     if (TUCAMRET_SUCCESS != TUCAM_Dev_Open(&opCam))
24.     {
25.         // 打开相机失败
26.         return 0;
27.     }
28.
29.     // 应用程序可以使用 opCam.hIdxTUCam 句柄
30.
31.     TUCAM_Dev_Close(opCam.hIdxTUCam);        // 关闭相机
32.     TUCAM_Api_Uninit();                        // 反初始化 SDK API 环境
33.
34.     return 0;
35. }
```

4.2.2 属性获取和设置

4.2.2.1 调用顺序

1) 获取性能属性

利用 TUCAM_Capa_GetAttr 函数，通过设置性能 ID 获取性能属性，其中包括最大值、最小值、默认值。

2) 获取、设置性能值

通过 TUCAM_Capa_GetValue 和 TUCAM_Capa_SetValue 函数来设置和获取性能值。所有的相机性能值都被处理成一个整型，某些性能具有多个性能值。设置和获取通常在相机进行捕获之前或者之后已经完成。如果设置函数在数据捕获时被调用，在某些情况下可能会返回错误的代码 TUCAMRET 。[接口参考 5.3.3](#)

3) 获取性能值文本

TUCAM_Capa_GetValueText 函数获取性能值文本，用于描述每个性能值具体含义。

4.2.2.2 性能索引

每个性能都有唯一的 ID，参考 [TUCAM_IDCAPA](#)

注意：如果相机不支持该性能 ID，将返回错误代码 TUCAMRET_NOT_SUPPORT 或 TUCAMRET_INVALID_IDCAPA。

4.2.2.3 示例代码

参考以下代码或 Samples 目录 capa_list 工程：

```
1. // 以分辨率 TUIDC_RESOLUTION 为例
2. // 获取分辨率范围
3. void GetResolutionRange()
4. {
5.     TUCAM_CAPA_ATTR attrCapa;
6.     TUCAM_VALUE_TEXT valText;
7.
8.     char szRes[64] = {0};
9.     valText.nTextSize = 64;
10.    valText.pText = &szRes[0];
11.    attrCapa.idCapa = TUIDC_RESOLUTION;
12.    if (TUCAMRET_SUCCESS == TUCAM_Capa_GetAttr(opCam.hIdxTUCam, &attrCapa))
13.    {
14.        // 获取分辨率个数
15.        int nCnt = attrCapa.nValMax - attrCapa.nValMin + 1;
16.        valText.nID = TUIDC_RESOLUTION;
17.
18.        for (int i=0; i<nCnt; ++i)
19.        {
20.            valText.dbValue = i;
21.            TUCAM_Capa_GetValueText(opCam.hIdxTUCam, &valText);
22.            szRes = valText.pText;
23.            // 将分辨率文本添加到下拉菜单
24.        }
```

```
25.     }
26. }
27.
28. // 获取当前分辨率
29. void GetCurrentResolution()
30. {
31.     int nVal = 0;
32.
33.     if (TUCAMRET_SUCCESS == TUCAM_Capa_GetValue(opCam.hIdxTUCam, \
34.                                                  TUIDC_RESOLUTION, \
35.                                                  &nVal))
36.     {
37.         // nVal 返回当前分辨率索引
38.     }
39. }
40.
41. // 设置当前分辨率
42. void SetCurrentResolution(int nIdxRes)
43. {
44.     TUCAM_Capa_SetValue(opCam.hIdxTUCam, TUIDC_RESOLUTION, nIdxRes);
45. }
```

4.2.3 属性获取和设置

4.2.3.1 调用顺序

1) 获取属性

TUCAM_Prop_GetAttr 函数，通过设置属性 ID 获取属性属性，其中包括最大值、最小值、默认值。

2) 查询、设置属性值

通过 TUCAM_Prop_GetValue 和 TUCAM_Prop_SetValue 函数来设置和获取属性值。所有的相机属性值都被处理成一个双精度浮点型，某些属性具有多个属性值。设置和获取属性通常在相机进行捕获之前或者之后已经完成。如果设置函数在数据捕获时被调用，在某些情况下可能会返回错误的代码 TUCAMRET 。[接口参考 5.3.4](#)

3) 获取性能值文本

TUCAM_Prop_GetValueText 函数获取值文本，用于描述每个属性值具体含义。

4.2.3.2 属性索引

每个属性都有唯一的 ID，参考 [TUCAM_IDPROP](#)

注意：如果相机不支持该属性 ID，将返回错误代码 TUCAMRET_NOT_SUPPORT 或 TUCAMRET_INVALID_IDPROP。

4.2.3.3 示例代码

参考以下代码或 Samples 目录 prop_list 工程：

```
1. // 以曝光时间为例
2. // 获取曝光时间范围
3. void GetExposureTimeRange()
4. {
5.     TUCAM_PROP_ATTR attrProp;
6.
7.     attrProp.nIdxChn = 0;    // 当前通道
8.     attrProp.idProp = TUIDP_EXPOSURETM;
9.
10.    if (TUCAMRET_SUCCESS == TUCAM_Prop_GetAttr(opCam.hIdxTUCam, &attrProp))
11.    {
12.        // 曝光时间范围
13.        attrProp.dbValMin;    // 最小曝光时间
14.        attrProp.dbValMax;    // 最大曝光时间
15.        attrProp.dbValDft;    // 默认曝光时间
16.        attrProp.dbValStep;   // 曝光时间步长
17.    }
18. }
19.
20. // 获取当前曝光时间
21. void GetCurrentExposureTime()
22. {
23.     double dbVal = 1.0f;
24.
25.     if (TUCAMRET_SUCCESS == TUCAM_Prop_GetValue(opCam.hIdxTUCam, \
26.                                                    TUIDP_EXPOSURETM, \
```

```
27.                                     &dbVal))
28.     {
29.         // dbVal 返回当前曝光时间, 单位 ms
30.     }
31. }
32.
33. // 设置当前曝光时间
34. void SetCurrentExposureTime(double dbVal)
35. {
36.     TUCAM_Prop_SetValue(opCam.hIdxTUCam, TUIDP_EXPOSURETM, dbVal);
37. }
```

4.2.4 内存管理

4.2.4.1 调用顺序

1) 内存的分配

内存的分配 TUCAM_Buf_Alloc 必须在 TUCAM_Cap_Start 数据开始捕获之前调用, 且切换不同的分辨率都必须重新进行内存分配。

2) 数据的获取

必须在 TUCAM_Cap_Start 数据开始捕获之后调用 TUCAM_Buf_WaitForFrame

等待数据捕获的完成, 并且可以通过 TUCAM_Buf_CopyFrame 拷贝当前帧的不同格式数据, 一旦再次调用 TUCAM_Buf_WaitForFrame, TUCAM_Buf_CopyFrame 拷贝也会更新为下一帧数据。

3) 结束等待

如有进行数据等待和数据拷贝的调用, 在停止数据捕获之前调用 TUCAM_Buf_AbortWait 结束数据等待, 之后再调用 TUCAM_Cap_Stop 停止数据捕获。

4) 内存释放

内存的释放 TUCAM_Buf_Release 必须在 TUCAM_Cap_Stop 停止数据捕获之后调用。

4.2.4.2 帧结构体

用于描述一帧图像数据的结构体。参考 [TUCAM_FRAME](#)。变量 pBuffer 是指向帧数据指针，帧数据包括帧头部数据、帧图像数据两部分。将帧数据偏移 usHeader 字节获取到帧图像数据。

4.2.4.3 示例代码

参考以下代码或 Samples 目录 wait_frame 工程：

```
1. TUCAM_FRAME m_frame;           // 帧对象
2. HANDLE m_hThdGrab;             // 取图线程事件句柄
3. BOOL m_bLiving;                // 是否取图
4.
5. BOOL CDlgTUCam::StartCapture()
6. {
7.     m_frame.pBuffer = NULL;
8.     m_frame.ucFormatGet = TUFRM_FMT_RGB888; // 帧数据格式 (RGB888)
9.     m_frame.uiRsdSize = 1;        // 一次捕获帧数 (预留参数, 当前只能设置 1)
10.
11.     if (TUCAMRET_SUCCESS != TUCAM_Buf_Alloc(m_opCam.hIdxTUCam, &m_frame))
12.     {
13.         return FALSE;
14.     }
15.
16.     if (TUCAMRET_SUCCESS != TUCAM_Cap_Start(m_opCam.hIdxCam,
17.         TUCCM_SEQUENCE))
18.     {
19.         TUCAM_Buf_Release(m_opCam.hIdxTUCam);
20.         return FALSE;
21.     }
22.     m_bLiving = TRUE;
23.     m_hThdGrab = CreateEvent(NULL, TRUE, FALSE, NULL);
24.     _beginthread(GrabThread, 0, this);
25.
26.     return TRUE;
```



```
27. }
28.
29. Void __cdecl CDlgTUCam::GrabThread(LPVOID IParam)
30. {
31.     CDlgTUCam *pTuCam = (CDlgTUCam *)IParam;
32.
33.     While (pTUCam->m_bLiving)
34.     {
35.         pTUCam->m_frame.ucFormatGet = TUFRM_FMT_RGB888;
36.         if(TUCAMRET_SUCCESS ==
TUCAM_Buf_WaitForFrame(pTUCam->m_opCam.hIdxTUCam,\
37.                                     &pTUCam->m_frame))
38.         {
39.             // pTUCam->m_frame.pBuffer 返回捕获的图像数据，格式为
TUFRM_FMT_RGB88
40.             // 该数据可以用于显示
41.
42.             TUCAM_IMG_HEADER    frmhead;
43.             memcpy(&frmhead, pIn->m_frame.pBuffer, sizeof(TUCAM_IMG_HEADER));
44.             // 该数据可获取头部信息
45.
46.             // 可获取其他格式数据
47.             pTUCam->m_frame.ucFormatGet = TUFRM_FMT_USUAL;
48.
49.             if(TUCAMRET_SUCCESS==TUCAM_Buf_CopyFrame(pTUCam->m_opCam.hIdxTUCam,\
&pTUCam->m_frame))
50.             {
51.                 // pTUCam->m_frame.pBuffer 返回捕获的图像数据
52.             }
53.         }
54.     }
55.
56.     SetEvent(pTUCam->m_hThdGrab);
57.     _endthread();
58. }
59.
60. void CDlgTUCam::StopCapture()
61. {
62.     m_bLiving = FALSE;
63.     TUCAM_BUF_AbortWait();    // 如果调用 TUCAM_Buf_WaitForFrame 接口
64.
65.     WaitForSingleObject(m_hThdGrab, INFINITE);    // 等待取图线程退出
```

```
66.    CloseHandle(m_hThdGrab);
67.    m_hThdGrab = NULL;
68.
69.    TUCAM_Cap_Stop(m_opCam.hIdxTUCam);        // 停止数据捕获
70.    TUCAM_Buf_Release(m_opCam.hIdxTUCam);      // 释放分配的内存
71. }
```

4.2.5 捕获控制

4.2.5.1 调用顺序

设置 ROI 属性 TUCAM_Cap_SetROI 和触发属性 TUCAM_Cap_SetTrigger，需要在开始捕获数据之前调用，如果在数据捕获时调用可能返回 TUCAMRET 错误代码。

4.2.5.2 捕获模式索引

参考 [TUCAM_CAPTURE_MODES](#)

4.2.5.3 示例代码

参考以下代码或 Samples 目录 roi_mode 工程：

```
1.  // 设置 ROI 模式
2.  void SetROIMode()
3.  {
4.      TUCAM_ROI_ATTR roiAttr;
5.      roiAttr.bEnable = TRUE;
6.      roiAttr.nVOffset= 100;
7.      roiAttr.nHOffset = 100;
8.      roiAttr.nWidth   = 800;
9.      roiAttr.nHeight  = 600;
10.
11.     TUCAM_Cap_SetROI(m_opCam.hIdxTUCam, roiAttr);
12.     TUCAM_Cap_Start(m_opCam.hIdxCam, TUCAM_SEQUENCE); // 序列模式（即流
    模式）
13.
14.     // 数据获取参考内存管理示例代码
15. }
```

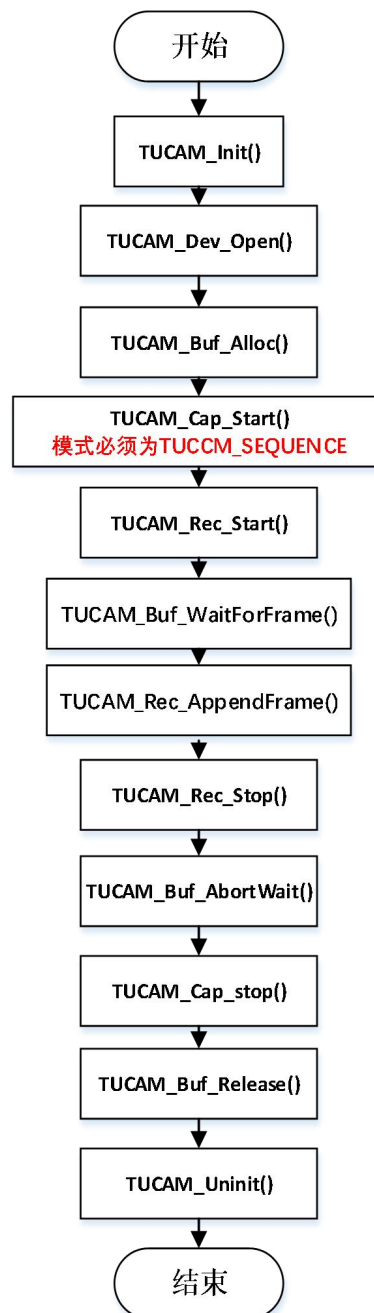
```
16.  
17. // 设置触发模式  
18. void SetTriggerMode()  
19. {  
20.     TUCAM_TRIGGER_ATTR tgrAttr;  
21.  
22.     tgrAttr.nTgrMode = TUCCM_TRIGGER_STANDARD;    // 标准触发模式  
23.     tgrAttr.nExpMode = TUCTE_EXPTM;              // 曝光模式  
24.     tgrAttr.nEdgeMode= TUCTE_RISING;              // 激发上升沿  
25.     tgrAttr.nFrames  = 1;                          // 触发 1 帧  
26.     tgrAttr.nDelayTm = 0;                          // 延时 0 ms  
27.  
28.     TUCAM_Cap_SetTrigger(m_opCam.hIdxTUCam, tgrAttr);  
29.     TUCAM_Cap_Start(m_opCam.hIdxCam, TUCCM_STANDARD);    // 标准触发模式  
30.  
31.     // 数据获取参考内存管理示例代码  
32. }  
33.
```

4.2.6 文件管理

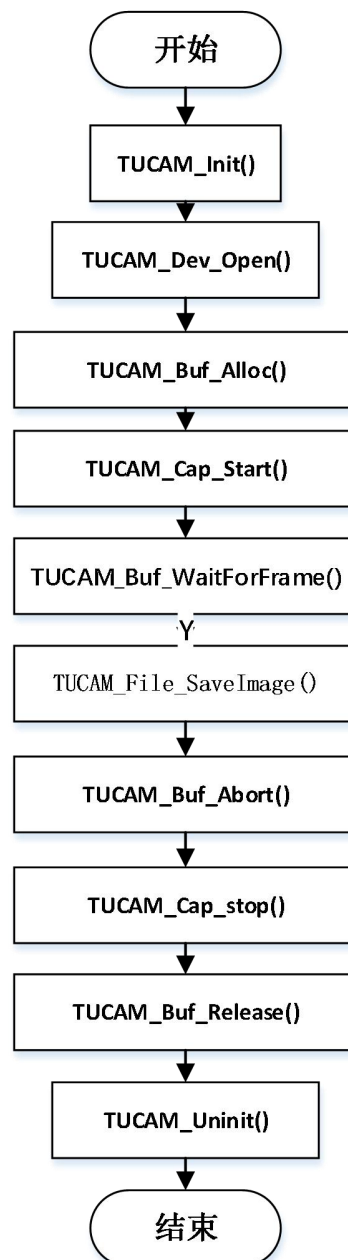
4.2.6.1 调用顺序

1) 视频流保存流程

录像开始 TUCAM_Rec_Start 需要在 TUCAM_Cap_Start 开始捕获数据之后调用，并且设置的捕获方式必须是 TUCCM_SEQUENCE 模式。录像过程中通过调用 TUCAM_Rec_AppendFrame 要将图像数据以追加的方式写入文件中，录像结束调用 TUCAM_Rec_Stop 来结束录像过程，再调用 TUCAM_CAP_STOP 结束采集。



2) 图片保存流程:



4.2.6.2 文件结构体

文件保存结构体参考 [TUCAM_File_Save](#)

录像保存结构体参考 [TUCAM_Rec_Save](#)

4.2.6.3 示例代码

参考以下代码或 Samples 目录 save_image/save_video 工程：

```
1. // 保存图片文件
2. void SaveImage()
3. {
4.     m_frame.ucFormatGet = TUFRM_FMT_USUAL;
5.     if(TUCAMRET_SUCCESS==TUCAM_Buf_WaitForFrame(m_opCam.hIdxTUCam,
6.         &m_frame))
7.     {
8.         TUCAM_FILE_SAVE fileSave;
9.         fileSave.nSaveFmt      = TUFMT_TIF;           // 保存 Tiff 格式
10.        fileSave.pFrame        = &m_frame;            // 需要保存的帧指针
11.        fileSave.pstrSavePath = "C:\\image";          // 路径包含文件名（不包含扩展名）
12.
13.        if (TUCAMRET_SUCCESS == TUCAM_File_SaveImage(m_opCam.hIdxTUCam,
14.            fileSave))
15.        {
16.            // 保存图像文件成功
17.        }
18.    }
19. // 保存录像文件
20. void StartRecording()
21. {
22.     TUCAM_REC_SAVE recSave;
23.     recSave.fFps          = 15.0f;                  // 需要保存的帧率
24.     recSave.nCodec        = m_dwFccHandler;
25.     recSave.pstrSavePath = "C:\\TUVideo.avi" // 全路径
26.
27.     if (TUCAMRET_SUCCESS == TUCAM_Rec_Start(m_opCam.hIdxTUCam, recSave))
28.     {
29.         // 开始录像。。。
30.     }
31. }
32.
33. Void AppendFrame()
34. {
35.     m_frame.ucFormatGet = TUFRM_FMT_RGB888;
36.     if(TUCAMRET_SUCCESS==TUCAM_Buf_WaitForFrame(m_opCam.hIdxTUCam,
37.         &m_frame))
38.     {
39.         TUCAM_Rec_AppendFrame(m_opCam.hIdxTUCam, &m_frame);
40.     }
```

```
40. }  
41.  
42. void StopRecording()  
43. {  
44.     TUCAM_Reg_Stop(m_opCam.hIdxTUCam);  
45. }  
46.
```

4.2.7 扩展控制

4.2.7.1 调用顺序

读写寄存器 TUCAM_Reg_Read / TUCAM_Reg_Write 的调用必须在 TUCAM_Dev_Open 打开相机后调用, 如果 TUCAM_Dev_Close 关闭相机后将无法读写寄存器, 必须重新打开相机。

4.2.7.2 寄存器结构体

参考 [TUCAM_REG_RW](#) 结构体

4.2.7.3 示例代码

```
1. // 读取寄存器数据  
2. void ReadRegisterData()  
3. {  
4.     char cSN[TUSN_SIZE] = {0};  
5.     TUCAM_REG_RW regRW;  
6.  
7.     regRW.nRegType = TUREG_SN;  
8.     regRW.pBuf      = &cSN[0];  
9.     regRW.nBufSize  = TUSN_SIZE;  
10.  
11.     if (TUCAMRET_SUCCESS == TUCAM_Reg_Read(m_opCam.hIdxTUCam, regRW))  
12.     {  
13.         // 获取 SN 数据  
14.     }  
15. }  
16.
```

```
17. // 写入寄存器数据
18. void WriteRegisterData()
19. {
20.     char cSN[TUSN_SIZE] = {'S', 'N', '1', '2', '3', '4', '5', '6'}; // "SN123456"
21.     TUCAM_REG_RW regRW;
22.
23.     regRW.nRegType = TUREG_SN;
24.     regRW.pBuf = &cSN[0];
25.     regRW.nBufSize = TUSN_SIZE;
26.
27.     if (TUCAMRET_SUCCESS == TUCAM_Reg_Write(m_opCam.hIdxTUCam, regRW))
28.     {
29.         // 将 SN 写入寄存器成功
30.     }
31. }
```

4.2.8 其它

4.2.8.1 制冷温度设置

注意： 温度获取 API 接口可以独立一个线程间隔 1s 来获取，并且此接口和其他接口需要互斥处理。

```
1. // 获取温度
2. void GetTemperature()
3. {
4.     double dblVal = 1.0f;
5.     TUCAM_Prop_GetValue(m_opCam.hIdxTUCam, TUIDP_TEMPERATURE, &dblVal);
6. }
7.
8. // 设置温度
9. void SetTemperature()
10. {
11.     int nVal = 40; // 设置-10℃ (40 - 50 = -10)
12.     TUCAM_Prop_SetValue(m_opCam.hIdxTUCam, TUIDP_TEMPERATURE, nVal);
13. } // 设置温度到 SDK 接口里数值是 0~100 实际对应温度时-50℃~50℃
14.
```


4.2.8.2 自动 / 一次曝光 / 自动曝光模式

```
1. // 相机设置自动曝光
2. TUCAM_Capa_SetValue(m_opCam.hldxTUCam, TUIDC_ATEXPOSURE, 1);
3.
4. // 相机设置一次曝光
5. TUCAM_Capa_SetValue(m_opCam.hldxTUCam, TUIDC_ATEXPOSURE, 2);
6. // 一直等待获取
7. int nVal = 0;
8. TUCAM_Capa_GetValue(m_opCam.hldxTUCam, TUIDC_ATEXPOSURE, &nVal);
9. // 一次曝光结束
10. if (0 == nVal)
11.
12. // 默认为居右曝光模式（仅 Microme 支持），此时不支持自动曝光目标值
    (TUIDP_BRIGHTNESS)
13. TUCAM_Capa_SetValue(m_opCam.hldxTUCam, TUIDC_ATEXPOSURE_MODE, 0);
14.
15. // 设置居中曝光模式，此时支持自动曝光目标值的设置
16. TUCAM_Capa_SetValue(m_opCam.hldxTUCam, TUIDC_ATEXPOSURE_MODE, 3);
17. TUCAM_Prop_SetValue(m_opCam.hldxTUCam, TUIDP_BRIGHTNESS, 128);
18.
```

4.2.8.3 自动色阶调用

```
1. // 自动设置模式(0 关闭,1 自动左色阶,2 自动右色阶,3 自动左右色阶)
2. int nVal = 0;
3.
4. If ( 0 == nVal )
5.     TUCAM_Capa_SetValue(m_opCam.hldxTUCam, TUIDC_HISTC, 0); // 关闭直方图统
    计
6. else
7.     TUCAM_Capa_SetValue(m_opCam.hldxTUCam, TUIDC_HISTC, 1); // 开启直方图统
    计
8.
9. // 设置自动色阶模式
10. TUCAM_Capa_SetValue(m_opCam.hldxTUCam, TUIDC_ATLEVELS, nVal);
11.
```

4.2.8.4 转 OpenCV 的 cvMat 格式

```
1. if(TUCAMRET_SUCCESS==TUCAM_Buf_WaitForFrame(m_opCam.hIdxTUCam,
   &m_frame))
2. {
3.     int type = CV_MAKETYPE(m_frame.ucDepth, m_frame.ucChannels);
4.     uchar *data = m_frame.pBuffer + m_frame.usHeader;
5.     Mat dst = Mat(m_frame.usHeight, m_frame.usWidth, type, data);
6. }
7.
```

4.2.8.5 转 C# 的 Bitmap 格式

```
1. m_frame.ucFormatGet = TUFRM_FMT_RGB888;
2. if (TUCAMRET.TUCAMRET_SUCCESS ==
   TUCamAPI.TUCAM_Buf_WaitForFrame(m_opCam.hIdxTUCam, ref m_frame))
3. {
4.     int nWidthStep = m_frame.uiWidthStep;
5.     int nSize = (int)(m_frame.uiImgSize + m_frame.usHeader);
6.     byte[] buffer = new byte[nSize];
7.     Marshal.Copy(m_frame.pBuffer, buffer, 0, nSize);
8.     // 偏移头部数据
9.     Buffer.BlockCopy(buffer, (int)(m_frame.usHeader), buffer, 0, (int)(m_frame.uiImgSize));
10.
11.    // 转换为 Bitmap
12.    int stride = (int)(m_frame.uiWidthStep);
13.    GCHandle handle = GCHandle.Alloc(buffer, GCHandleType.Pinned);
14.    int scan0 = (int)handle.AddrOfPinnedObject();
15.    scan0 += (m_frame.usHeight - 1) * stride;
16.    System.Drawing.Bitmap bitmap = new System.Drawing.Bitmap(m_frame.usWidth,
   m_frame.usHeight, -stride, System.Drawing.Imaging.PixelFormat.Format24bppRgb,
   (IntPtr)scan0);
17.    handle.Free();
18.
19.    if (null != bmpDraw)
20.    {
21.        bmpDraw.Dispose();
22.    }
23.
24.    if (null != bitmap)
25.    {
```

```
26.         // 这个数据就是 C#的 Bitmap, 可以转成 Image
27.         Bitmap bmpDraw = bitmap.Clone(new Rectangle(0, 0,
    bitmap.Width,bitmap.Height),bitmap.PixelFormat);
28.     }
29. }
30.
```

4.2.8.6 计算平均灰度值

```
1.  if(TUCAMRET_SUCCESS == TUCAM_Buf_WaitForFrame(m_opCam.hIdxTUCam,
    &m_frame))
2.  {
3.      // 计算全幅区域或 ROI 的平均灰度值
4.      bool isRoi = false;
5.
6.      // ROI 参数
7.      int roiX = 200;
8.      int roiY = 400;
9.      int roiW = 800;
10.     int roiH = 600;
11.
12.     int startX  = isRoi ? roiX : 0;
13.     int startY  = isRoi ? roiY : 0;
14.     int width   = isRoi ? roiW : m_frame.usWidth;
15.     int height  = isRoi ? roiH : m_frame.usHeight;
16.     int finishX = startX + width;
17.     int finishY = startY + height;
18.
19.     int pixels = width * height;
20.     double avg = 0;
21.     long long sum = 0;
22.
23.     // 16bit
24.     if ( 2 == m_frame.ucElemBytes )
25.     {
26.         unsigned short *data = (unsigned short *)(m_frame.pBuffer + m_frame.usHeader);
27.
28.         for (int i = startY; i < finishY; ++i)
29.         {
30.             for (int j = startX; j < finishX; ++j)
```

```
31.         {
32.             sum += data[i * m_frame.usWidth + j];
33.         }
34.     }
35. }
36. else // 8bit
37. {
38.     unsigned char *data = (unsigned char*)(m_frame.pBuffer + m_frame.usHeader);
39.
40.     for (int i = startY; i < finishY; ++i)
41.     {
42.         for (int j = startX; j < finishX; ++j)
43.         {
44.             sum += data[i * m_frame.usWidth + j];
45.         }
46.     }
47. }
48.
49. // 计算平均值
50. avg = sum * 1.0 / pixels;
51. }
52.
```

5. 参考

5.1 类型和常量

5.1.1 TUCAMRET 错误代码

错误代码常量 TUCAMRET_ 为前缀，按照类型分为初始化错误、状态错误、等待错误、调用错误、相机或接口错误。

名称	错误代码	描述
TUCAMRET_SUCCESS	0x00000001	没有错误，成功代码，应用程序应检查值为正
TUCAMRET_RECEIVE_FINISH	0x00000002	没有错误，厂商收到帧信息
TUCAMRET_EXTERNAL_TRIGGER	0x00000003	没有错误，接收到外部触发信号
TUCAMRET_FAILURE	0x80000000	调用 API 接口失败

初始化错误		
TUCAMRET_NO_MEMORY	0x80000101	没有足够的内存
TUCAMRET_NO_RESOURCE	0x80000102	没有足够的资源（不包括内存）
TUCAMRET_NO_MODULE	0x80000103	没有支持的子模块
TUCAMRET_NO_DRIVER	0x80000104	没有支持的驱动
TUCAMRET_NO_CAMERA	0x80000105	没有连接的相机
TUCAMRET_NO_GRABBER	0x80000106	没有取图
TUCAMRET_NO_PROPERTY	0x80000107	没有代替的属性 ID
TUCAMRET_FAILOPEN_CAMERA	0x80000110	打开相机失败
TUCAMRET_FAILOPEN_BULKIN	0x80000111	打开批传输输入端点失败（USB 接口）
TUCAMRET_FAILOPEN_BULKOUT	0x80000112	打开批传输输出端点失败（USB 接口）
TUCAMRET_FAILOPEN_CONTROL	0x80000113	打开控制端点失败
TUCAMRET_FAILCLOSE_CAMERA	0x80000114	关闭相机失败
TUCAMRET_FAILOPEN_FILE	0x80000115	打开文件失败
TUCAMRET_FAILOPEN_CODEEC	0x80000116	打开编码器失败
TUCAMRET_FAILOPEN_CONTEXT	0x80000117	打开上下文失败
状态错误		
TUCAMRET_INIT	0x80000201	API 需要初始化状态
TUCAMRET_BUSY	0x80000202	API 处于繁忙状态
TUCAMRET_NOT_INIT	0x80000203	API 未初始化
TUCAMRET_EXCLUDED	0x80000204	一些资源被独占使用
TUCAMRET_NOT_BUSY	0x80000205	API 未处于繁忙状态
TUCAMRET_NOT_READY	0x80000206	API 未处于就绪状态
等待错误		
TUCAMRET_ABORT	0x80000207	终止处理
TUCAMRET_TIMEOUT	0x80000208	超时
TUCAMRET_LOSTFRAME	0x80000209	帧丢失
TUCAMRET_MISSFRAME	0x8000020A	帧丢失但是是底层驱动的问题
TUCAMRET_USB_STATUS_ERROR	0x8000020B	USB 状态错误
调用错误		
TUCAMRET_INVALID_CAMERA	0x80000301	无效相机
TUCAMRET_INVALID_HANDLE	0x80000302	无效相机句柄
TUCAMRET_INVALID_OPTION	0x80000303	无效配置的值
TUCAMRET_INVALID_IDPROP	0x80000304	无效属性 ID
TUCAMRET_INVALID_IDCAPA	0x80000305	无效性能 ID
TUCAMRET_INVALID_IDPARAM	0x80000306	无效参数 ID
TUCAMRET_INVALID_PARAM	0x80000307	无效参数
TUCAMRET_INVALID_FRAMEIDX	0x80000308	无效帧序列号
TUCAMRET_INVALID_VALUE	0x80000309	无效值
TUCAMRET_INVALID_EQUAL	0x8000030A	值相等，参数无效

TUCAMRET_INVALID_CHANNEL	0x8000030B	属性 ID 指定通道，但通道无效
TUCAMRET_INVALID_SUBARRAY	0x8000030C	子数组的值是无效的
TUCAMRET_INVALID_VIEW	0x8000030D	无效的显示窗口句柄
TUCAMRET_INVALID_PATH	0x8000030E	无效的文件路径
TUCAMRET_INVALID_IDVPROP	0x8000030F	无效的厂商属性 ID
TUCAMRET_NO_VALUETEXT	0x80000310	属性没有值的文本
TUCAMRET_OUT_OF_RANGE	0x80000311	值超出范围
TUCAMRET_NOT_SUPPORT	0x80000312	相机不支持性能或属性
TUCAMRET_NOT_WRITABLE	0x80000313	属性不可写
TUCAMRET_NOT_READABLE	0x80000314	属性不可读
TUCAMRET_WRONG_HANDSHAKE	0x80000410	错误发生在获取错误代码时
TUCAMRET_NEWAPI_REQUIRED	0x80000411	旧 API 不支持，只有新 API 支持
TUCAMRET_ACCESSDENY	0x80000412	访问拒绝（可能是没有足够权限）
TUCAMRET_NO_CORRECTIONDATA	0x80000501	没有彩色校点数据
TUCAMRET_INVALID_PRFSETS	0x80000601	配置文件设置名称无效
TUCAMRET_INVALID_IDPPROP	0x80000602	无效的属性 ID
TUCAMRET_DECODE_FAILURE	0x80000701	解码失败
TUCAMRET_COPYDATA_FAILURE	0x80000702	拷贝数据失败
TUCAMRET_ENCODE_FAILURE	0x80000703	编码失败
TUCAMRET_WRITE_FAILURE	0x80000704	写入失败
相机或总线错误		
TUCAMRET_FAIL_READ_CAMERA	0x83001001	从相机读取失败
TUCAMRET_FAIL_WRITE_CAMERA	0x83001002	写入相机失败
TUCAMRET_OPTICS_UNPLUGGED	0x83001003	光学部件已拆下，请检查

5.1.2 TUCAM_IDINFO 产品信息代码

名称	代码	描述
TUIDI_BUS	0x00	USB 接口类型。0x200、0x210 对应 USB2.0。
TUIDI_VENDOR	0x01	厂商 ID
TUIDI_PRODUCT	0x02	产品 ID
TUIDI_VERSION_API	0x03	当前使用 TUCAM-API 接口版本
TUIDI_VERSION_FRMW	0x04	当前相机固件版本号
TUIDI_VERSION_FPGA	0x06	当前相机 FPGA 版本号
TUIDI_VERSION_DRIVER	0x07	当前相机驱动版本号
TUIDI_TRANSFER_RATE	0x08	传输速率
TUIDI_CAMERA_MODEL	0x09	当前相机型号，字符串类型（例如 Dhyana 400BSI V3）

TUIDI_CURRENT_WIDTH	0x0A	相机图像数据宽度（必须使用 TUCAM_Dev_GetInfoEx, 并在 TUCAM_Buf_Alloc 后调用）
TUIDI_CURRENT_HEIGHT	0x0B	相机图像数据高度（必须使用 TUCAM_Dev_GetInfoEx, 并在 TUCAM_Buf_Alloc 后调用）
TUIDI_CAMERA_CHANNELS	0x0C	相机图像数据通道数。彩色相机为 3, 黑白相机为 1。
TUIDI_BCDDEVICE	0x0D	BCD 码
TUIDI_TEMPALARMFLAG	0x0E	温度预警标志
TUIDI_UTCTIME	0x0F	获取 UTC（世界统一时间、世界标准时间）
TUIDI_LONGITUDE_LATITUDE	0x10	获取经纬度
TUIDI_WORKING_TIME	0x11	获取工作时间
TUIDI_FAN_SPEED	0x12	获取风扇速度
TUIDI_FPGA_TEMPERATURE	0x13	获取 FPGA 温度
TUIDI_PCBA_TEMPERATURE	0x14	获取 PCBA 温度
TUIDI_ENV_TEMPERATURE	0x15	获取环境温度
TUIDI_DEVICE_ADDRESS	0x16	USB 设备地址
TUIDI_USB_PORT_ID	0x17	USB 设备端口号
TUIDI_CONNECTSTATUS	0x18	相机连接状态 0：连接、1：断连
TUIDI_ZEROTEMPERATURE_VALUE	0x1D	获取相机 0 度的参数值
TUIDI_VALID_FRAMEBIT	0x1E	获取相机帧有效位数
TUIDI_ENDINFO	0x20	产品信息 ID 结束位

5.1.3 TUCAM_IDCAPA 性能代码

名称	代码	描述	读/写
TUIDC_RESOLUTION	0x00	分辨率	R/W
TUIDC_PIXELCLOCK	0x01	像素时钟	R/W
TUIDC_BITOFDEPTH	0x02	数据位宽（8bit, 16bit）	R/W
TUIDC_ATEXPOSURE	0x03	自动曝光（0：关闭 1：打开）	R/W
TUIDC_HORIZONTAL	0x04	水平镜像（0：关闭 1：打开）	R/W
TUIDC_VERTICAL	0x05	垂直镜像（0：关闭 1：打开）	R/W
TUIDC_ATWBALANCE	0x06	自动白平衡（仅彩色相机支持） （0：关闭 1：打开）	R/W
TUIDC_FAN_GEAR	0x07	风扇档位（仅制冷相机支持）	R/W
TUIDC_ATLEVELS	0x08	自动色阶[0, 3] 0：关闭 （1：左色阶 2：右色阶 3：左右色阶）	R/W
TUIDC_SHIFT	0x09	显示移位[16bit 有效]，选取其中连续 8 位(15~8, 14~7, 13~6, 12~5, 11~4, 10~3, 9~2, 8~1)	R/W

		10~3, 9~2, 8~1, 7~0)	
TUIDC_HISTC	0x0A	直方图统计使能 (0: 关闭 1: 打开)	R/W
TUIDC_CHANNELS	0x0B	当前通道索引 (参考 TUCHN_SELECT, 仅彩色相机)	R/W
TUIDC_ENHANCE	0x0C	图像增强使能	R/W
TUIDC_DFTCORRECTION	0x0D	坏点校正 (某几款相机支持) (0: 不校正 1: 计算 2: 校正)	R/W
TUIDC_ENABLEDENOISE	0x0E	降噪使能, 受降噪等级影响 (参考 TUIDP_NOISELEVEL)	R/W
TUIDC_FLTCORRECTION	0x0F	平场校正 0: 关闭 (1: 抓取帧 2: 计算 3: 校正)	R/W
TUIDC_RESTARTLONGTM	0x10	重启长时间曝光 (仅 CCD 相机支持)	R/W
TUIDC_DATAFORMAT	0x11	数据格式 (某款相机支持) (仅支持 0: YUV 1: RAW 切换)	R/W
TUIDC_DRCORRECTION	0x12	动态范围校正	R/W
TUIDC_VERCORRECTION	0x13	垂直镜像校正 (校正数据垂直显示, 在 Windows 系统中默认值为 1)	R/W
TUIDC_MONOCHROME	0x14	单色使能	R/W
TUIDC_BLACKBALANCE	0x15	黑平衡使能	R/W
TUIDC_IMGMODESELECT	0x16	图像模式选择 (CMS 模式使能)	R/W
TUIDC_CAM_MULTIPLE	0x17	多相机设置 (设置同时捕获数据的相机个数)	RO
TUIDC_ENABLEPOWEEFREQUENCY	0x18	消频闪使能 (0: 50hz 1: 60hz)	R/W
TUIDC_ROTATE_R90	0x19	顺时针转 90 度使能	R/W
TUIDC_ROTATE_L90	0x1A	逆时针转 90 度使能	R/W
TUIDC_NEGATIVE	0x1B	负片使能	R/W
TUIDC_HDR	0x1C	HDR 使能	R/W
TUIDC_ENABLEIMGPRO	0x1D	图像处理使能	R/W
TUIDC_ENABLELED	0x1E	相机 LED 灯使能	R/W
TUIDC_TUIDC_ENABLETIMESTAMP	0x1F	时间戳使能	R/W
TUIDC_ENABLEBLACKLEVEL	0x20	黑电平	R/W
TUIDC_ATFOCUS	0x21	自动对焦使能 (0: 手动 1: 自动对焦 2: 一次对焦)	R/W
TUIDC_ATFOCUS_STATUS	0x22	自动对焦状态 (参考 TUFOCUS_STATUS)	R/W
TUIDC_PGAGAIN	0x23	传感器增益 (针对某些相机仅包含传感器 高增益或传感器低增益其中一种)	R/W
TUIDC_ATEXPOSURE_MODE	0x24	自动曝光模式 (0: 居中曝光 3: 居右曝 光, 支持目标值)	R/W
TUIDC_BINNING_SUM	0x25	软件 Binning, 像素求和模式	R/W
TUIDC_BINNING_AVG	0x26	软件 Binning, 像素求平均模式	R/W

TUIDC_FOCUS_C_MOUNT	0x27	C-Mount 聚焦模式 (0: 普通模式 1: C-Mount 模式)	R/W
TUIDC_ENABLEPI	0x28	PI (加热膜) 使能	R/W
TUIDC_ATEXPOSURE_STATUS	0x29	自动曝光状态 (0: 正在执行 1: 完成)	RO
TUIDC_ATWBALANCE_STATUS	0x2A	自动白平衡 (0: 正在执行 1: 完成)	RO
TUIDC_TESTIMGMODE	0x2B	测试图片模式 (多种测试图, 与拍摄场景无关)	R/W
TUIDC_SENSORRESET	0x2C	传感器重置 (设置一次)	WO
TUIDC_PGAAHIGH	0x2D	传感器高增益	R/W
TUIDC_PGALOW	0x2E	传感器低增益	R/W
TUIDC_PIXCLK1_EN	0x2F	像素时钟 1 使能	R/W
TUIDC_PIXCLK2_EN	0x30	像素时钟 2 使能	R/W
TUIDC_ATLEVELGEAR	0x31	自动色阶档位	R/W
TUIDC_ENABLEDSNU	0x32	暗信号非均匀性使能	R/W
TUIDC_ENABLEOVERLAP	0x33	启用曝光时间重叠模式	R/W
TUIDC_CAMSTATE	0x34	相机状态	R/W
TUIDC_ENABLETRIOUT	0x35	触发输出使能	R/W
TUIDC_ROLLINGSCANMODE	0x36	卷帘扫描模式使能	R/W
TUIDC_ROLLINGSCANLTD	0x37	卷帘扫描行延迟时间	R/W
TUIDC_ROLLINGSCANSPLIT	0x38	卷帘扫描间隔高度	R/W
TUIDC_ROLLINGSCANDIR	0x39	卷帘扫描方向 (0: 向下 1: 向上 2: 自下而上循环)	R/W
TUIDC_ROLLINGSCANRESET	0x3A	卷帘扫描方向重置	R/W
TUIDC_ENABLETEC	0x3B	TEC (制冷开关) 使能	R/W
TUIDC_ENABLEBLC	0x3C	背光补偿使能	R/W
TUIDC_ENABLETHROUGHFOG	0x3D	电子透雾使能 (有等级) 图像处理	R/W
TUIDC_ENABLEGAMMA	0x3E	伽马使能	R/W
TUIDC_ENABLEFILTER	0x3F	滤波使能	R/W
TUIDC_ENABLEHLC	0x40	强光抑制使能	R/W
TUIDC_CAMPARASAVE	0x41	相机参数保存	R/W
TUIDC_CAMPARALOAD	0x42	相机参数读取	R/W
TUIDC_ENABLEISP	0x43	相机 ISP 算法使能	R/W
TUIDC_BUFFERHEIGHT	0x44	缓存高度参数	R/W
TUIDC_VISIBILITY	0x45	可见等级参数	R/W
TUIDC_SHUTTER	0x46	Shutter 模式设置	R/W
TUIDC_SIGNALFILTER	0x47	触发信号滤波参数	R/W
TUIDC_ATEXPOSURE_TYPE	0x48	自动曝光类型参数 (0: 曝光和增益 1: 仅曝光 2: 仅增益)	R/W
TUIDC_ATWBALANCE_TYPE	0x49	自动白平衡类型参数 (0: 正常模式 1: AICC 模式)	R/W
TUIDC_ENABLELOWPOWER	0x4A	相机低功耗模式配置	R/W

TUIDC_ENDCAPABILITY	0x4B	相机性能 ID 结束位	R/W
---------------------	------	-------------	-----

5.1.4 TUCAM_IDPROP 属性代码

名称	代码	描述	读/写
TUIDP_GLOBALGAIN	0x00	全局增益（传感器调节信号放大程度）	R/W
TUIDP_EXPOSURETM	0x01	曝光时间 ms	R/W
TUIDP_BRIGHTNESS	0x02	自动曝光目标值 （居右自动曝光状态有效）	R/W
TUIDP_BLACKLEVEL	0x03	黑电平	R/W
TUIDP_TEMPERATURE	0x04	相机（Sensor）温度	R/W
TUIDP_SHARPNESS	0x05	锐化等级	R/W
TUIDP_NOISELEVEL	0x06	降噪等级	R/W
TUIDP_HDR_KVALUE	0x07	HDR 的 K 值（sCMOS 相机支持）	R/W
图片处理属性			
TUIDP_GAMMA	0x08	伽玛	R/W
TUIDP_CONTRAST	0x09	对比度	R/W
TUIDP_LFTLEVELS	0x0A	左色阶	R/W
TUIDP_RGTLEVELS	0x0B	右色阶	R/W
TUIDP_CHNLGAIN	0x0C	通道增益(彩色相机支持) 独立调节 RGB 通道信号强度的参数	R/W
TUIDP_SATURATION	0x0D	饱和度（彩色相机支持）	R/W
TUIDP_CLRTEMPERATURE	0x0E	色温	R/W
TUIDP_CLRMATRIX	0x0F	色彩矩阵	R/W
TUIDP_DPCLEVEL	0x10	坏点校正	R/W
TUIDP_BLACKLEVELHG	0x11	黑电平高增益	R/W
TUIDP_BLACKLEVELLG	0x12	黑电平低增益	R/W
TUIDP_POWEEFREQUENCY	0x13	消频闪(50HZ 或 60HZ)	R/W
TUIDP_HUE	0x14	色调（HSL 色彩空间的色调）	R/W
TUIDP_LIGHT	0x15	亮度（HSL 色彩空间的亮度）	R/W
TUIDP_ENHANCE_STRENGTH	0x16	增强（通透性）	R/W
TUIDP_NOISELEVEL_3D	0x17	3D 降噪	R/W
TUIDP_FOCUS_POSITION	0x18	对焦位置	R/W
TUIDP_FRAME_RATE	0x19	帧率	R/W
TUIDP_START_TIME	0x1A	GPS 开始时间	R/W
TUIDP_FRAME_NUMBER	0x1B	GPS 采集张数	R/W
TUIDP_INTERVAL_TIME	0x1C	GPS 间隔时间	R/W
TUIDP_GPS_APPLY	0x1D	GPS 使能开关	R/W
TUIDP_AMB_TEMPERATURE	0x1E	环境温度（外接设备传入）	R/W
TUIDP_AMB_HUMIDITY	0x1F	环境湿度（外接设备传入）	R/W

TUIDP_AUTO_CTRLTEMP	0x20	自动控制温度使能	R/W
TUIDP_AVERAGEGRAY	0x21	平均灰度值设置	R/W
TUIDP_AVERAGEGRAYTHD	0x22	平均灰度值阈值	R/W
TUIDP_ENHANCETHD	0x23	增强阈值设置	R/W
TUIDP_ENHANCEPARA	0x24	增强参数设置	R/W
TUIDP_EXPOSUREMAX	0x25	最大曝光时间设置	R/W
TUIDP_EXPOSUREMIN	0x26	最小曝光时间设置	R/W
TUIDP_GAINMAX	0x27	最大增益设置	R/W
TUIDP_GAINMIN	0x28	最小增益设置	R/W
TUIDP_THROUGHFOGPARA	0x29	透雾参数设置	R/W
TUIDP_ATLEVEL_PERCENTAGE	0x2A	自动色阶忽略值设置	R/W
TUIDP_TEMPERATURE_TARGET	0x2B	温度目标值设置	R/W
TUIDP_PIXELRATIO	0x2C	一次曝光像素比例设置	R/W
TUIDP_ENDPROPERTY	0x2D	属性 ID 结束位	R/W

5.1.5 TUCAM_IDPPROP 图像拼接代码

名称	代码	描述
TUIDPP_EDF_QUALITY	0x00	景深融合质量（1：低质量 2：中质量 3：高质量）
TUIDPP_STITCH_SPEED	0x01	图像拼接速度（0：快速 1：正常）
TUIDPP_STITCH_BGC_RED	0x02	图像拼接背景红色通道值
TUIDPP_STITCH_BGC_GREEN	0x03	图像拼接背景绿色通道值
TUIDPP_STITCH_BGC_BLUE	0x04	图像拼接背景蓝色通道值
TUIDPP_STITCH_VALID	0x05	图像拼接结果无效（只读）
TUIDPP_STITCH_AREA_X	0x06	图像拼接结果当前 X 坐标值（只读）
TUIDPP_STITCH_AREA_Y	0x07	图像拼接结果当前 Y 坐标值（只读）
TUIDPP_STITCH_NEXT_X	0x08	图像拼接结果下一个 X 坐标值（只读）
TUIDPP_STITCH_NEXT_Y	0x09	图像拼接结果下一个 Y 坐标值（只读）
TUIDPP_ENDPPROPERTY	0x0A	图片拼接属性 ID 结束位

5.1.6 TUCAM_IDCROI 区域计算代码

名称	代码	描述
TUIDCR_WBALANCE	0x00	计算区域白平衡
TUIDCR_BBALANCE	0x01	计算区域黑平衡
TUIDCR_BLOFFSET	0x02	计算区域黑电平偏移
TUIDCR_FOCUS	0x03	计算区域对焦
TUIDCR_EXPOSURETM	0x04	计算区域曝光时间

TUIDCR_END	0x05	计算区域 ID 结束位
------------	------	-------------

5.1.7 TUCAM_CAPTURE_MODES 捕获模式代码

名称	代码	描述
TUCCM_SEQUENCE	0x00	采用序列模式（流模式），捕获连续的图像数据
TUCCM_TRIGGER_STANDARD	0x01	采用标准触发模式
TUCCM_TRIGGER_SYNCHRONOUS	0x02	采用同步触发模式
TUCCM_TRIGGER_GLOBAL	0x03	采用全局触发模式
TUCCM_TRIGGER_SOFTWARE	0x04	采用软件触发模式
TUCCM_TRIGGER_GPS	0x05	采用 GPS 触发模式
TUCCM_TRIGGER_STANDARD_NONOVERLAP	0x11	采用标准触发模式

5.1.8 TUIMG_FORMATS 图像格式代码

名称	代码	描述
TUFMT_RAW	0x01	保存图像为 RAW 格式。Tucsen 自定义的一种相机图像文件格式（包含头部 + 图像数据）
TUFMT_TIF	0x02	保存图像为 TIF 格式，也叫 TIF 格式。支持不同颜色模式、路径、透明度和通道，是打印文档最常用的格式。 优点：保存丰富的图像层次和细节，画面质量无损。 缺点：占用存储空间大。
TUFMT_PNG	0x04	保存图像为 PNG 格式。 优点：支持高级别无损压缩；支持透明背景。 缺点：对旧浏览器和软件兼容性较差。
TUFMT_JPG	0x08	保存图像为 JPG 格式。 优点：能将图片压缩至很小的存储空间，对色彩信息的保留较好。 缺点：有损压缩会降低图片数据质量。
TUFMT_BMP	0x10	保存图像为 BMP 格式。Windows 系统中标准图像文件格式，支持多种 Windows 应用程序使用。 优点：无损压缩；图像画质优秀。 缺点：占用存储空间大。

5.1.9 TUREG_TYPE 寄存器类型代码

名称	代码	描述
TUREG_SN	0x01	读写 SN 码寄存器

TUREG_DATA	0x02	读写数据内容寄存器
TUREG_BAD_ROW	0x03	读写坏行寄存器（厂商使用）
TUREG_BAD_COL	0x04	读写坏列寄存器（厂商使用）
TUREG_BGC	0x05	读写背景寄存器（厂商使用）
TUREG_HDR	0x06	读写 HDR 曝光参数寄存器（厂商使用）
TUREG_HBG	0x07	读写 HBG 曝光参数寄存器（厂商使用）
TUREG_CMS	0x08	读写 CMS 曝光参数寄存器（厂商使用）
TUREG_CBG	0x09	读写 CBG 曝光参数寄存器（厂商使用）
TUREG_CODE	0x0A	读写代码参数寄存器（厂商使用）
TUREG_DPC	0x0B	读写 DPC（坏点校正）参数寄存器（厂商使用）
TUREG_TEMPERATUREOFFSET	0x0C	读写温度寄存器（厂商使用）
TUREG_HIGHSPEEDBGYLIST	0x0D	读写列扣背景寄存器（厂商使用）

5.1.10 TUCAM_TRIGGER_EXP 触发曝光时间模式代码

名称	代码	描述
TUCTE_EXPTM	0x00	接收到触发信号后，由 TUIDP_EXPOSURETM 设置的曝光时间决定
TUCTE_WIDTH	0x01	接收到触发信号后，曝光时间由电平的宽度所决定

5.1.11 TUCAM_TRIGGER_EDGE 激发边沿类型代码

名称	代码	描述
TUCTD_RISING	0x01	接收到的触发电平处于上升沿时开始曝光
TUCTD_FALLING	0x00	接收到的触发电平处于下降沿时开始曝光

5.1.12 TUCAM_TRIGGER_READOUTDIRRESET 触发读取输出复位代码

名称	代码	描述
TUCTD_YES	0x00	重置
TUCTD_NO	0x01	不重置

5.1.13 TUCAM_TRIGGER_READOUTDIR 触发输出方向代码

名称	代码	描述
----	----	----

TUCTD_DOWN	0x00	向下
TUCTD_UP	0x01	向上
TUCTD_DOWNUPCYC	0x02	向下循环

5.1.14 TUFRM_FORMATS 帧格式代码

名称	代码	描述
TUFRM_FMT_RAW	0x10	RAW 格式数据
TUFRM_FMT_USUAI	0x11	通常格式数据（8bit/16bit、黑白/彩色）
TUFRM_FMT_RGB888	0x12	RGB888 格式数据（可用于显示）

5.1.15 TUGAIN_MODE 增益模式代码

名称	代码	描述
TUGAIN_HDR	0x00	HDR 模式
TUGAIN_HIGH	0x01	高增益模式
TUGAIN_LOW	0x02	低增益模式

5.1.16 TUCHN_SELECT 通道选择代码

名称	代码	描述
TUCHN_SHARE	0x00	共享通道（灰色或 RGB）
TUCHN_RED	0x01	通道 1（红色通道）
TUCHN_GREEN	0x02	通道 2（绿色通道）
TUCHN_BLUE	0x03	通道 3（蓝色通道）

5.1.17 TUCAM_OUTPUTTRG_PORT 触发输出端口代码

名称	代码	描述
TUPORT_ONE	0x00	使用端口 1
TUPORT_TWO	0x01	使用端口 2
TUPORT_THREE	0x02	使用端口 3

5.1.18 TUCAM_OUTPUTTRG_KIND 触发输出种类代码

名称	代码	描述
TUOPT_GND	0x00	低电平

TUOPT_VCC	0x01	高电平
TUOPT_IN	0x02	触发输入
TUOPT_EXPSTART	0x03	曝光开始参考点
TUOPT_EXPGLOBAL	0x04	全局曝光
TUOPT_READEND	0x05	读出结束
TUOPT_TRIREADY	0x06	触发就绪

5.1.19 TUCAM_OUTPUTTRG_EDGE 触发输出边沿代码

名称	代码	描述
TUOPT_RISING	0x00	触发上升边沿
TUOPT_FAILING	0x01	触发下降边沿

5.1.20 TUDRAW_MODE 图像绘制模式代码

仅支持 Windows 操作系统，TUDRAW_DFT 模式

名称	代码	描述
TUDRAW_DFT	0x00	默认绘制模式
TUDRAW_DIB	0x01	DIB 绘制模式
TUDRAW_DX9	0x02	DirectX 9.0

5.1.21 TUREC_MODE 录制文件关键帧模式代码

名称	代码	描述
TUREC_TIMESTAMP	0x01	根据时间戳确定关键帧位置（To Disk 模式使用）
TUREC_SEQUENCE	0x02	根据图像序列确定关键帧位置（To Ram 模式使用）

5.1.22 TUPROC_TYPE 图像处理模式代码

名称	代码	描述
TUPROC_EDF	0x00	景深融合模式
TUPROC_STITCH	0x01	图像拼接

5.1.23 TUSTITCH_MODE 图像处理拼接模式代码

名称	代码	描述
----	----	----

TUSM_FINE	0x00	拼接正常质量（速度快）
TUSM_EXCELLENT	0x01	拼接高质量（速度慢）

5.1.24 TUFOCUS_STATUS 对焦状态代码

名称	代码	描述
TUFS_STOP	0x00	对焦停止状态
TUFS_FOCUSING	0x01	正在对焦状态
TUFS_COMPLETED	0x02	对焦完成状态
TUFS_DEFOCUS	0x03	离焦状态

5.1.25 TUCAM_IDVPROP 厂商属性（仅限厂商使用）

名称	代码	描述
TUIDV_ADDR_FLASH	0x00	供应商闪存地址
TUIDV_ODDEVENH	0x01	奇偶高值
TUIDV_ODDEVENL	0x02	奇偶低值
TUIDV_HDRHGBOFFSET	0x03	HDR 高增益位偏移
TUIDV_HDRLGBOFFSET	0x04	HDR 低增益位偏移
TUIDV_CMSHGBOFFSET	0x05	CMS 高增益位偏移
TUIDV_CMSLGBOFFSET	0x06	CMS 低增益位偏移
TUIDV_FPNENABLE	0x07	FPN（图像固定噪声）减小使能
TUIDV_WORKING_TIME	0x08	工作时间
TUIDV_CALC_DSNU	0x09	计算 DSNU（暗信号非均匀性）
TUIDV_CALC_PRNU	0x0A	计算 PSNU（光信号非均匀性）
TUIDV_CALC_DPC	0x0B	计算 DPC（坏点校正）
TUIDV_CALC_STOP	0x0C	计算停止
TUIDV_CALC_STATE	0x0D	获取计算状态，参考值有 0：接口空闲 2：接口繁忙 3：计算繁忙
TUIDV_HDR_LVALUE	0x0E	HDR 低值
TUIDV_HDR_HVALUE	0x0F	HDR 高值
TUIDV_FW_CHECK	0x10	固件检查值
TUIDV_HIGHSPEEDHGBOFFSET	0x11	高速高增益位偏移
TUIDV_HIGHSPEEDLGBOFFSET	0x12	高速低增益位偏移
TUIDV_TEMPERATURE_OFFSET	0x13	温度偏移
TUIDV_ENDVPROPERTY	0x14	厂商属性 ID 结束位

5.2 结构体

一些函数需要一个指向结构体的指针作为参数，对于结构体参数，结构体中有[in]标识变量，意味着应用程序必须在调用之前设置值，有[out]标识变量，意味着函数返回时填充一个值。

5.2.1 初始化

```
31. typedef struct _tagTUCAM_INIT
32. {
33.     UINT32    uiCamCount;
34.     INT32     uiHostCamIdx;
35.     PCHAR     pstrConfigPath;
36. }TUCAM_INIT, *PTUCAM_INIT;
```

接口 TUCAM_API_Init 输入参数：

名称	描述
uiCamCount	[out] 返回当前连接的相机个数
uiHostCamIdx	[out] 返回主相机 ID(当前无使用为默认值-1)
pstrConfigPath	[in] 输入保存相机参数的路径，当执行 TUCAM_File_SaveProfiles 函数时保存该路径下(保存文件格式是 xml，保存路径不要带特殊字符)

5.2.2 打开相机

```
1. typedef struct _tagTUCAM_OPEN
2. {
3.     UINT32    uiIdxOpen;
4.     HDTUCAM  hIdxTUCam;
5. }TUCAM_OPEN, *PTUCAM_OPEN;
```

接口 TUCAM_Dev_Open 输入参数：

名称	描述
----	----

uIdxOpen	[in] 打开指定索引相机
hIdxTUCam	[out] 已打开相机的句柄

5.2.3 打开图像

```

1. typedef struct _tagTUIMG_OPEN
2. {
3.     PCHAR    pszfileName;
4.     HDTUIMG hIdxTUImg;
5. }TUIMG_OPEN, *PTUIMG_OPEN;
6.

```

接口 TUIMG_File_Open 输入参数：

名称	描述
pszfileName	[in] 打开指定索引相机
hIdxTUImg	[out] 已打开图像的句柄

5.2.4 相机信息

```

1. typedef struct _tagTUCAM_VALUE_INFO
2. {
3.     INT32    nID;
4.     INT32    nValue;
5.     PCHAR    pText;
6.     INT32    nTextSize;
7. }TUCAM_VALUE_INFO, *PTUCAM_VALUE_INFO;
8.

```

接口 TUCAM_Get_Info \ TUCAM_Get_InfoEx 输入参数：

名称	描述
nID	[in] 设备信息 ID,参考 TUCAM_IDINFO
nValue	[in] 设备信息的值
pText	[in / out] 指向设备信息文本的指针
nTextSize	[in] 文本缓存大小

5.2.5 性能的属性

```

1. typedef struct _tagTUCAM_CAPA_ATTR
2. {
3.     INT32    idCapa;
4.     INT32    nValMin;
5.     INT32    nValMax;
6.     INT32    nValDft;
7.     INT32    nValStep;
8. }TUCAM_CAPA_ATTR, *PTUCAM_CAPA_ATTR;
9.

```

接口 TUCAM_Capa_GetAttr 输入参数：

名称	描述
idCapa	[in] 相机性能 ID,参考 TUCAM_IDCAPA
nValMin	[out] 可取最小值
nValMax	[out] 可取最大值
nValDft	[out] 设备默认值
nValStep	[out] 有效值范围内，最小步进

5.2.6 属性的属性

```

1. typedef struct _tagTUCAM_PROP_ATTR
2. {
3.     INT32    idProp;
4.     INT32    nIdxChn;
5.     DOUBLE   dbValMin;
6.     DOUBLE   dbValMax;
7.     DOUBLE   dbValDft;
8.     DOUBLE   dbValStep;
9. }TUCAM_PROP_ATTR, *PTUCAM_PROP_ATTR;
10.

```

接口 TUCAM_Prop_GetAttr 输入参数：

名称	描述
idProp	[in] 属性 ID，参考 TUCAM_IDPROP
nIdxChn	[in/out] 当前通道索引，如果使用默认通道手动设置 0。

dbValMin	[out] 可取最小值
dbValMax	[out] 可取最大值
dbValDft	[out] 设备默认值
dbValStep	[out] 可取范围内最小步进

5.2.7 性能 / 属性值文本

```

1. typedef struct _tagTUCAM_VALUE_TEXT
2. {
3.     INT32    nID;
4.     DOUBLE   dbValue;
5.     PCHAR    pText;
6.     INT32    nTextSize;
7. }TUCAM_VALUE_TEXT, *PTUCAM_VALUE_TEXT;
8.

```

接口 TUCAM_Capa_GetValueText 和 TUCAM_Prop_GetValueText 输入参数：

名称	描述
nID	[in] 属性或性能 ID，参考 TUCAM_IDPROP / TUCAM_IDCAPA 值。
dbValue	[in] 性能或属性有效值
pText	[in/out] 指向值文本指针
nTextSize	[out] 值文本缓存大小

5.2.8 厂商属性的属性

```

1. typedef struct _tagTUCAM_VPROP_ATTR
2. {
3.     INT32    idVProp;
4.     INT32    nIdxChn;
5.     DOUBLE   dbValMin;
6.     DOUBLE   dbValMax;
7.     DOUBLE   dbValDft;
8.     DOUBLE   dbValStep;
9. }TUCAM_VPROP_ATTR, *PTUCAM_VPROP_ATTR;
10.

```

接口 TUCAM_Prop_GetValueText 输入参数：

名称	描述
idVProp	[in] 厂商属性 ID，参考 TUCAM_IDVPROP
nIdxChn	[in/out] 当前通道索引，如果使用默认通道手动设置 0。
dbValMin	[out] 可取最小值
dbValMax	[out] 可取最大值
dbValDft	[out] 设备默认值
dbValStep	[out] 可取范围内最小步进

5.2.9 处理图像属性

```

1. typedef struct _tagTUCAM_PPROP_ATTR
2. {
3.     INT32    idVProp;
4.     INT32    procType;
5.     DOUBLE   dbValMin;
6.     DOUBLE   dbValMax;
7.     DOUBLE   dbValDft;
8.     DOUBLE   dbValStep;
9. }TUCAM_PPROP_ATTR, *PTUCAM_PPROP_ATTR;
10.

```

接口 TUCAM_Prop_GetValueText 输入参数：

名称	描述
idPProp	[in] 图像处理 ID，参考 TUCAM_IDPPROP
procType	[in] 图像处理类型，参考 TUPROC_TYPE
dbValMin	[out] 可取最小值
dbValMax	[out] 可取最大值
dbValDft	[out] 设备默认值
dbValStep	[out] 可取范围内最小步进

5.2.10 ROI 的属性

```

1. typedef struct _tagTUCAM_ROI_ATTR
2. {
3.     BOOL     bEnable;
4.     INT32    nHOffset;

```

```

5.     INT32    nVOffset;
6.     INT32    nWidth;
7.     INT32    nHeight;
8. }TUCAM_ROI_ATTR, *PTUCAM_ROI_ATTR;
9.

```

接口 TUCAM_Cap_SetROI 和 TUCAM_Cap_GetROI 输入参数：

名称	描述
bEnable	[in/out] ROI 使能
nHOffset	[in/out] 水平方向偏移量，以像素为单位。默认为 4 倍数步进为 4
nVOffset	[in/out] 垂直方向偏移量，以像素为单位。默认为 4 倍数步进为 4
nWidth	[in/out] ROI 宽度。默认为 4 倍数步进为 4，部分相机为 8 倍数步进为 8
nHeight	[in/out] ROI 高度。默认为 4 倍数步进为 4

5.2.11 区域计算属性

```

1. typedef struct _tagTUCAM_CALC_ROI_ATTR
2. {
3.     BOOL    bEnable;
4.     INT32    idCalc;
5.     INT32    nHOffset;
6.     INT32    nVOffset;
7.     INT32    nWidth;
8.     INT32    nHeight;
9. }TUCAM_CALC_ROI_ATTR, *PTUCAM_CALC_ROI_ATTR;
10.

```

接口 TUCAM_Calc_SetROI 和 TUCAM_Calc_GetROI 输入参数：

名称	描述
bEnable	[in/out] ROI 使能
idCalc	[in] 区域计算类型，参考 TUCAM_IDCROI
nHOffset	[in/out] 水平方向偏移量，以像素为单位
nVOffset	[in/out] 垂直方向偏移量，以像素为单位
nWidth	[in/out] ROI 宽度
nHeight	[in/out] ROI 高度

5.2.12 触发的属性

```

1. typedef struct _tagTUCAM_TRIGGER_ATTR
2. {
3.     INT32    nTgrMode;
4.     INT32    nExpMode;
5.     INT32    nEdgeMode;
6.     INT32    nDelayTm;
7.     INT32    nFrames;
8.     INT32    nBufFrames;
9. }TUCAM_TRIGGER_ATTR, *PTUCAM_TRIGGER_ATTR;

```

接口 TUCAM_Cap_SetTrigger 和 TUCAM_Cap_GetTrigger 输入参数：

名称	描述
nTgrMode	[in/out] 触发模式参考值 TUCAM_CAPTURE_MODES
nExpMode	[in/out] 曝光模式[0, 1], 0: 曝光时间, 1: 电平宽度, 参考 TUCAM_TRIGGER_EXP
nEdgeMode	[in/out] 边沿激发模式[0, 1], 1: 上升沿, 0: 下降沿, 参考 TUCAM_TRIGGER_EDGE
nDelayTm	[in/out] 触发延迟时间, 单位毫秒
nFrames	[in/out] 一次触发输出帧数
nBufFrames	[in/out] 缓冲区中有多少帧, 根据电脑空闲 70% 内存大小限定, 不宜设置太大否则释放内存需要较长时间 (建议设置帧率的大小)

5.2.13 触发输出的属性

```

1. typedef struct _tagTUCAM_TRGOUT_ATTR
2. {
3.     INT32    nTgrOutPort;
4.     INT32    nTgrOutMode;
5.     INT32    nEdgeMode;
6.     INT32    nDelayTm;
7.     INT32    nWidth;
8. }TUCAM_TRGOUT_ATTR, *PTUCAM_TRGOUT_ATTR;
9.

```

接口 TUCAM_Cap_SetTriggerOut 和 TUCAM_Cap_GetTriggerOut 输入参数：

名称	描述
nTgrOutPort	[in/out] 触发输出口, 参考 TUCAM_OUTPUTTRG_PORT
nTgrOutMode	[in/out] 触发输出模式, 参考 TUCAM_OUTPUTTRG_KIND

nEdgeMode	[in/out] 边沿激发模式
nDelayTm	[in/out] 触发输出延迟时间
nWidth	[in/out] 触发输出脉宽

5.2.14 任意 Bin 属性

```

1. typedef struct _tagTUCAM_BIN_ATTR
2. {
3.     BOOL    bEnable;
4.     INT32   nMode;
5.     INT32   nWidth;
6.     INT32   nHeight;
7. }TUCAM_TRGOUT_ATTR, *PTUCAM_TRGOUT_ATTR;
8.

```

接口 TUCAM_Cap_SetBIN 和 TUCAM_Cap_GetBIN 输入参数：

名称	描述
bEnable	[in/out] 任意 bin 使能
nMode	[in/out] 任意 bin 模式 0：累加 Bin、1：平均 Bin
nWidth	[in/out] 任意 bin 宽度
nHeight	[in/out] 任意 bin 高度

5.2.15 帧结构

```

1. typedef struct _tagTUCAM_FRAME
2. {
3.     CHAR szSignature[8];
4.     USHORT usHeader;
5.     USHORT usOffset;
6.     USHORT usWidth;
7.     USHORT usHeight;
8.     UINT32 uiWidthStep;
9.     UCHAR ucDepth;
10.    UCHAR ucFormat;
11.    UCHAR ucChannels;
12.    UCHAR ucElemBytes;
13.    UCHAR ucFormatGet;

```



```

14.     UINT32 uiIndex;
15.     UINT32 uiImgSize;
16.     UINT32 uiRsdSize;
17.     UINT32 uiHstSize;
18.     PCHAR pBuffer;
19. }TUCAM_FRAME, *PTUCAM_FRAME;
20.

```

TUCAM_Buf_WaitForFrame 输入参数:

名称	描述
szSignature[8]	[out] 版权信息
usHeader	[out] 帧头部大小
usOffset	[out] 帧数据偏移大小
usWidth	[out] 水平像素数
usHeight	[out] 垂直像素数
uiWidthStep	[out] 帧图像的宽度步长
ucDepth	[out] 帧图像数据位深
ucFormat	[out] 帧图像数据格式
ucChannels	[out] 帧图像的通道数
ucElemBytes	[out] 帧图像的数据字节数
ucFormatGet	[in/out] 需要获取的图像格式。参考 TUFrm_FORMATS
uiIndex	[out] 帧图像序列号(保留)
uiImgSize	[out] 帧图像数据的大小
uiRsdSize	[in] 需要获取的帧数（预留参数目前只能设置 1）
uiHstSize	[out] 帧图像保留字段
pBuffer	[out] 指向帧数据缓存的指针

pBuffer 数据排列如下，可以根据帧头部大小获取到每一帧的头部信息，也可以根据帧数据偏移获取到图像数据，进行绘制展示：



5.2.16 文件保存

```
1. typedef struct _tagTUCAM_FILE_SAVE
2. {
3.     INT32    nSaveFmt;
4.     PCHAR    pstrSavePath;
5.     PTUCAM_FRAME pFrame;
6. }TUCAM_FILE_SAVE, *PTUCAM_FILE_SAVE;
7.
```

TUCAM_File_SaveImage 图像保存用到这个结构体：

名称	描述
nSaveFmt	[in] 保存图像的格式 TUIMG_FORMATS
pstrSavePath	[in] 保存图像路径(不包括扩展名)
pFrame	[in] 指向帧数据指针

5.2.17 录像保存

```
1. typedef struct _tagTUCAM_REC_SAVE
2. {
3.     INT32    nCodec;
4.     PCHAR    pstrSavePath;
5.     float    fFps;
6. }TUCAM_REC_SAVE, *PTUCAM_REC_SAVE;
```

接口 TUCAM_Rec_Start 输入参数：

名称	描述
nCodec	[in] 编码解码类型，默认为 0
pstrSavePath	[in] 保存路径(包含文件名)
fFps	[in] 需要保存的帧率

5.2.18 寄存器读写

```

1. typedef struct _tagTUCAM_REG_RW
2. {
3.     INT32    nRegType;
4.     PCHAR    pBuf;
5.     INT32    nBufSize;
6. }TUCAM_REG_RW, *PTUCAM_REG_RW;
7.

```

接口 TUCAM_Reg_Read 和 TUCAM_Reg_Write 输入参数：

名称	描述
nRegType	[in] 读写寄存器类型，参考 TUREG_TYPE
pBuf	[in/out] 指向读写内容缓冲区指针
nBufSize	[in] 缓冲区大小

5.2.19 图像绘制初始化

```

1. typedef struct _tagTUCAM_DRAW_INIT
2. {
3. #ifdef TUCAM_TARGETOS_IS_WIN32;
4.     HWND    hWnd;
5. #endif;
6.     INT32    nMode;
7.     UCHAR    ucChannels;
8.     INT32    nWidth;
9.     INT32    nHeight;
10. }TUCAM_DRAW_INIT, *PTUCAM_DRAW_INIT;
11.

```

接口 TUCAM_Draw_Init 输入参数：

名称	描述
----	----

hWnd	[in] 绘制窗口句柄，仅支持 Windows 操作系统
nMode	[in] 是否使用硬件加速（预留 GPU 支持），默认值 TUDRAW_DFT
ucChannels	[in] 相机通道数
nWidth	[in] 绘制数据宽度
nHeight	[in] 绘制数据高度

5.2.20 图像绘制参数（仅 Windows）

```

1. typedef struct _tagTUCAM_DRAW
2. {
3.     INT32    nSrcX;
4.     INT32    nSrcY;
5.     INT32    nSrcWidth;
6.     INT32    nSrcHeight;
7.     INT32    nDstX;
8.     INT32    nDstY;
9.     INT32    nDstWidth;
10.    INT32    nDstHeight;
11.    PTUCAM_FRAME pFrame;
12. }TUCAM_DRAW, *PTUCAM_DRAW;

```

接口 TUCAM_Draw_Frame 输入参数：

名称	描述
nSrcX	[in/out] 源矩形左上角的 x 坐标（以像素为单位）
nSrcY	[in/out] 源矩形左上角的 y 坐标（以像素为单位）
nSrcWidth	[in/out] 源矩形的宽度（以像素为单位）
nSrcHeight	[in/out] 源矩形的高度（以像素为单位）
nDstX	[in/out] 目标矩形左上角的 x 坐标，以 MM_TEXT 客户端坐标表示。
nDstY	[in/out] 目标矩形左上角的 y 坐标，以 MM_TEXT 客户端坐标表示。
nDstWidth	[in/out] 目标矩形的宽度，以 MM_TEXT 客户端坐标表示。
nDstHeight	[in/out] 目标矩形的高度，以 MM_TEXT 客户端坐标表示。
pFrame	[in] 指向待绘制帧结构体指针

5.3 函数

5.3.1 API 初始化 / 反初始化

5.3.1.1 TUCAM_Api_Init 和 TUCAM_Api_Uninit

描述

卸载 TUCAM-API 库，包括释放驱动绑定和一些内部资源。整个程序结束时调用一次。

声明

```
TUCAMRET TUCAM_Api_Uninit ();
```

参数

无输入参数

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
-------------------	----------------

相关接口

TUCAM_Api_Uninit

5.3.2 打开、关闭相机

5.3.2.1 TUCAM_Dev_Open

描述

打开相机，调用后相机处于工作模式，可以响应其他接口的调用。在此之前要保证有相机存在，即要在调用 TUCAM_Api_Init 初始化之后。

声明

```
TUCAMRET TUCAM_Dev_Open (PTUCAM_OPEN pOpenParam);
```

参数

PTUCAM_OPEN pOpenParam 打开相机结构体指针，参考结构体 TUCAM_OPEN

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_PARAM	无效参数，当 pOpenParam 指针为空时
TUCAMRET_OUT_OF_RANGE	超出范围，当需要打开的相机索引号超出连接相机的范围时
TUCAMRET_FAILOPEN_CAMERA	打开相机失败
A	
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Close

5.3.2.2 TUCAM_Dev_Close

描述

关闭相机，调用后相机处于待机状态，不响应其他接口的调用。

声明

TUCAMRET TUCAM_Dev_Close ();

参数

无输入参数

错误代码

TUCAMRET TUCAM-API 未初始化

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open

5.3.2.3 TUCAM_Dev_GetInfo

描述

获取相机的相关信息，如 USB 口类型，相机产品号，API 版本号、固件版本号、相机类型等。在此之前要保证有相机存在，并且保证相机打开，即要在调用 TUCAM_Api_Init 初始化和调用 TUCAM_Api_Open 之后。

声明

TUCAMRET TUCAM_Dev_GetInfo (HDTUCAM hTUCam, PTUCAM_VALUE_INFO pInfo);

参数

HDTUCAM hTUCam	相机的句柄
PTUCAM_VALUE_INFO pInfo	相机信息值的结构体指针，参考 TUCAM_VALUE_INFO

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_PARAM	无效参数,当产品信息代码不存在时,参考 TUCAM_IDINFO
TUCAMRET_INVALID_CAMERA	无效的相机, 相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close

5.3.2.4 TUCAM_Dev_GetInfoEx

描述

获取相机的相关信息，如 USB 口类型，相机产品号，API 版本号、相机类型等。在此之前要保证有相机存在，并且保证接口初始化，即只需要在调用 TUCAM_Api_Init 之后，无需调用 TUCAM_Api_Open。仅支持 TUIDI_CAMERA_MODEL、TUIDI_BUS、TUIDI_VENDOR、TUIDI_PRODUCT、TUIDI_VERSION_API、TUIDI_BCDDEVICE。

声明

TUCAMRET TUCAM_Dev_GetInfoEx (UINT32 uilCam, PTUCAM_VALUE_INFO pInfo);

参数

UINT32 uilCam	相机的索引号
PTUCAM_VALUE_INFO pInfo	相机信息值的结构体指针，参考 TUCAM_VALUE_INFO

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_PARAM	无效参数,当产品信息代码不存在时,参考 TUCAM_IDINFO
TUCAMRET_INVALID_CAMERA	无效的相机, 相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit

5.3.3 性能获取和设置

5.3.3.1 TUCAM_Capa_GetAttr

描述

获取性能参数的属性值。获取的属性包含该参数的最小值、最大值、默认值和步长。具体支持的性能类型参考 TUCAM_IDCAPA。

声明

TUCAMRET TUCAM_Capa_GetAttr (HDTUCAM hTUCam, PTUCAM_CAPA_ATTR pAttr);

参数

HDTUCAM hTUCam	相机的句柄
PTUCAM_CAPA_ATTR pAttr	相机性能属性结构体指针，参考 TUCAM_CAPA_ATTR

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_IDCAPA	无效的性能代码，当性能代码不存在时，参考 TUCAM_IDCAPA
TUCAMRET_INVALID_VALUE	无效的值，当 pAttr 指针为空时
TUCAMRET_NOT_SUPPORT	当底层请求不支持时
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Capa_GetValue、TUCAM_Capa_SetValue、TUCAM_Capa_GetValueText

5.3.3.2 TUCAM_Capa_GetValue

描述

获取性能参数的当前属性值。具体支持的性能类型参考 TUCAM_IDCAPA。

声明

TUCAMRET TUCAM_Capa_GetValue (HDTUCAM hTUCam, INT32 nCapa, INT32 *pnVal);

参数

HDTUCAM hTUCam	相机的句柄
INT32 nCapa	相机性能属性 ID，参考 TUCAM_IDCAPA
INT32 *pnVal	返回当前值

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_IDCAPA	无效的性能代码，当性能代码不存在时，参考 TUCAM_IDCAPA
TUCAMRET_INVALID_VALUE	无效的值，当 pAttr 指针为空时
TUCAMRET_NOT_SUPPORT	当底层请求不支持时
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Capa_GetAttr、TUCAM_Capa_SetValue、TUCAM_Capa_GetValueText

5.3.3.3 TUCAM_Capa_SetValue

描述

设置性能参数的当前属性值。具体支持的性能类型参考 TUCAM_IDCAPA。

声明

TUCAMRET TUCAM_Capa_SetValue (HDTUCAM hTUCam, INT32 nCapa, INT32 nVal);

参数

HDTUCAM hTUCam	相机的句柄
INT32 nCapa	相机性能属性 ID，参考 TUCAM_IDCAPA
INT32 nVal	需要设置的当前值

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_IDCAPA	无效的性能代码，当性能代码不存在时，参考 TUCAM_IDCAPA
TUCAMRET_INVALID_VALUE	无效的值，当 pAttr 指针为空时
TUCAMRET_NOT_SUPPORT	当底层请求不支持时
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Capa_GetAttr、TUCAM_Capa_GetValue、TUCAM_Capa_GetValueText

5.3.3.4 TUCAM_Capa_GetValueText

描述

获取性能参数的当前属性值的文本信息。具体支持的性能类型参考 TUCAM_IDCAPA。

声明

```
TUCAMRET TUCAM_Capa_GetValueText (HDTUCAM hTUCam, PTUCAM_VALUE_TEXT pVal);
```

参数

HDTUCAM hTUCam	相机的句柄
PTUCAM_VALUE_TEXT pVal	获取性能参数的文本信息结构体指针， TUCAM_VALUE_TEXT

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_FAILURE	文本缓冲区大小为 0 或 pText 指针为空时
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Capa_GetAttr、TUCAM_Capa_SetValue、TUCAM_Capa_GetValue

5.3.4 属性获取和设置

5.3.4.1 TUCAM_Prop_GetAttr

描述

获取属性参数的属性值。获取的属性包含该参数的最小值、最大值、默认值和步长。具体支持的性能类型参考 TUCAM_IDPROP。

声明

```
TUCAMRET TUCAM_Prop_GetAttr (HDTUCAM hTUCam, PTUCAM_PROP_ATTR pAttr);
```

参数

HDTUCAM hTUCam	相机的句柄
PTUCAM_PROP_ATTR pAttr	相机性能属性结构体指针，参考 TUCAM_PROP_ATTR

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_IDPROP	无效的性能代码，当性能代码不存在时，参考

TUCAMRET_INVALID_VALUE	TUCAM_IDPROP 无效的值，当 pAttr 指针为空时
TUCAMRET_NOT_SUPPORT	当底层请求不支持时
TUCAMRET_OUT_OF_RANGE	需要获取的通道超出范围时
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Prop_GetValue、TUCAM_Prop_SetValue、TUCAM_Prop_GetValueText

5.3.4.2 TUCAM_Prop_GetValue

描述

获取属性参数的当前属性值。具体支持的性能类型参考 TUCAM_IDPROP。

声明

TUCAMRET TUCAM_Prop_GetValue (HDTUCAM hTUCam, INT32 nProp, DOUBLE *pdbVal, INT32 nChn = 0);

参数

HDTUCAM hTUCam	相机的句柄
INT32 nProp	相机属性的 ID，参考 TUCAM_IDPROP
DOUBLE *pdbVal	返回当前值
INT32 nChn	需要获取的当前通道（默认为 0，黑白相机为 0）

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_IDPROP	无效的性能代码，当性能代码不存在时，参考 TUCAM_IDPROP
TUCAMRET_INVALID_VALUE	无效的值，当 pAttr 指针为空时
TUCAMRET_NOT_SUPPORT	当底层请求不支持时
TUCAMRET_OUT_OF_RANGE	需要获取的通道超出范围时
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Prop_GetAttr、TUCAM_Prop_SetValue、TUCAM_Prop_GetValueText

5.3.4.3 TUCAM_Prop_SetValue

描述

设置属性参数的当前属性值。具体支持的性能类型参考 TUCAM_IDPROP。

参数 nChn 默认为 0，彩色相机 TUIDP_CHNLGAIN 属性类型，为独立调节 RGB 通道信号强度的参数，nChn = 1 表示设置红色通道，nChn = 2 表示设置绿色通道，nChn = 3 表示设置蓝色通道。

声明

```
TUCAMRET TUCAM_Prop_SetValue (HDTUCAM hTUCam, INT32 nProp, DOUBLE dbVal, INT32 nChn = 0);
```

参数

HDTUCAM hTUCam	相机的句柄
INT32 nProp	相机属性的 ID，参考 TUCAM_IDPROP
DOUBLE dbVal	需要设置的当前值
INT32 nChn	需要获取的当前通道（默认为 0，黑白相机为 0）

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_IDPROP	无效的性能代码，当性能代码不存在时，参考 TUCAM_IDPROP
TUCAMRET_INVALID_VALUE	无效的值，当 pAttr 指针为空时
TUCAMRET_NOT_SUPPORT	当底层请求不支持时
TUCAMRET_OUT_OF_RANGE	需要获取的通道超出范围时
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Prop_GetAttr、TUCAM_Prop_GetValue、TUCAM_Prop_GetValueText

5.3.4.4 TUCAM_Prop_GetValueText

描述

获取属性参数的当前属性值的文本信息。具体支持的性能类型参考 TUCAM_IDPROP。

参数 nChn 默认为 0，彩色相机 TUIDP_CHNLGAIN 属性类型，为独立调节 RGB 通道信号强度的参数，nChn = 1 表示获取红色通道，nChn = 2 表示获取绿色通道，nChn = 3 表示获取蓝色通道。

声明

```
TUCAMRET TUCAM_Prop_GetValueText (HDTUCAM hTUCam, PTUCAM_VALUE_TEXT pVal);
```

参数

HDTUCAM hTUCam	相机的句柄
PTUCAM_VALUE_TEXT pVal	获取性能参数的文本信息结构体指针， TUCAM_VALUE_TEXT

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_FAILURE	文本缓冲区大小为 0 或 pText 指针为空时
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Prop_GetAttr、TUCAM_Prop_SetValue、TUCAM_Prop_GetValue

5.3.5 内存管理

5.3.5.1 TUCAM_Buf_Alloc

描述

分配内存空间用于数据捕获。当应用程序调用这个接口时，SDK 分配必要的内部缓冲区来缓冲图像采集。捕获不从这一时刻开始。开始采集，应用程序必须调用 TUCAM_Cap_Start 接口。如果缓冲区不再是必要的，应用程序应该调用 TUCAM_Buf_Release 接口来释放内部缓冲区。

声明

TUCAMRET TUCAM_Buf_Alloc (HDTUCAM hTUCam, PTUCAM_FRAME pFrame);

参数

HDTUCAM hTUCam	相机的句柄
PTUCAM_VALUE_TEXT pVal	图像帧结构的指针，参考 TUCAM_FRAME

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_PARAM	当 pFrame 指针为空时
TUCAMRET_EXCLUDED	当 TUCAM_Buf_Alloc 被调用，且未释放时
TUCAMRET_OUT_OF_RANGE	当需要获取的帧数超出范围时
TUCAMRET_NO_MEMORY	当内存不足时
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit

TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Release
TUCAM_Buf_AbortWait、TUCAM_Buf_WaitForFrame、TUCAM_Buf_CopyFrame
TUCAM_Cap_Start、TUCAM_Cap_Stop

5.3.5.2 TUCAM_Buf_Release

描述

释放用于数据捕获的内存空间。如果在捕获过程中调用该接口，这个接口将返回相机处于繁忙的状态。

声明

TUCAMRET TUCAM_Buf_Release (HDTUCAM hTUCam);

参数

HDTUCAM hTUCam	相机的句柄
----------------	-------

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_BUSY	相机处于繁忙状态
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc
TUCAM_Buf_AbortWait、TUCAM_Buf_WaitForFrame、TUCAM_Buf_CopyFrame
TUCAM_Cap_Start、TUCAM_Cap_Stop

5.3.5.3 TUCAM_Buf_AbortWait

描述

用于停止数据捕获时的等待。调用 TUCAM_Buf_WaitForFrame 进行数据捕获等待之后，使用该接口终止等待。

声明

TUCAMRET TUCAM_Buf_AbortWait (HDTUCAM hTUCam);

参数

HDTUCAM hTUCam	相机的句柄
----------------	-------

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release
TUCAM_Buf_WaitForFrame、TUCAM_Buf_CopyFrame
TUCAM_Cap_Start、TUCAM_Cap_Stop

5.3.5.4 TUCAM_Buf_WaitForFrame

描述

用于等待数据捕获完成。通过调用 TUCAM_Buf_Alloc 分配来的空间，来获取捕获到的帧数据。必须在调用 TUCAM_Cap_Start 开始捕获之后使用，否则会返回未准备的状态。

该函数属于阻塞函数，直至数据捕获完成或调用 TUCAM_Buf_AbortWait 终止。

帧结构体中 uiRsdSize 设置需要捕获的帧数，此参数是预留参数，当前只能设置 1。

注意：返回的帧结构 pBuffer 的数据排列，是帧头部(usHeader)+图像数据(uiImgSize) + 保留位(uiHstSize)。如果是多帧返回则按此顺序往后排列。为了不影响数据速度导致丢帧，建议开一个子线程获取数据流，建议获取流后不要做耗时长的数据处理。

声明

TUCAMRET TUCAM_Buf_WaitForFrame (HDTUCAM hTUCam, PTUCAM_FRAME pFrame, INT32 nTimeOut = 1000);

参数

HDTUCAM hTUCam	相机的句柄
PTUCAM_FRAME pFrame	帧结构体的指针
INT32 nTimeOut	超时退出的时间，默认 1S

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_NOT_READY	当未调用 TUCAM_Cap_Start 开始捕获时
TUCAMRET_NO_MEMORY	当未调用 TUCAM_Buf_Alloc 创建内存空间时
TUCAMRET_NO_RESOURCE	当 pFrame 指针为空时
TUCAMRET_OUT_OF_RANGE	当获取的帧数大于 1 时且获取的格式和 TUCAM_Buf_Alloc 不同
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release
TUCAM_Buf_AbortWait、TUCAM_Buf_CopyFrame
TUCAM_Cap_Start、TUCAM_Cap_Stop

5.3.5.5 TUCAM_Buf_CopyFrame

描述

用于等待数据捕获完成之后拷贝不同于 TUCAM_Buf_Alloc 分配的图像格式数据。必须在 TUCAM_Buf_WaitForFrame 返回之后调用，否则无法获取正确的图像数据。

例如：分配的图像格式为 TUCAM_FMT_RGB888，通过该函数可以拷贝其他格式的数据（如 TUCAM_FMT_RAW），此接口无法拷贝大于 1 帧的数据即帧结构中的 uiRsdSize 不能大于 1。

注意：返回的帧结构 pBuffer 的数据排列，是帧头部(usHeader)+图像数据(uiImgSize) +保留位(uiHstSize)。不支持多帧数据返回。

声明

TUCAMRET TUCAM_Buf_CopyFrame (HDTUCAM hTUCam, PTUCAM_FRAME pFrame);

参数

HDTUCAM hTUCam	相机的句柄
PTUCAM_FRAME pFrame	帧结构体的指针

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_NOT_READY	当未调用 TUCAM_Cap_Start 开始捕获时
TUCAMRET_NO_MEMORY	当未调用 TUCAM_Buf_Alloc 创建内存空间时
TUCAMRET_NO_RESOURCE	当 pFrame 指针为空时
TUCAMRET_OUT_OF_RANGE	当获取的帧数大于 1 时且获取的格式和 TUCAM_Buf_Alloc 不同
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release
TUCAM_Buf_AbortWait、TUCAM_Buf_WaitForFrame
TUCAM_Cap_Start、TUCAM_Cap_Stop

5.3.5.6 TUCAM_Buf_DataCallBack

描述

用于获取原始数据流注册回调函数，注册回调之后通过调用 TUCAM_Buf_Alloc 分配来的空间，来获取捕获到的帧数据。必须在调用 TUCAM_Cap_Start 开始捕获之后使用 TUCAM_Buf_GetData 函数便可获取原始数据流。

声明

```
TUCAMRET TUCAM_Buf_DataCallBack(HDTUCAM hTUCam, BUFFER_CALLBACK cbBuffer, void *pUserContext);
```

参数

HDTUCAM hTUCam	相机的句柄
BUFFER_CALLBACK cbBuffer	获取原始数据流回调函数
void *pUserContext	用户数据回调函数注册

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release
TUCAM_Buf_GetData
TUCAM_Cap_Start、TUCAM_Cap_Stop

5.3.5.7 TUCAM_Buf_GetData

描述

用于获取原始数据流函数，当使用 TUCAM_Buf_DataCallBack 函数。

注册回调之后通过调用 TUCAM_Buf_Alloc 分配来的空间，来获取捕获到的帧数据。必须在调用 TUCAM_Cap_Start 开始捕获之后使用。

注意：为了不影响数据速度导致丢帧，建议获取流后不要做耗时的数据处理

声明

```
TUCAMRET TUCAM_Buf_GetData (HDTUCAM hTUCam, PTUCAM_RAWIMG_HEADER pFrame);
```

参数

HDTUCAM hTUCam	相机的句柄
PTUCAM_RAWIMG_HEADER pFrame	原始帧结构体指针

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_FAILURE	获取帧失败
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release
TUCAM_Buf_DataCallBack
TUCAM_Cap_Start、TUCAM_Cap_Stop

5.3.6 捕获控制

5.3.6.1 TUCAM_Cap_SetROI

描述

用于设定图像的感兴趣区域，以左上角为坐标原点。设置的水平偏移量、垂直偏移量、宽度和高度必须为 4 的倍数。（11BIT 模式下宽度大于 32，宽度乘高度必须为 32 的倍数）

声明

TUCAMRET TUCAM_Cap_SetROI (HDTUCAM hTUCam, TUCAM_ROI_ATTR roiAttr);

参数

HDTUCAM hTUCam	相机的句柄
TUCAM_ROI_ATTR roiAttr	ROI 属性结构体的对象

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_NOT_SUPPORT	不支持 ROI 设置
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release
TUCAM_Buf_AbortWait、TUCAM_Buf_WaitForFrame
TUCAM_Cap_Start、TUCAM_Cap_Stop
TUCAM_Cap_GetROI

5.3.6.2 TUCAM_Cap_GetROI

描述

用于获取图像的感兴趣区域，以左上角为坐标原点。设置的水平偏移量、垂直偏移量、宽度和高度必须为 4 的倍数。（11BIT 模式下宽度大于 32,宽度乘高度必须为 32 的倍数）

捕获不从这一时刻开始。开始采集调用 TUCAM_Cap_Start 接口之后。

声明

```
TUCAMRET TUCAM_Cap_GetROI (HDTUCAM hTUCam, PTUCAM_ROI_ATTR pRoiAttr);
```

参数

HDTUCAM hTUCam	相机的句柄
PTUCAM_ROI_ATTR pRoiAttr	ROI 属性结构体的指针

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_NOT_SUPPORT	不支持 ROI 设置
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release
TUCAM_Buf_AbortWait、TUCAM_Buf_WaitForFrame
TUCAM_Cap_Start、TUCAM_Cap_Stop
TUCAM_Cap_SetROI

5.3.6.3 TUCAM_Cap_SetTrigger

描述

用于设定触发的属性。捕获不从这一时刻开始，开始采集调用 TUCAM_Cap_Start 接口之后。

曝光模式：

TUCTE_EXPTM	表示曝光时间由软件设定
TUCTE_WIDTH	表示曝光时间由输入电平宽度设定

激发边沿模式：

TUCTD_RISING	表示触发信号为上升沿有效
TUCTD_FAILING	表示触发信号为下降沿有效

帧数：表示接收一个触发信号后，拍摄多少张图像，每张图像的曝光时间是相同的，取决于软件设定。（选择电平宽度时，该功能无效。）

延迟：表示接收到一个触发信号后，可以设置多长的延迟时间才使相机进行触发曝光。

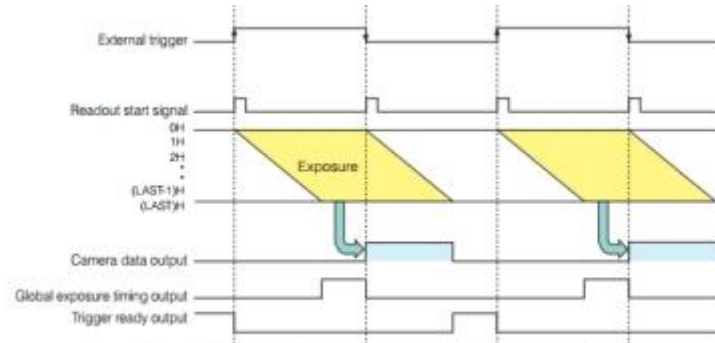


Fig. 19: Normal level trigger mode (rising edge)

触发模式支持的参数：

模式	TUCAM_TRIGGER_EXP	TUCAM_TRIGGER_EDGE	延时	帧数
标准触发	支持	支持	支持	支持
同步触发	支持	支持	不支持	不支持
全局触发	不支持	支持	不支持	不支持
软件触发	不支持	不支持	不支持	不支持

同步触发：即同步取图，第一次触发启动，第二次触发输出同步图像。

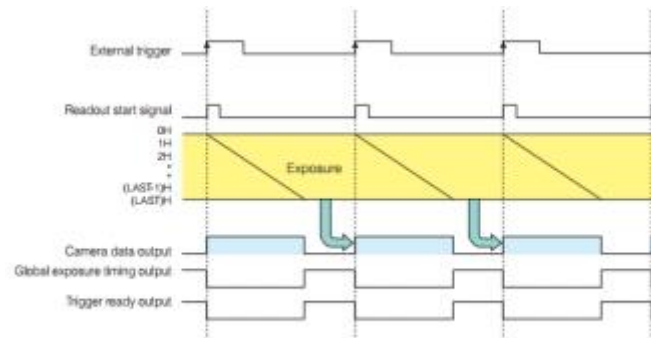


Fig. 21: Normal synchronous readout trigger mode (rising edge)

全局触发：一般用于光源可控的场景。

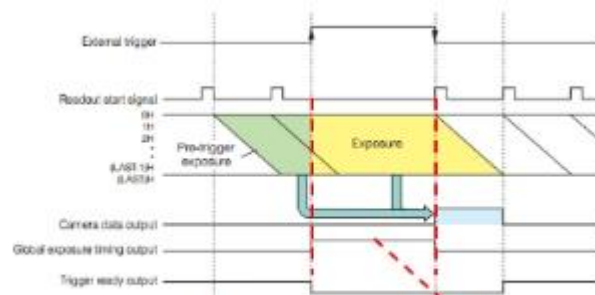


Fig. 20: Global exposure level trigger mode

有光照区域

软件触发：通过软件下发命令模拟触发信号。

声明

TUCAMRET TUCAM_Cap_SetTrigger (HDTUCAM hTUCam, TUCAM_TRIGGER_ATTR tgrAttr);

参数

HDTUCAM hTUCam	相机的句柄
TUCAM_TRIGGER_ATTR tgrAttr	触发属性结构体的对象

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release
TUCAM_Buf_AbortWait、TUCAM_Buf_WaitForFrame
TUCAM_Cap_Start、TUCAM_Cap_Stop
TUCAM_Cap_GetTrigger、TUCAM_Cap_DoSoftwareTrigger

5.3.6.4 TUCAM_Cap_GetTrigger

描述

用于获取触发的属性。

声明

TUCAMRET TUCAM_Cap_GetTrigger (HDTUCAM hTUCam, PTUCAM_TRIGGER_ATTR pTgrAttr);

参数

HDTUCAM hTUCam	相机的句柄
PTUCAM_TRIGGER_ATTR pTgrAttr	触发属性结构体的指针

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release

TUCAM_Buf_AbortWait、TUCAM_Buf_WaitForFrame
TUCAM_Cap_Start、TUCAM_Cap_Stop
TUCAM_Cap_SetTrigger

5.3.6.5 TUCAM_Cap_DoSoftwareTrigger

描述

执行软件触发命令。

声明

TUCAMRET TUCAM_Cap_DoSoftwareTrigger(HDTUCAM hTUCam);

参数

HDTUCAM hTUCam	相机的句柄
----------------	-------

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_FAILURE	执行触发命令失败
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release
TUCAM_Buf_AbortWait、TUCAM_Buf_WaitForFrame
TUCAM_Cap_Start、TUCAM_Cap_Stop
TUCAM_Cap_SetTrigger

5.3.6.6 TUCAM_Cap_SetTriggerOut

描述

用于设定触发输出的属性。

声明

TUCAMRET TUCAM_Cap_SetTriggerOut (HDTUCAM hTUCam, TUCAM_TRGOUT_ATTR tgroutAttr);

参数

HDTUCAM hTUCam	相机的句柄
TUCAM_TRGOUT_ATTR tgroutAttr	触发输出属性结构体的对象

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_NOT_SUPPORT	不支持设置
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release

5.3.6.7 TUCAM_Cap_GetTriggerOut**描述**

用于获取触发输出的属性。

声明

TUCAMRET TUCAM_Cap_GetTriggerOut (HDTUCAM hTUCam, PTUCAM_TRGOUT_ATTR pTgrOutAttr);

参数

HDTUCAM hTUCam	相机的句柄
PTUCAM_TRGOUT_ATTR pTgrOutAttr	触发输出属性结构体指针

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_NOT_SUPPORT	不支持设置
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release

5.3.6.8 TUCAM_Cap_Start**描述**

开始进行数据捕获。在开始捕获之前，应使用 TUCAM_Buf_alloc 准备捕获缓冲区，如果使用 TUCAM_SEQUENCE 模式将持续捕获，直到调用 TUCAM_Cap_Stop 被调用。适用配置好感兴趣区域和触发模式。

声明

TUCAMRET TUCAM_Cap_Start(HDTUCAM hTUCam, UINT32 uiMode);

参数

HDTUCAM hTUCam	相机的句柄
UINT32 uiMode	相机捕获的模式, 参考 TUCAM_CAPTURE_MODES

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_FAILOPEN_BULKIN	打开相机捕获失败
TUCAMRET_INVALID_CAMERA	无效的相机, 相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release
TUCAM_Buf_AbortWait、TUCAM_Buf_WaitForFrame
TUCAM_Cap_Start
TUCAM_Cap_SetTrigger、TUCAM_SetROI

5.3.6.9 TUCAM_Cap_Stop

描述

停止进行数据捕获, 例如改变分辨率、ROI、位深、相机采集模式等需要先调用此接口停流。

声明

TUCAMRET TUCAM_Cap_Stop (HDTUCAM hTUCam);

参数

HDTUCAM hTUCam	相机的句柄
----------------	-------

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_FAILOPEN_BULKIN	打开相机捕获失败
TUCAMRET_INVALID_CAMERA	无效的相机, 相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release

TUCAM_Buf_AbortWait、TUCAM_Buf_WaitForFrame
TUCAM_Cap_Stop

5.3.7 文件管理

5.3.7.1 TUCAM_File_SaveImage

描述

对帧数据进行保存。

声明

TUCAMRET TUCAM_File_SaveImage (HDTUCAM hTUCam, TUCAM_FILE_SAVE fileSave);

参数

HDTUCAM hTUCam	相机的句柄
TUCAM_FILE_SAVE fileSave	文件保存结构体

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_PARAM	输入的参数无效
TUCAMRET_INVALID_PATH	输入的路径不存在
TUCAMRET_FAILURE	文件保存失败
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release
TUCAM_Buf_WaitForFrame、TUCAM_Buf_CopyFrame

5.3.7.2 TUCAM_Rec_Start

描述

打开录像文件，对帧数据进行录像保存，此时并未写入数据。设置的帧率需要大于 1fps，不足 1fps 的将按 1fps 来创建录像文件。

声明

TUCAMRET TUCAM_Rec_Start(HDTUCAM hTUCam, TUCAM_REC_SAVE recSave);

参数

HDTUCAM hTUCam	相机的句柄
----------------	-------

TUCAM_REC_SAVE recSave 录像文件保存结构体

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_INVALID_PARAM	输入的参数无效
TUCAMRET_INVALID_PATH	输入的路径不存在
TUCAMRET_FAILOPEN_FILE	文件打开失败
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release
TUCAM_Buf_WaitForFrame
TUCAM_Cap_Start、TUCAM_Cap_Stop
TUCAM_Rec_Stop、TUCAM_Rec_AppendFrame

5.3.7.3 TUCAM_Rec_AppendFrame

描述

将图像数据写入文件，在 TUCAM_Buf_WaitForFrame 调用该接口。

声明

TUCAMRET TUCAM_Rec_AppendFrame(HDTUCAM hTUCam, PTUCAM_FRAME pFrame);

参数

HDTUCAM hTUCam	相机的句柄
PTUCAM_FRAME pFrame	帧结构体的指针

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_NOT_READY	未调用 TUCAM_Rec_Start 接口
TUCAMRET_OUT_OF_RANGE	写入的图像宽度和高度与创建时不一致
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release
TUCAM_Buf_WaitForFrame
TUCAM_Cap_Start、TUCAM_Cap_Stop

TUCAM_Rec_Start、TUCAM_Rec_Stop

5.3.7.4 TUCAM_Rec_Stop

描述

关闭录像文件，此时调用 TUCAM_Rec_AppendFrame 将无法写入数据。

声明

TUCAMRET TUCAM_Rec_Stop (HDTUCAM hTUCam);

参数

HDTUCAM hTUCam 相机的句柄

错误代码

TUCAMRET_NOT_INIT TUCAM-API 未初始化
TUCAMRET_INVALID_CAMERA 无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Buf_Alloc、TUCAM_Buf_Release
TUCAM_Buf_WaitForFrame
TUCAM_Cap_Start、TUCAM_Cap_Stop
TUCAM_Rec_Start、TUCAM_Rec_AppendFrame

5.3.8 扩展控制

5.3.8.1 TUCAM_Reg_Read

描述

读取寄存器的内容。读取的类型参考 TUREG_TYPE。

声明

TUCAMRET TUCAM_Reg_Read (HDTUCAM hTUCam, TUCAM_REG_RW regRW);

参数

HDTUCAM hTUCam 相机的句柄
TUCAM_REG_RW regRW 寄存器读写结构体

错误代码

TUCAMRET_NOT_INIT TUCAM-API 未初始化

TUCAMRET_NO_MEMORY	传入的缓冲区未分配内存空间
TUCAMRET_NOT_SUPPORT	不支持该类型读取
TUCAMRET_INVALID_IDPARAM	无效的类型，参考 TUREG_TYPE
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Reg_Write

5.3.8.2 TUCAM_Reg_Write**描述**

写入寄存器的内容。写入的类型参考 TUREG_TYPE。

声明

```
TUCAMRET TUCAM_Reg_Write(HDTUCAM hTUCam, TUCAM_REG_RW regRW);
```

参数

HDTUCAM hTUCam	相机的句柄
TUCAM_REG_RW regRW	寄存器读写结构体

错误代码

TUCAMRET_NOT_INIT	TUCAM-API 未初始化
TUCAMRET_NO_MEMORY	传入的缓冲区未分配内存空间
TUCAMRET_NOT_SUPPORT	不支持该类型读取
TUCAMRET_INVALID_IDPARAM	无效的类型，参考 TUREG_TYPE
TUCAMRET_INVALID_CAMERA	无效的相机，相机句柄不存在时

相关接口

TUCAM_Api_Init、TUCAM_Api_Uninit
TUCAM_Dev_Open、TUCAM_Dev_Close
TUCAM_Reg_Read

6. 常见问题解答（FAQ）**6.1 问题排查思路**

1. 基于 SDK 开发的程序有异常，建议先运行 SamplePro 或是 Mosaic 软件，查看相应功能是否能够正常运行。

2. 如果 SamplePro 或 Mosaic 软件正常，而开发程序异常，建议重点排查二次开发的程序代码。
3. 如果 SamplePro 或 Mosaic 也异常，建议查看如下常见问题解决方法，看是否有帮助。
4. 如果以上排查都不能解决，建议请记录相关问题现象和保存图片数据、以及我们 SamplePro 或 Mosaic 与 SDK 的文件版本和产品版本号，联系本公司的技术支持同事获取帮助支持。

6.2 常见问题解决方法

问题 1：用 SDK 开发存图异常。

问题原因：TUCAM_FRAME 结构体中的 pBuffer 需要偏置到 usHeader 头部长度（当前长度是 1024Bytes）后才是真正的帧数据指针。

解决方法：uchar *pBuf = m_frame.pBuffer+m_frame.usHeader，即 pBuf 指针才是帧数据指针。

问题 2：用 TUCAM_Buf_GetData 获取到的 TUCAM_RAWIMG_HEADER 结构体帧数据偏移头部图像异常。

问题原因：TUCAM_RAWIMG_HEADER 中的 plmgData 是没有头部的，直接就是图像数据了，所以不必要偏移头部。

解决方法：直接从 plmgData 指针拿到的数据就是帧数据。

问题 3：设置垂直镜像返回 TUCAMRET_NO_RESOURCE 问题。

问题原因：由于垂直镜像是软件层做的，即需要先 CaptureStart 后才能设置生效。

解决方法：先调用 TUCAM_Cap_Start 后，再做垂直镜像。

问题 4：在 CaptureStart 后设置垂直镜像仍是无效。

问题原因：由于垂直镜像是软件层做的，只是针对 TUCAM_Buf_WaitForFrame 获取数据才生效，对 TUCAM_Buf_GetData 接口获取是原始图是无效。

解决方法：可通过 TUCAM_Buf_WaitForFrame 获取数据做显示。

问题 5：调用 TUCAM_Buf_Alloc 后报错（0x80000204）。

问题原因：由于多次重复调用了此接口。

解决方法：此接口只调用一次即可，如果还要调需先调用 TUCAM_Buf_Release 后再调。

问题 6：设置自动色阶图像不生效。

问题原因：直方图计算没有开启，并且直方图开启需要在 CaptureStart 后才能设置生效。另外一方面注意设置参数接口是否用错了（TUCAM_Capa_SetValue、TUCAM_Prop_SetValue）。

解决方法：先开启直方图，然后设置自动色阶参数即可。

问题 7：16bit 相机为何拿到帧灰度值最大值是 255。

问题原因：我们帧数据都是按最小字节单位数据输出，16bit 帧需要两个字节拼接成一个像素，小端排列即低字节在前高字节在后。

解决方法：ushort *pBuf = (ushort*)(m_frame.pBuffer+m_frame.usHeader),接口获取到灰度值转换。

问题 8：设置曝光时间返回成功了但是数据流上没有效果。

问题原因：开启了自动曝光导致手动曝光设置无效。

解决方法：先将自动曝光接口设置为关闭状态，然后再来设置曝光时间即可生效。

问题 9：二次开发亮度和我们软件显示的亮度不一致问题。

问题原因：我们软件是显示有效位的高八位数据，还有曝光、增益、色阶、伽马、对比度、锐度等功能参数是否一致。

解决方法：相机功能参数值需要设置一致，界面显示数据有效位需要一致。

问题 10：TUCAM_Buf_WaitForFrame 函数获取图像超时失败。

问题原因：接口获取数据超时退出了。

解决方法：可以将超时时间适当的设大。

问题 11：TUCAM_Cap_Stop 是否一定要调用 TUCAM_Buf_Release 函数。

问题原因：不一定，如果有改变位深、分辨率、ROI 等功能，涉及到图像内存大小变化就一定要释放内存调用 TUCAM_Buf_Release 函数，其他方式非必要。

解决方法：满足图像内存大小变化的就一定要调用 TUCAM_Buf_Release 函数，其他方式非必要。

问题 12：TUCAM_Buf_GetData 接口获取图像为什么不带色阶、对比度、伽马。

问题原因：TUCAM_Buf_GetData 接口获取图像是原图不带软件层面做的图像处理，TUCAM_Buf_WaitForFrame 函数在 TUFRM_FMT_USUAL、TUFRM_FMT_RGB888 模式下就生效。

解决方法：快速获取原图可采用回调中 TUCAM_Buf_GetData 函数获取，需要图像处理就采用 TUCAM_Buf_WaitForFrame 函数获取数据。

问题 13：TUCAM_Cap_Stop 接口卡住问题。

问题原因：原因是在回调中调用用了 TUCAM_Cap_Stop API 接口，进入 SDK 内部取流线程事件无法停止。

解决方法：不要在回调函数内部调用 TUCAM_Cap_Stop 接口，避免卡住问题。

问题 14：关于旋转的补充说明。

问题原因：不知道如何进行顺时针 180°、270° 的旋转。

解决方法：1) 180°：TUIDC_ROTATE_R90 配置 0，TUIDC_ROTATE_L90 配置 0，TUIDC_HORIZONTAL 配置 1，TUIDC_VERTICAL 配置 1；

2) 270°：等价于逆时针旋转 90°，即 TUIDC_ROTATE_R90 配置 0，TUIDC_ROTATE_L90 配置 1，TUIDC_HORIZONTAL 配置 0，TUIDC_VERTICAL 配置 0。

问题 15：通过接口判断丢帧位置说明。

问题原因：因电脑性能达到极限或是硬盘存储带宽不够都会导致丢帧情况。

解决方法：帧位置定位可以通过接口拿到数据值来判断。首先是支持 TUIDC_ENABLETIMESTAMP 相机使能要被开启，然后第一点可通过 TUCAM_Buf_GetData 函数中 pFrame 数据结构体里面的 uiIndex 序列号是否连续来作为一项判断指标，或是通过 dbfTimeLast 稳定的帧时间来判别是否丢帧，如果是丢帧了一定是成倍增长的。另外一方面如果数据流接口给的数据流数量是够的，但是存到磁



盘上不够那就是磁盘写速需要提高，也可通过写下来数据上的 uiIndex 来判别哪些图没有被存储到。