

USB Camera SDK Manual

简介

USB相机兼容UVC (USB Video Class) 协议，支持Window7/8/10/11、Linux 等操作系统。目前可提供Windows/Mac/Linux 平台的相机SDK

部分相机从以下固件版本版本开始，获取范围接口有效，其余相机均有效 (M112U/M114U暂无)

U108_v1.2.0、U106_v1.2.0、U112_v1.2.0、U120_v1.2.0、M202U_v1.2.0、C312_v1.3.0、C313_v1.3.0、C314_v1.3.0

注：SDK提供相机控制、获取视频流、拍照录像的功能，不包含视频显示功能

Mac/Linux版本暂不支持M112U/M114U/M202U/U408

Mac版本需root权限，如开发的APP不可使用root权限，则需申请 `com.apple.vm.device-access` 权限

Linux版本无需root权限，但需拷贝udev文件夹中的 `99-slcam.rules` 至 `/etc/udev/rules.d` 路径下，`glibc >= 2.27`，目前无arm版本

UVC SDK文件结构

slcam为相机SDK库，FFmpeg为SDK依赖库，两者均为必须。支持多平台，库文件在对应平台路径下

sdk

└─**linux**(udev存放linux所需的相机规则文件)

```
| └─FFmpeg
|   └─include
|     └─lib
|       └─x64
|         └─x86
| └─slcam
|   └─include
|     └─lib
|       └─x64
|         └─x86
└─udev
```

└─**mac**(动态库版本支持x64和arm64)

```
| └─FFmpeg
|   └─include
|     └─lib
└─slcam
  └─include
    └─lib
```

└─**win**(SDL2为Demo使用的显示库，SDK不依赖，可不使用)

```
| └─FFmpeg
|   └─bin
|     └─x64
|       └─x86
| └─include
└─lib
```

```
|   ├──x64
|   └──x86
├──SDL2
|   ├──bin
|   |   ├──x64
|   |   └──x86
|   ├──include
|   └──lib
|       ├──x64
|       └──x86
└──slcam
    ├──bin
    |   ├──x64
    |   └──x86
    ├──include
    └──lib
        ├──x64
        └──x86
```

x86为32位版本，x64为64位版本，根据需要使用对应版本

本SDK依赖FFmpeg，版本为 n4.1.3，需要拷贝对应的动态库文件使用

cpp C++例子.本例子演示了枚举设备，打开设备，预览视频，抓拍图像，设置分辨率,多种图片格式(.bmp, .jpg, .png等)保存图像到文件，mp4格式录像，相机控制，多视频流演示（控制台和Qt）

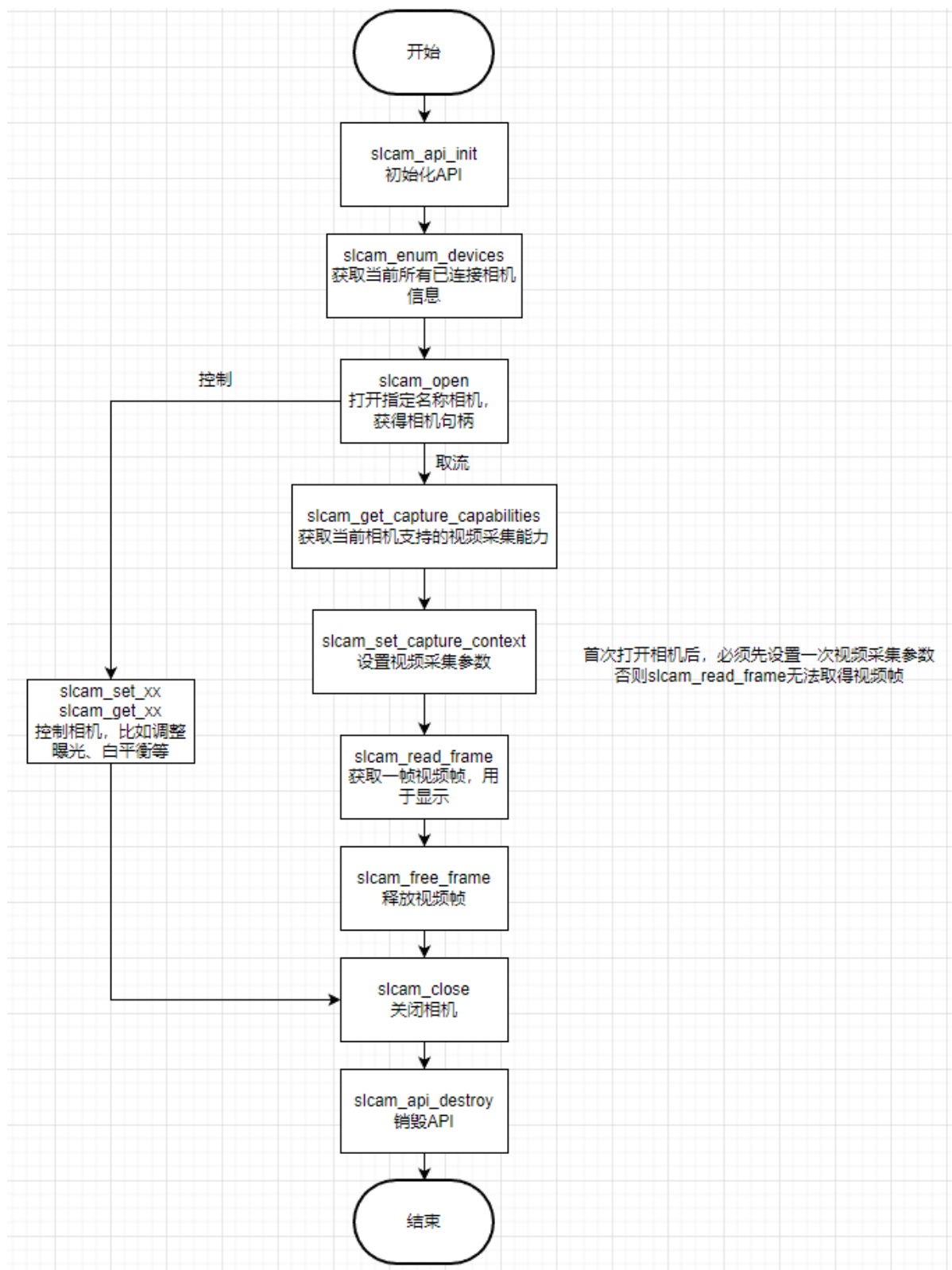
dotnet C# 例子，支持打开设备，预览视频，抓拍图像，保存图片到文件，设置白平衡。（RGB流 WinForms窗口，YUV流SDL窗口，多视频流演示）

python python 例子，支持打开设备，预览视频，拍照，白平衡和曝光设置。samples.py 为使用RGB方式显示视频的例子，pyqt5.py/pyqt6.py 是使用qt作为UI，使用YUV显示的例子

extras Labview SDK和demo程序，DirectShow驱动及demo程序，Twain驱动，驱动安装方式见文件夹中的说明

快速上手

调用流程如下图



流程说明

上图主要包含了相机控制和获取视频流的流程，两者互不影响，可以单独使用，也可共同使用

SDK API初始化

首先调用 `slcam_api_init` 进行初始化操作，主要是初始化取流和相机控制，整个程序开始时调用一次，不建议频繁调用

枚举相机

调用 `slcam_enum_devices` 获取当前已连接的所有相机信息，供用户自行选择

开启相机

调用 `slcam_open` 打开指定名称的相机，获取相机句柄，用于调用其他接口

控制相机

调用 `slcam_set_xx/slcam_get_xx` 来控制/获取相机的某些参数，例如曝光、白平衡、聚焦等

取流

首先调用 `slcam_get_capture_capabilities` 获取相机支持的视频采集能力，从中挑选合适的分辨率、格式

然后调用 `slcam_set_capture_context` 设置对应参数，首次打开相机后，必须设置一次后才能获取到视频帧

再调用 `slcam_read_frame` 获取一帧视频帧，可用于显示，或图像处理等其他用途

视频帧使用完毕后调用 `slcam_free_frame` 进行释放，避免内存泄漏

关闭相机

调用 `slcam_close` 关闭相机，调用完毕后，相机句柄将无效，不可再使用

SDK API销毁

当SDK使用完毕后，调用 `slcam_api_destroy` 释放SDK占用的资源，整个程序结束时调用一次，不建议频繁调用

规则

句柄相关内容

名称	描述
HSLcam	API上指定目标相机的句柄，几乎所有API都需要这个句柄作为参数。 <code>slcam_open</code> 函数提供该句柄， <code>slcam_close</code> 函数释放句柄，终止设备使用

函数相关内容

函数名称

所有函数都以“`slcam_`”作为前缀

函数前缀	函数名称	说明
<code>slcam_api_</code>	<code>init</code> , <code>destroy</code>	初始化，销毁API

函数前缀	函数名称	说明
slcam_log_	set_level, set_pattern, set_callback	设置日志等级, 设置日志格式, 设置日志回调函数
slcam_enum_	devices	枚举当前已连接相机
slcam_record_	start, append_frame, stop	录像
slcam_	open, close, get_capture_capabilities, set_capture_context, read_frame, free_frame, increase_focus, decrease_focus, upgrade, file_save_image	打开/关闭相机, 获取相机支持的采集视频能力集, 设置相机采集视频的参数, 读取/释放视频帧, 增加/减少一点聚焦值, 升级, 拍照
slcam_set_	focus_mode, focus, focus_region_v1, focus_region_v2, focus_region_v3, power_line_frequency, mirror, flip, hue, sharpness, saturation, contrast, zoom, zoom_relative, zoom_speed, exposure_mode, exposure_compensation, exposure_time, long_exposure_time, exposure_gain, exposure_gain_unit, gamma, white_balance_mode, white_balance_temperature, white_balance_component_red, white_balance_component_green, balance_component_blue, auto_exposure_region, auto_whitebalance_region, scene, drc, cac, ldci, bayer_shp, led, a3xx_all_led, a3xx_led, defog, user_id, windows_version, auto_focus_range, device_name, gamma_mode, gamma_bezier_curve, roi_region	设置相机相关属性

函数前缀	函数名称	说明
slcam_get_	focus_mode, focus_mode_ex, focus_state, focus, focus_range, focus_region_v1, focus_region_v2, focus_region_range, focus_region_v3, version, power_line_frequency, mirror, flip, hue, hue_range, sharpness, sharpness_range, saturation, saturation_range, contrast, contrast_range, zoom, ozoom_range, dzoom_range, zoom_relative, zoom_speed, zoom_speed_range, exposure_mode, exposure_compensation, exposure_compensation_range, exposure_time, exposure_time_range, long_exposure_time, long_exposure_time_range, exposure_gain, exposure_gain_range, exposure_gain_unit, gamma, gamma_range, white_balance_mode, white_balance_temperature, white_balance_temperature_range, white_balance_component_red, white_balance_component_red_range, white_balance_component_green, white_balance_component_green_range, balance_component_blue, balance_component_blue_range, auto_exposure_region, auto_whitebalance_region, scene, drc, cac, ldci, bayer_shp, led, a3xx_all_led, a3xx_led, led_info, defog, user_id, auto_focus_range, model_suffix, unique_id, gamma_mode, gamma_bezier_curve, roi_region	获取相机相关属性

函数参数

除了 slcam_api_init、slcam_api_destroy、slcam_log_set_level、slcam_open、slcam_enum_devices 等函数，几乎所有的 API 都需要 HSLCAM 句柄作为第一个参数。

函数返回

API 所有函数均由返回值，且均为int32_t类型。其值对应 SLCAMRET_ 开头的枚举类型。如果函数执行成功会返回 SLCAMRET_SUCCESS。建议在执行完函数后，判断返回结果值为 SLCAMRET_SUCCESS 再进行下一步操作，否则给出错误提示，方便定位问题。

接口参考

类型和常量

SLCAMRET 错误代码

名称	错误代码	描述
SLCAMRET_SUCCESS	0x00000000	成功代码，没有错误
SLCAMRET_FAILURE	0x80000000	调用接口失败

SLcamModel 支持的相机型号

名称	代码	描述
SLCAM_MODEL_M112U	0x0112	M112U
SLCAM_MODEL_M114U	0x0114	M114U
SLCAM_MODEL_M202U	0x0202	M202U
SLCAM_MODEL_M122	0x0122	M122
SLCAM_MODEL_M124	0x0124	M124
SLCAM_MODEL_M212	0x0212	M212
SLCAM_MODEL_C310	0xC310	C310
SLCAM_MODEL_C311	0xC311	C311
SLCAM_MODEL_C312	0xC312	C312
SLCAM_MODEL_C313	0xC313	C313
SLCAM_MODEL_C314	0xC314	C314
SLCAM_MODEL_U106	0x1106	U106
SLCAM_MODEL_U108	0x1108	U108
SLCAM_MODEL_U112	0x1112	U112
SLCAM_MODEL_U120	0x1120	U120
SLCAM_MODEL_A311U	0xA311	A311U
SLCAM_MODEL_A312U	0xA312	A312U
SLCAM_MODEL_A313	0xA313	A313
SLCAM_MODEL_A314	0xA314	A314
SLCAM_MODEL_A321U	0xA321	A321U
SLCAM_MODEL_A322U	0xA322	A322U
SLCAM_MODEL_A323	0xA323	A323
SLCAM_MODEL_A324	0xA324	A324
SLCAM_MODEL_U405	0x1405	U405
SLCAM_MODEL_U406	0x1406	U406
SLCAM_MODEL_U408	0x1408	U408
SLCAM_MODEL_U202	0x1202	U202
SLCAM_MODEL_U205	0x1205	U205

名称	代码	描述
SLCAM_MODEL_U208	0x1208	U208
SLCAM_MODEL_U304	0x1304	U304
SLCAM_MODEL_U305	0x1305	U305
SLCAM_MODEL_U306	0x1306	U306
SLCAM_MODEL_B201	0x00F9	B201
SLCAM_MODEL_L311	0x2311	L311
SLCAM_MODEL_L313	0x2313	L313
SLCAM_MODEL_L314	0x2314	L314
SLCAM_MODEL_R304	0x3314	R304
SLCAM_MODEL_UNSUPPORT	0xFFFF	未受支持的相机型号

SLcamLogLevel 日志等级

名称	代码	描述
SLCAM_LOG_TRACE	0x00	Trace等级，最低等级，所有日志均会输出
SLCAM_LOG_DEBUG	0x01	Debug等级
SLCAM_LOG_INFO	0x02	Info等级，默认为Info等级
SLCAM_LOG_WARNING	0x03	Warning等级，警告
SLCAM_LOG_ERROR	0x04	Error等级，错误
SLCAM_LOG_CRITICAL	0x05	Critical等级，严重错误
SLCAM_LOG_OFF	0x06	OFF等级，屏蔽所有日志

SLcamFocusMode 聚焦模式

名称	代码	描述
SLCAM_FOCUS_MODE_MANUAL	0x00	手动聚焦模式
SLCAM_FOCUS_MODE_AUTO	0x01	自动聚焦模式
SLCAM_FOCUS_MODE_ONCE	0x02	一键聚焦模式

一键聚焦 的意思是，触发后进行一次自动聚焦，对焦完毕后不再自动聚焦

SLcamExposureMode 曝光模式

名称	代码	描述
SLCAM_EXPOSURE_MODE_MANUAL	0x00	手动曝光模式
SLCAM_EXPOSURE_MODE_AUTO	0x01	自动曝光模式

SLcamWhiteBalanceMode 白平衡模式

名称	代码	描述
SLCAM_WHITE_BALANCE_MODE_MANUAL	0x00	手动白平衡模式
SLCAM_WHITE_BALANCE_MODE_AUTO	0x01	自动白平衡模式

SLcamPowerLineFrequency 抗频闪模式

名称	代码	描述
SLCAM_POWER_LINE_FREQUENCY_60HZ	0x00	60hz模式
SLCAM_POWER_LINE_FREQUENCY_50HZ	0x01	50hz模式

SLcamExposureGainUnit 增益单位

名称	代码	描述
SLCAM_EXPOSURE_GAIN_MAGNIFICATION	0x00	倍率
SLCAM_EXPOSURE_GAIN_DB	0x01	DB

SLcamVideoFormat 视频流格式

名称	代码	描述
SLCAM_VIDEO_FORMAT_UNKNOWN	-1	未知的视频流格式
SLCAM_VIDEO_FORMAT_I420	0x00	I420格式
SLCAM_VIDEO_FORMAT_J420	0x01	J420格式
SLCAM_VIDEO_FORMAT_IYUV	0x02	IYUV格式
SLCAM_VIDEO_FORMAT_RGB24	0x03	RGB24格式
SLCAM_VIDEO_FORMAT_BGR24	0x04	BGR24格式
SLCAM_VIDEO_FORMAT_ABGR	0x05	ABGR格式
SLCAM_VIDEO_FORMAT_ARGB	0x06	ARGB格式
SLCAM_VIDEO_FORMAT_RGBA	0x07	RGBA格式
SLCAM_VIDEO_FORMAT_BGRA	0x08	BGRA格式

名称	代码	描述
SLCAM_VIDEO_FORMAT_RGB565	0x09	RGB565格式
SLCAM_VIDEO_FORMAT_YUY2	0x0A	YUY2格式
SLCAM_VIDEO_FORMAT_UYVY	0x0C	UYVY格式
SLCAM_VIDEO_FORMAT_MJPEG	0x0D	MJPEG格式
SLCAM_VIDEO_FORMAT_H264	0x0E	H264格式
SLCAM_VIDEO_FORMAT_H265	0x0F	H265格式
SLCAM_VIDEO_FORMAT_NV12	0x10	NV12格式

SLcamPixelFormat 视频帧格式

名称	代码	描述
SLCAM_PIX_FORMAT_UNKNOWN	-1	未知的视频帧格式
SLCAM_PIX_FORMAT_I420	0x00	I420格式
SLCAM_PIX_FORMAT_J420	0x01	J420格式
SLCAM_PIX_FORMAT_IYUV	0x02	IYUV格式
SLCAM_PIX_FORMAT_RGB24	0x03	RGB24格式
SLCAM_PIX_FORMAT_BGR24	0x04	BGR24格式
SLCAM_PIX_FORMAT_ABGR	0x05	ABGR格式
SLCAM_PIX_FORMAT_ARGB	0x06	ARGB格式
SLCAM_PIX_FORMAT_RGBA	0x07	RGBA格式
SLCAM_PIX_FORMAT_BGRA	0x08	BGRA格式
SLCAM_PIX_FORMAT_RGB565	0x09	RGB565格式
SLCAM_PIX_FORMAT_YUY2	0x0A	YUY2格式
SLCAM_PIX_FORMAT_UYVY	0x0C	UYVY格式
SLCAM_PIX_FORMAT_NV12	0x0D	NV12格式
SLCAM_PIX_FORMAT_GRAY8	0x0E	灰度格式

SLcamDevUsbSpeed USB连接速率

名称	代码	描述
SLCAM_USB_LOW_SPEED	0x00	USB1.0
SLCAM_USB_FULL_SPEED	0x01	USB1.1

名称	代码	描述
SLCAM_USB_HIGH_SPEED	0x02	USB2.0
SLCAM_USB_SUPER_SPEED	0x03	USB3.0

SLcamImgFormat 图片保存格式

名称	代码	描述
SLCAM_IMG_FORMAT_UNKNOWN	-1	未知格式
SLCAM_IMG_FORMAT_PNG	0x00	PNG格式
SLCAM_IMG_FORMAT_JPG	0x01	JPG格式
SLCAM_IMG_FORMAT_BMP	0x02	BMP格式

结构体

```
1 typedef struct SLCamVideoResolution
2 {
3     int32_t width;
4     int32_t height;
5 } SLCamVideoResolution;
```

SLcamVideoResolution 视频分辨率

名称	描述
width	宽
height	高

```
1 typedef struct SLCamVideoFrame
2 {
3     uint8_t *data[4];
4     SLCamPixelFormat fmt;
5     int32_t width;
6     int32_t height;
7     int32_t linesize[4];
8     int64_t pts;
9 } SLCamVideoFrame;
```

SLcamVideoFrame 视频帧

名称	描述
data	视频数据
fmt	视频帧格式
width	宽

名称	描述
height	高
linesize	行宽，与stride、pitch含义相同，代表对应data中每一行的宽度
pts	时间戳

```

1  当fmt为
2  SLCAM_PIX_FORMAT_I420
3  SLCAM_PIX_FORMAT_J420
4  SLCAM_PIX_FORMAT_IYUV
5  时，仅data[0]~data[2]存储数据
6
7  当fmt为
8  SLCAM_PIX_FORMAT_NV12
9  时，仅data[0]~data[1]存储数据
10
11 当fmt为
12 SLCAM_PIX_FORMAT_UYVY
13 SLCAM_PIX_FORMAT_YUY2
14 SLCAM_PIX_FORMAT_RGB24
15 SLCAM_PIX_FORMAT_BGR24
16 SLCAM_PIX_FORMAT_ABGR
17 SLCAM_PIX_FORMAT_ARGB
18 SLCAM_PIX_FORMAT_RGBA
19 SLCAM_PIX_FORMAT_BGRA
20 SLCAM_PIX_FORMAT_RGB565
21 SLCAM_PIX_FORMAT_GRAY8
22 时，仅data[0]存储数据

```

pts并非标准时间戳，如需用于录像，请按下图将时间戳转换为录像的时间戳

```

// 视频时间基，固定值
const AVRational streamTimeBase = av_make_q(1, 1000000);
// 录像时间基，此处的30为录像帧率，按需修改
m_outStreamTimeBase = av_make_q(1, 30);
// 将视频流时间戳由视频时间基转换为录像时间基
// starEncodingTS是录像首帧的时间戳
// frame.pts是当前帧的时间戳
m_avFrame->pkt_dts = m_avFrame->pts = av_rescale_q_rnd(
    frame.pts - starEncodingTS, streamTimeBase,
    m_outStreamTimeBase,
    static_cast<AVRounding>(
        AV_ROUND_NEAR_INF | AV_ROUND_PASS_MINMAX));

```

```
1 typedef struct SLcamCaptureContext
2 {
3     char uniqueName[128];
4     SLcamVideoResolution resolution;
5     SLcamVideoFormat capFmt;
6     SLcamPixelFormat readFmt;
7 } SLcamCaptureContext;
```

接口 slcam_set_capture_context 输入参数

名称	描述
uniqueName	相机唯一描述符
resolution	需采集的视频分辨率
capFmt	需采集的视频流格式
readFmt	需读取的视频帧格式
fps	需采集的帧率

```
1 typedef struct SLcamVideoCaptureCapability
2 {
3     SLcamVideoResolution resolution;
4     int32_t maxFps;
5     SLcamVideoFormat videoFmt;
6 } SLcamVideoCaptureCapability;
7
8 typedef struct SLcamVideoCaptureCapabilities
9 {
10     SLcamVideoCaptureCapability videoCaps[SLCAM_MAX_CAP_SIZE];
11     int32_t capNum;
12 } SLcamVideoCaptureCapabilities;
```

SLcamVideoCaptureCapability 代表相机支持的视频采集能力

名称	描述
resolution	分辨率
maxFps	最大帧率
videoFmt	视频流格式

SLcamVideoCaptureCapabilities 视频采集能力集

接口 slcam_get_capture_capabilities 输出参数

名称	描述
videoCaps	视频采集能力集数组
capNum	视频采集能力集数组数量

```
1 typedef struct SLcamDevInfo
2 {
3     char name[128];
4     char uniqueName[128];
5     uint16_t vendorId;
6     SLcamModel model;
7     SLcamDevUsbSpeed speed;
8 } SLcamDevInfo;
9
10 typedef struct SLcamDevInfos
11 {
12     SLcamDevInfo cameras[SLCAM_MAX_DEVICES];
13     int32_t cameraNum;
14 } SLcamDevInfos;
```

SLcamDevInfo 单个已连接的相机信息

名称	描述
name	相机名称
uniqueName	相机唯一描述符
vendorId	厂商ID
model	相机型号
speed	相机连接速率

SLcamDevInfos 所有已连接相机信息

slcam_enum_devices 输出参数

名称	描述
cameras	相机信息数组
cameraNum	相机数量

```
1 typedef struct SLcamFileSaveInfo
2 {
3     SLcamImgFormat format;
4     char *savePath;
5     SLcamVideoFrame *frame;
6 } SLcamFileSaveInfo;
```

SLcamFileSaveInfo 拍照相关信息

名称	描述
format	图片保存格式
savePath	图片保存路径(包含文件名)

名称	描述
frame	需要保存为图片的视频帧

```
1 typedef struct SLcamRecordSaveInfo
2 {
3     char *savePath;
4     int width;
5     int height;
6     int32_t fps;
7 } SLcamRecordSaveInfo;
```

SLcamFileSaveInfo 录像相关信息

名称	描述
savePath	视频保存路径(包含文件名)
width	视频宽度
height	视频高度
fps	视频帧率

函数

API初始化/销毁

```
1 // 描述
2 // 初始化API库，加载部分相关资源，整个程序开始时调用一次，不建议频繁调用
3 int32_t slcam_api_init();
4
5 // 描述
6 // 卸载API库，释放内部资源，整个程序结束时调用一次，不建议频繁调用
7 int32_t slcam_api_destroy();
```

日志

```
1 // 描述
2 // 设置日志等级
3 // 参数
4 // SLcamLogLevel level 日志等级枚举类
5 int32_t slcam_log_set_level(SLcamLogLevel level);
```

```
1 // 描述
2 // 设置日志输出格式
3 int32_t slcam_log_set_pattern(const char *pattern);
```

默认日志输出格式如下

```
[2014-10-31 23:46:59.678] [my_loggername] [info] Some message
```

例子

```
1 | slcam_log_set_pattern("**** [%H:%M:%S %z] [thread %t] %v ****");
```

模式标志

模式标志的形式类似于 [strftime](#) 函数: %flag

标志	含义	例
%v	实际文本	"some user text"
%t	线程ID	"1232"
%P	进程ID	"3456"
%n	Logger的名称	"some logger name"
%l	信息的日志级别	"debug", "info", etc
%L	信息的短日志级别	"D", "I", etc
%a	工作日名称缩写	"Thu"
%A	完整的工作日名称	"Thursday"
%b	月份缩写	"Aug"
%B	月份全称	"August"
%c	日期和时间表示形式	"Thu Aug 23 15:35:46 2014"
%C	两位数字的年份	"14"
%Y	四位数字的年份	"2014"
%D 或 %x	MM/DD/YY 短日期	"08/23/14"
%m	月	"11"
%d	日	"29"
%H	24小时格式	"23"
%I	12小时格式	"11"
%M	分	"59"
%S	秒	"58"
%e	毫秒	"678"
%f	微妙	"056789"
%F	纳秒	"256789123"
%p	AM/PM	"AM"
%r	12小时制	"02:55:02 PM"

标志	含义	例
%R	等同于 %H:%M	"23:55"
%T 或 %X	等同于 %H:%M:%S	"23:55:59"
%z	ISO 8601 与时区 UTC 的偏移量 ([+/-]HH:MM)	" +02:00"
%E	自纪元以来的秒数	"1528834770"
%%	% 符号	"%"
%+	默认格式	"[2014-10-31 23:46:59.678] [mylogger] [info] Some message"
%^	起始颜色范围（只能使用一次）	"[mylogger] [info(green)] Some message"
%%\$	结束颜色范围（例如 %^[+++]%\$ %v）（只能使用一次）	[+++] Some message
%@	源文件和行	/some/dir/my_file.cpp:123
%s	源文件的基名	my_file.cpp
%g	源文件的完整路径或相对路径	/some/dir/my_file.cpp
%#	源行	123
%!	源函数	my_func
%o	自上一条消息以来经过的时间（以 毫秒为单位）	456
%i	自上一条消息以来经过的时间（以 微秒为单位）	456
%u	自上一条消息以来经过的时间（以 纳秒为单位）	11456
%O	自上一条消息以来经过的时间（以 秒为单位）	4

对齐

每个模式标志可以通过在前面加上一个宽度数字（最多 64）来对齐。

使用 - （左对齐）或 = （居中对齐）控制对齐侧

对齐	含义	例	结果
%<width><flag>	右对齐	%81	" info"
%-<width><flag>	左对齐	%-81	"info "
%=<width><flag>	居中对齐	%=81	" info "

Optionally add `!` to truncate the result if its size exceeds the specified width:

对齐	含义	例	结果
<code>%<width>!<flag></code>	右对齐或截断	<code>%3!l</code>	"inf"
<code>%-<width>!<flag></code>	左对齐或截断	<code>%-2!l</code>	"in"
<code>%=<width>!<flag></code>	居中对齐或截断	<code>%=1!l</code>	"i"

注意：若要截断函数名称，请使用“`!!`”。例如，将函数名称限制为 10 个字符。`%10!!`

```
1 // 参数
2 // int32_t level 日志等级，值对应SLcamLogLevel
3 // const char *msg 信息字符串指针
4 typedef void (*SLcamLogCallback)(int32_t level, const char *msg);
5
6 // 描述
7 // 设置日志回调函数，可用于将日志接入至用户自己的日志系统中管理
8 // 参数
9 // SLcamCallback callback 回调函数 定义如上
10 int32_t slcam_log_set_callback(SLcamLogCallback callback);
```

枚举设备

```
1 // 描述
2 // 枚举当前已连接的相机，必须在 slcam_api_init 初始化后调用
3 // 参数
4 // SLcamDevInfos *info 相机信息指针
5 int32_t slcam_enum_devices(SLcamDevInfos *info);
```

打开/关闭相机

```
1 // 描述
2 // 打开相机，必须在 slcam_api_init 初始化后调用
3 // 参数
4 // const char *uniqueName 相机唯一标识符
5 // HSLcam *cam 相机句柄指针
6 int32_t slcam_open(const char *uniqueName, HSLcam *cam);
7
8 // 描述
9 // 关闭相机，关闭后相机句柄会被释放，不可再使用
10 // 参数
11 // HSLcam cam 相机句柄
12 int32_t slcam_close(HSLcam cam);
```

获取相机视频采集能力

```
1 // 描述
2 // 获取相机视频采集能力集，比如支持什么分辨率、帧率、格式等
3 // 参数
4 // HSLcam cam 相机句柄
5 // SLcamVideoCaptureCapabilities *capabilities 相机视频采集能力集指针
6 int32_t slcam_get_capture_capabilities(HSLcam cam,
    SLcamVideoCaptureCapabilities *capabilities);
```

设置相机视频采集参数

```
1 // 描述
2 // 设置相机视频采集参数，首次打开相机后，需设置一次才可出流
3 // 参数
4 // HSLcam cam 相机句柄
5 // const SLcamCaptureContext *ctx 视频采集参数，分辨率、格式等
6 int32_t slcam_set_capture_context(HSLcam cam, const SLcamCaptureContext *ctx);
```

读取/释放视频帧

```
1 // 描述
2 // 读取视频帧，在 slcam_set_capture_context 调用一次后才能读取
3 // 参数
4 // HSLcam cam 相机句柄
5 // SLcamVideoFrame *frame 视频帧指针
6 int32_t slcam_read_frame(HSLcam cam, SLcamVideoFrame *frame);
7
8 // 描述
9 // 释放视频帧，视频帧使用完毕后需要调用此函数进行释放，否则会导致内存泄漏
10 // 参数
11 // SLcamVideoFrame *frame 视频帧指针
12 int32_t slcam_free_frame(SLcamVideoFrame *frame);
```

拍照

```
1 // 描述
2 // 保存视频帧（拍照）
3 // 参数
4 // HSLcam cam 相机句柄
5 // SLcamFileSaveInfo fileSaveInfo 拍照相关信息
6 int32_t slcam_file_save_image(HSLcam cam, SLcamFileSaveInfo fileSaveInfo);
```

录像

```
1 // 描述
2 // 开始录像
3 // 参数
4 // HSLcam cam 相机句柄
5 // SLcamRecordSaveInfo recordSaveInfo 录像相关信息
6 int32_t slcam_record_start(HSLcam cam, SLcamRecordSaveInfo recordSaveInfo);
7
```

```

8 // 描述
9 // 添加待录像帧
10 // 参数
11 // HSLcam cam 相机句柄
12 // SLcamVideoFrame *frame 传入待编码的视频帧
13 int32_t slcam_record_append_frame(HSLcam cam, const SLcamVideoFrame *frame);
14
15 // 描述
16 // 停止录像
17 // 参数
18 // HSLcam cam 相机句柄
19 int32_t slcam_record_stop(HSLcam cam);

```

聚焦

```

1 // 描述
2 // 设置当前聚焦模式，一键聚焦模式仅部分相机支持，见附表
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t mode 聚焦模式，对应SLcamFocusMode
6 int32_t slcam_set_focus_mode(HSLcam cam, int32_t mode);
7
8 // 描述
9 // 获取当前聚焦模式（不包含一键聚焦模式），所有相机均支持
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *mode 聚焦模式，对应SLcamFocusMode
13 int32_t slcam_get_focus_mode(HSLcam cam, int32_t *mode);
14
15 // 描述
16 // 获取聚焦模式（包含一键聚焦模式），仅支持具有一键聚焦功能的相机
17 // 参数
18 // HSLcam cam 相机句柄
19 // int32_t *mode 聚焦模式，对应SLcamFocusMode
20 int32_t slcam_get_focus_mode_ex(HSLcam cam, int32_t *mode);
21
22 // 描述
23 // 获取当前聚焦状态，仅部分相机支持，见附表
24 // 参数
25 // HSLcam cam 相机句柄
26 // int32_t *state 聚焦状态，0代表聚焦中，1代表聚焦完毕
27 int32_t slcam_get_focus_state(HSLcam cam, int32_t *state);
28
29 // 描述
30 // 设置手动聚焦值，M系列不可设置聚焦值
31 // 参数
32 // HSLcam cam 相机句柄
33 // int32_t value 聚焦值
34 int32_t slcam_set_focus(HSLcam cam, int32_t value);
35
36 // 描述
37 // 获取当前手动聚焦值
38 // 参数
39 // HSLcam cam 相机句柄
40 // int32_t *value 当前聚焦值

```

```

41 int32_t slcam_get_focus(HSLcam cam, int32_t *value);
42
43 // 描述
44 // 获取当前相机聚焦范围等信息，M系列不支持
45 // 参数
46 // HSLcam cam 相机句柄
47 // int32_t *minValue 最小值
48 // int32_t *maxValue 最大值
49 // int32_t *defValue 默认值
50 // int32_t *stepValue 步长
51 int32_t slcam_get_focus_range(HSLcam cam, int32_t *minValue, int32_t
    *maxValue, int32_t *defValue, int32_t *stepValue);
52
53 // 描述
54 // 增加一点聚焦值（仅对M系列有效）
55 // 参数
56 // HSLcam cam 相机句柄
57 int32_t slcam_increase_focus(HSLcam cam);
58
59 // 描述
60 // 减少一点聚焦值（仅对M系列有效）
61 // 参数
62 // HSLcam cam 相机句柄
63 int32_t slcam_decrease_focus(HSLcam cam);

```

自动聚焦范围

```

1 // 描述
2 // 设置自动聚焦范围
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t minValue 范围最小值
6 // int32_t maxValue 范围最大值
7 int32_t slcam_set_auto_focus_range(HSLcam cam, int32_t minValue, int32_t
    maxValue);
8
9 // 描述
10 // 获取自动聚焦范围
11 // 参数
12 // HSLcam cam 相机句柄
13 // int32_t *minValue 范围最小值
14 // int32_t *maxValue 范围最大值
15 int32_t slcam_get_auto_focus_range(HSLcam cam, int32_t *minValue, int32_t
    *maxValue);

```

聚焦区域



小框模式如上图，以图像左上角为坐标轴，将图像分为17x15块区域，例如蓝色区域为坐标(0, 1)，则slcam_set_focus_region_v1(cam,0,1)，绿色区域为坐标(3,1), 则slcam_set_focus_region_v1(cam,3,1)

```
1 // 描述
2 // 设置聚焦区域，支持所有聚焦相机，M系列无法调节
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t x x轴坐标
6 // int32_t y y轴坐标
7 int32_t slcam_set_focus_region_v1(HSLcam cam, int32_t x, int32_t y);
8
9 // 描述
10 // 获取聚焦区域，支持所有聚焦相机
11 // 参数
12 // HSLcam cam 相机句柄
13 // int32_t *x x轴坐标
14 // int32_t *y y轴坐标
15 int32_t slcam_get_focus_region_v1(HSLcam cam, int32_t *x, int32_t *y);
16
17 // 描述
18 // 获取聚焦区域，仅支持获取小框模式下的范围
19 // 参数
20 // HSLcam cam 相机句柄
21 // int32_t *minX x轴坐标最小值
22 // int32_t *maxX x轴坐标最大值
23 // int32_t *defX x轴坐标默认值
24 // int32_t *stepX x轴坐标步长
25 // int32_t *minY y轴坐标x轴坐标最小值
26 // int32_t *maxY y轴坐标最大值
27 // int32_t *defY y轴坐标默认值
28 // int32_t *stepY y轴坐标步长
29 int32_t slcam_get_focus_region_range(HSLcam cam, int32_t *minX, int32_t
    *minY, int32_t *maxX, int32_t *maxY, int32_t *defX, int32_t *defY, int32_t
    *stepX, int32_t *stepY);
```

以上接口所有自动聚焦相机均可支持

```
1 // 描述
2 // 设置聚焦区域，支持部分相机，见附表
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t enable 区域聚焦使能标志，0代表关闭，1代表开启
6 // int32_t mode 区域大小，0代表小框模式，1代表大框模式
7 // (小框 0 <= x <= 16, 0 <= y <= 14)
8 // (大框 0 <= x <= 4, 0 <= y <= 4)
9 // int32_t x x轴坐标
10 // int32_t y y轴坐标
11 int32_t slcam_set_focus_region_v2(HSLcam cam, int32_t enable, int32_t mode,
    int32_t x, int32_t y);
12
13 // 描述
14 // 获取聚焦区域
15 // 参数
16 // HSLcam cam 相机句柄
17 // int32_t *enable 区域聚焦使能标志，0代表关闭，1代表开启
18 // int32_t *mode 区域大小，0代表小框模式，1代表大框模式
19 // int32_t *x x轴坐标
20 // int32_t *y y轴坐标
21 int32_t slcam_get_focus_region_v2(HSLcam cam, int32_t *enable, int32_t
    *mode, int32_t *x, int32_t *y);
22
23 // 描述
24 // 设置聚焦区域，可自定义位置及大小，仅支持部分相机，见附表
25 // 参数
26 // HSLcam cam 相机句柄
27 // int32_t enable 区域聚焦使能标志，0代表关闭，1代表开启
28 // int32_t x 区域左上角x坐标
29 // int32_t y 区域左上角y坐标
30 // int32_t width 区域宽度
31 // int32_t height 区域高度
32 SLCAM_API int32_t slcam_set_focus_region_v3(HSLcam cam, int32_t enable,
    int32_t x, int32_t y, int width, int height);
33
34 // 描述
35 // 获取聚焦区域
36 // 参数
37 // HSLcam cam 相机句柄
38 // int32_t *enable 区域聚焦使能标志，0代表关闭，1代表开启
39 // int32_t *x 区域左上角x坐标
40 // int32_t *y 区域左上角y坐标
41 // int32_t *width 区域宽度
42 // int32_t *height 区域高度
43 SLCAM_API int32_t slcam_get_focus_region_v3(HSLcam cam, int32_t *enable,
    int32_t *x, int32_t *y, int *width, int *height);
```

以上接口需要新版本固件支持

获取版本号

```
1 // 描述
2 // 获取相机固件版本号
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t *version 版本号
6 // 实际版本号 = static_cast<double>(*version) / 100;
7 // 例外B201: 实际版本号 = static_cast<double>(*version & 0xFFFF) / 100;
8 int32_t slcam_get_version(HSLcam cam, int32_t *version);
```

抗频闪

```
1 // 描述
2 // 设置电力线频率
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t flag 标记, 值对应SLcamPowerLineFrequency
6 int32_t slcam_set_power_line_frequency(HSLcam cam, int32_t flag);
7
8 // 描述
9 // 获取电力线频率
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *flag 标记, 值对应SLcamPowerLineFrequency
13 int32_t slcam_get_power_line_frequency(HSLcam cam, int32_t *flag);
```

翻转

```
1 // 描述
2 // 设置镜像状态（水平翻转）
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t enable 使能标志, 0关闭, 1开启
6 int32_t slcam_set_mirror(HSLcam cam, int32_t enable);
7
8 // 描述
9 // 获取镜像状态（水平翻转）
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *enable 使能标志, 0关闭, 1开启
13 int32_t slcam_get_mirror(HSLcam cam, int32_t *enable);
```



```

1 // 描述
2 // 设置翻转状态（垂直翻转）
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t enable 使能标志，0关闭，1开启
6 int32_t slcam_set_flip(HSLcam cam, int32_t enable);
7
8 // 描述
9 // 获取翻转状态（垂直翻转）
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *enable 使能标志，0关闭，1开启
13 int32_t slcam_get_flip(HSLcam cam, int32_t *enable);

```

色度

```

1 // 描述
2 // 设置当前色度值
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t value 色度值
6 int32_t slcam_set_hue(HSLcam cam, int32_t value);
7
8 // 描述
9 // 获取当前色度值
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *value 获取的色度值
13 int32_t slcam_get_hue(HSLcam cam, int32_t *value);
14
15 // 描述
16 // 获取色度值范围等信息
17 // 参数
18 // HSLcam cam 相机句柄
19 // int32_t *minValue 最小值
20 // int32_t *maxValue 最大值
21 // int32_t *defValue 默认值
22 // int32_t *stepValue 步长
23 int32_t slcam_get_hue_range(HSLcam cam, int32_t *minValue, int32_t
    *maxValue, int32_t *defValue, int32_t *stepValue);

```

锐度

```

1 // 描述
2 // 设置当前锐度值
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t value 锐度值
6 int32_t slcam_set_sharpness(HSLcam cam, int32_t value);
7
8 // 描述
9 // 获取当前锐度值
10 // 参数
11 // HSLcam cam 相机句柄

```

```

12 // int32_t *value 获取的锐度值
13 int32_t slcam_get_sharpness(HSLcam cam, int32_t *value);
14
15 // 描述
16 // 获取锐度范围等信息
17 // 参数
18 // HSLcam cam 相机句柄
19 // int32_t *minValue 最小值
20 // int32_t *maxValue 最大值
21 // int32_t *defValue 默认值
22 // int32_t *stepValue 步长
23 int32_t slcam_get_sharpness_range(HSLcam cam, int32_t *minValue, int32_t
    *maxValue, int32_t *defValue, int32_t *stepValue);

```

饱和度

```

1 // 描述
2 // 设置当前饱和度值
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t value 饱和度值
6 int32_t slcam_set_saturation(HSLcam cam, int32_t value);
7
8 // 描述
9 // 获取当前饱和度值
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *value 获取的饱和度值
13 int32_t slcam_get_saturation(HSLcam cam, int32_t *value);
14
15 // 描述
16 // 获取饱和度值范围等信息
17 // 参数
18 // HSLcam cam 相机句柄
19 // int32_t *minValue 最小值
20 // int32_t *maxValue 最大值
21 // int32_t *defValue 默认值
22 // int32_t *stepValue 步长
23 int32_t slcam_get_saturation_range(HSLcam cam, int32_t *minValue, int32_t
    *maxValue, int32_t *defValue, int32_t *stepValue);

```

对比度

```

1 // 描述
2 // 设置当前对比度值
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t value 对比度值
6 int32_t slcam_set_contrast(HSLcam cam, int32_t value);
7
8 // 描述
9 // 获取当前对比度值
10 // 参数
11 // HSLcam cam 相机句柄

```

```

12 // int32_t *value 获取到的对比度值
13 int32_t slcam_get_contrast(HSLcam cam, int32_t *value);
14
15 // 描述
16 // 获取对比度范围等信息
17 // 参数
18 // HSLcam cam 相机句柄
19 // int32_t *minValue 最小值
20 // int32_t *maxValue 最大值
21 // int32_t *defValue 默认值
22 // int32_t *stepValue 步长
23 int32_t slcam_get_contrast_range(HSLcam cam, int32_t *minValue, int32_t
    *maxValue, int32_t *defValue, int32_t *stepValue);

```

ZOOM

```

1 // 描述
2 // 设置当前变倍值（包含光学、数字变倍）
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t value 变倍值，是光学变倍和数字变倍之和。
6 //          当变倍值大于光学变倍最大值后，才启用数字变倍
7 //          数字变倍值 = 变倍值 - 光学变倍最大值
8 int32_t slcam_set_zoom(HSLcam cam, int32_t value);
9
10 // 描述
11 // 获取当前变倍值（包含光学、数字变倍）
12 // 参数
13 // HSLcam cam 相机句柄
14 // int32_t *value 获取到的变倍值，是光学变倍和数字变倍之和。
15 //          当变倍值大于光学变倍最大值时，光学变倍值 = 光学变倍最大值，数字变倍
16 //          值 = 变倍值 - 光学变倍最大值
17 //          当变倍值小于等于光学变倍最大值时，光学变倍值 = 变倍值， 数字变倍值
18 //          = 0
19 int32_t slcam_get_zoom(HSLcam cam, int32_t *value);
20
21 // 描述
22 // 获取光学变倍范围等信息
23 // 参数
24 // HSLcam cam 相机句柄
25 // int32_t *minValue 最小值
26 // int32_t *maxValue 最大值
27 // int32_t *defValue 默认值
28 // int32_t *stepValue 步长
29 int32_t slcam_get_ozoom_range(HSLcam cam, int32_t *minValue, int32_t
    *maxValue, int32_t *defValue, int32_t *stepValue);
30
31 // 描述
32 // 获取数字变倍等信息
33 // 参数
34 // HSLcam cam 相机句柄
35 // int32_t *minValue 最小值
36 // int32_t *maxValue 最大值
37 // int32_t *defValue 默认值
38 // int32_t *stepValue 步长

```

```

37  int32_t slcam_get_dzoom_range(HSLcam cam, int32_t *minValue, int32_t
    *maxValue, int32_t *defValue, int32_t *stepvalue);
38
39  // 描述
40  // 设置相对变倍信息
41  // 参数
42  // HSLcam cam 相机句柄
43  // int8_t zoomRel 相对变倍调节方向, 0 停止, 1放大, -1缩小
44  // uint8_t digitalZoom 暂无效
45  // uint8_t speed 暂无效
46  int32_t slcam_set_zoom_relative(HSLcam cam, int8_t zoomRel, uint8_t
    digitalZoom, uint8_t speed);
47
48  // 描述
49  // 获取相对变倍信息
50  // 参数
51  // HSLcam cam 相机句柄
52  // int8_t *zoomRel 相对变倍调节方向, 0 停止, 1放大, -1缩小
53  // uint8_t *digitalZoom 暂无效
54  // uint8_t *speed 暂无效
55  int32_t slcam_get_zoom_relative(HSLcam cam, int8_t *zoomRel, uint8_t
    *digitalZoom, uint8_t *speed);
56
57  // 描述
58  // 设置当前变倍速率
59  // 参数
60  // HSLcam cam 相机句柄
61  // int32_t speed 变倍速率, 0代表低, 1代表中, 2代表高
62  int32_t slcam_set_zoom_speed(HSLcam cam, int32_t speed);
63
64  // 描述
65  // 获取当前变倍速率
66  // 参数
67  // HSLcam cam 相机句柄
68  // int32_t *speed 获取到的变倍速率, 0代表低, 1代表中, 2代表高
69  int32_t slcam_get_zoom_speed(HSLcam cam, int32_t *speed);

```

曝光模式

```

1  // 描述
2  // 获取当前曝光模式
3  // 参数
4  // HSLcam cam 相机句柄
5  // int32_t mode 曝光模式, 对应SLcamExposureMode
6  int32_t slcam_set_exposure_mode(HSLcam cam, int32_t mode);
7
8  // 描述
9  // 设置当前曝光模式
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *mode 获取的曝光模式, 对应SLcamExposureMode
13 int32_t slcam_get_exposure_mode(HSLcam cam, int32_t *mode);

```

自动曝光补偿（亮度）

```
1 // 描述
2 // 设置当前亮度（仅在自动曝光模式下生效）
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t value 亮度值
6 int32_t slcam_set_exposure_compensation(HSLcam cam, int32_t value);
7
8 // 描述
9 // 获取当前亮度
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *value 获取的亮度值
13 int32_t slcam_get_exposure_compensation(HSLcam cam, int32_t *value);
14
15 // 描述
16 // 获取当前亮度值范围等信息
17 // 参数
18 // HSLcam cam 相机句柄
19 // int32_t *minValue 最小值
20 // int32_t *maxValue 最大值
21 // int32_t *defValue 默认值
22 // int32_t *stepValue 步长
23 int32_t slcam_get_exposure_compensation_range(HSLcam cam, int32_t *minValue,
int32_t *maxValue, int32_t *defValue, int32_t *stepValue);
```

曝光时间（快门）

```
1 // 描述
2 // 设置当前快门（仅在手动曝光下生效）
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t value 快门值
6 int32_t slcam_set_exposure_time(HSLcam cam, int32_t value);
7
8 // 描述
9 // 获取当前快门
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *value 获取的快门值
13 int32_t slcam_get_exposure_time(HSLcam cam, int32_t *value);
14
15 // 描述
16 // 获取快门范围等信息
17 // 参数
18 // HSLcam cam 相机句柄
19 // int32_t *minValue 最小值
20 // int32_t *maxValue 最大值
21 // int32_t *defValue 默认值
22 // int32_t *stepValue 步长
23 int32_t slcam_get_exposure_time_range(HSLcam cam, int32_t *minValue, int32_t
*maxValue, int32_t *defValue, int32_t *stepValue);
```

长曝光时间

```
1 // 描述
2 // 设置当前长曝光时间，只有在正常的曝光时间调至最大值时才可设置，否则无效（仅在手动曝光下生效）
3 // 最大曝光时间范围为普通曝光时间和长曝光时间范围之和
4 // 参数
5 // HSLcam cam 相机句柄
6 // int32_t value 长曝光时间值
7 int32_t slcam_set_long_exposure_time(HSLcam cam, int32_t value);
8
9 // 描述
10 // 获取当前长曝光时间
11 // 参数
12 // HSLcam cam 相机句柄
13 // int32_t *value 获取的长曝光时间值
14 int32_t slcam_get_long_exposure_time(HSLcam cam, int32_t *value);
15
16 // 描述
17 // 获取长曝光时间范围等信息
18 // 参数
19 // HSLcam cam 相机句柄
20 // int32_t *minValue 最小值
21 // int32_t *maxValue 最大值
22 // int32_t *defValue 默认值
23 // int32_t *stepValue 步长
24 int32_t slcam_get_long_exposure_time_range(HSLcam cam, int32_t *minValue,
int32_t *maxValue, int32_t *defValue, int32_t *stepValue);
```

曝光增益

```
1 // 描述
2 // 设置当前曝光增益（仅在手动曝光下生效）
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t value 增益值
6 int32_t slcam_set_exposure_gain(HSLcam cam, int32_t value);
7
8 // 描述
9 // 获取当前曝光增益
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *value 获取的增益值
13 int32_t slcam_get_exposure_gain(HSLcam cam, int32_t *value);
14
15 // 描述
16 // 获取曝光增益范围等信息
17 // 参数
18 // HSLcam cam 相机句柄
19 // int32_t *minValue 最小值
20 // int32_t *maxValue 最大值
21 // int32_t *defValue 默认值
22 // int32_t *stepValue 步长
```

```
23 int32_t slcam_get_exposure_gain_range(HSLcam cam, int32_t *minValue, int32_t
    *maxValue, int32_t *defValue, int32_t *stepValue);
```

曝光增益单位

```
1 // 描述
2 // 设置当前曝光增益单位
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t unit 增益单位, 值对应SLcamExposureGainUnit
6 int32_t slcam_set_exposure_gain_unit(HSLcam cam, int32_t unit);
7
8 // 描述
9 // 获取当前曝光增益单位
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *unit 获取的增益单位, 值对应SLcamExposureGainUnit
13 int32_t slcam_get_exposure_gain_unit(HSLcam cam, int32_t *unit);
```

区域曝光

```
1 // 描述
2 // 设置区域曝光范围
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t x x轴坐标
6 // int32_t y y轴坐标
7 // int32_t w 宽
8 // int32_t h 高
9
10 int32_t slcam_set_auto_exposure_region(HSLcam cam, int32_t x, int32_t y,
    int32_t w, int32_t h);
11
12 // 描述
13 // 获取区域曝光范围
14 // 参数
15 // HSLcam cam 相机句柄
16 // int32_t x x轴坐标
17 // int32_t y y轴坐标
18 // int32_t w 宽
19 // int32_t h 高
20 int32_t slcam_get_auto_exposure_region(HSLcam cam, int32_t *x, int32_t *y,
    int32_t *w, int32_t *h);
```

注：坐标为实际画面中的位置，不是聚焦区域分块的模式。当区域刚好覆盖整个画面时，即全局曝光。

Gamma

```
1 // 描述
2 // 设置当前伽马值
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t value 伽马值
```

```

6  int32_t slcam_set_gamma(HSLcam cam, int32_t value);
7
8  // 描述
9  // 获取当前伽马值
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *value 获取的伽马值
13 int32_t slcam_get_gamma(HSLcam cam, int32_t *value);
14
15 // 描述
16 // 获取伽马范围等信息
17 // 参数
18 // HSLcam cam 相机句柄
19 // int32_t *minValue 最小值
20 // int32_t *maxValue 最大值
21 // int32_t *defValue 默认值
22 // int32_t *stepValue 步长
23 int32_t slcam_get_gamma_range(HSLcam cam, int32_t *minValue, int32_t
    *maxValue, int32_t *defValue, int32_t *stepValue);

```

白平衡模式

```

1  // 描述
2  // 设置当前白平衡模式
3  // 参数
4  // HSLcam cam 相机句柄
5  // int32_t mode 白平衡模式，对应SLcamWhiteBalanceMode
6  int32_t slcam_set_white_balance_mode(HSLcam cam, int32_t mode);
7
8  // 描述
9  // 获取当前白平衡模式
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *mode 获取的白平衡模式，对应SLcamWhiteBalanceMode
13 int32_t slcam_get_white_balance_mode(HSLcam cam, int32_t *mode);

```

色温

```

1  // 描述
2  // 设置当前白平衡色温（仅在手动白平衡下生效）
3  // 参数
4  // HSLcam cam 相机句柄
5  // int32_t value 色温值
6  int32_t slcam_set_white_balance_temperature(HSLcam cam, int32_t value);
7
8  // 描述
9  // 获取当前白平衡色温
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *value 获取的色温值
13 int32_t slcam_get_white_balance_temperature(HSLcam cam, int32_t *value);
14
15 // 描述
16 // 获取当前白平衡色温范围等信息

```



```

17 // 参数
18 // HSLcam cam 相机句柄
19 // int32_t *minValue 最小值
20 // int32_t *maxValue 最大值
21 // int32_t *defValue 默认值
22 // int32_t *stepValue 步长
23 int32_t slcam_get_white_balance_temperature_range(HSLcam cam, int32_t
    *minValue, int32_t *maxValue, int32_t *defValue, int32_t *stepValue);

```

白平衡 RGB 增益

```

1 // 描述
2 // 设置当前白平衡红色分量
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t value 红色分量值
6 int32_t slcam_set_white_balance_component_red(HSLcam cam, int32_t value);
7
8 // 描述
9 // 获取当前白平衡红色分量
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *value 获取的红色分量值
13 int32_t slcam_get_white_balance_component_red(HSLcam cam, int32_t *value);
14
15 // 描述
16 // 获取当前白平衡红色分量范围等信息
17 // 参数
18 // HSLcam cam 相机句柄
19 // int32_t *minValue 最小值
20 // int32_t *maxValue 最大值
21 // int32_t *defValue 默认值
22 // int32_t *stepValue 步长
23 int32_t slcam_get_white_balance_component_red_range(HSLcam cam, int32_t
    *minValue, int32_t *maxValue, int32_t *defValue, int32_t *stepValue);

```

```

1 // 描述
2 // 设置当前白平衡绿色分量
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t value 绿色分量值
6 int32_t slcam_set_white_balance_component_green(HSLcam cam, int32_t value);
7
8 // 描述
9 // 获取当前白平衡绿色分量
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *value 获取的绿色分量值
13 int32_t slcam_get_white_balance_component_green(HSLcam cam, int32_t *value);
14
15 // 描述
16 // 获取当前白平衡绿色分量范围等信息
17 // 参数
18 // HSLcam cam 相机句柄

```

```

19 // int32_t *minValue 最小值
20 // int32_t *maxValue 最大值
21 // int32_t *defValue 默认值
22 // int32_t *stepValue 步长
23 int32_t slcam_get_white_balance_component_green_range(HSLcam cam, int32_t
    *minValue, int32_t *maxValue, int32_t *defValue, int32_t *stepValue);

```

```

1 // 描述
2 // 设置当前白平衡蓝色分量
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t value 蓝色分量值
6 int32_t slcam_set_white_balance_component_blue(HSLcam cam, int32_t value);
7
8 // 描述
9 // 获取当前白平衡蓝色分量
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *value 获取的蓝色分量值
13 int32_t slcam_get_white_balance_component_blue(HSLcam cam, int32_t *value);
14
15 // 描述
16 // 获取当前白平衡蓝色分量范围等信息
17 // 参数
18 // HSLcam cam 相机句柄
19 // int32_t *minValue 最小值
20 // int32_t *maxValue 最大值
21 // int32_t *defValue 默认值
22 // int32_t *stepValue 步长
23 int32_t slcam_get_white_balance_component_blue_range(HSLcam cam, int32_t
    *minValue, int32_t *maxValue, int32_t *defValue, int32_t *stepValue);

```

区域白平衡

```

1 // 描述
2 // 获取区域白平衡范围
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t x x轴坐标
6 // int32_t y y轴坐标
7 // int32_t w 宽
8 // int32_t h 高
9 int32_t slcam_set_auto_whitebalance_region(HSLcam cam, int32_t x, int32_t y,
    int32_t w, int32_t h);
10
11 // 描述
12 // 获取区域白平衡范围
13 // 参数
14 // HSLcam cam 相机句柄
15 // int32_t x x轴坐标
16 // int32_t y y轴坐标
17 // int32_t w 宽
18 // int32_t h 高

```

```
19  int32_t slcam_get_auto_whitebalance_region(HSLcam cam, int32_t *x, int32_t
    *y, int32_t *w, int32_t *h);
```

注：坐标为实际画面中的位置，不是聚焦区域分块的模式。当区域刚好覆盖整个画面时，即全局白平衡。

场景切换

```
1  // 描述
2  // 设置当前场景，设置之后需调整部分参数默认值以达到最佳效果，见附表`U系列场景切换参数配置`
3  // 参数
4  // HSLcam cam 相机句柄
5  // int32_t flag 场景标记，0代表生物场景，1代表工业场景，2代表金相场景
6  int32_t slcam_set_scene(HSLcam cam, int32_t flag);
7
8  // 描述
9  // 获取当前场景
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *flag 获取的场景标记，0代表生物场景，1代表工业场景，2代表金相场景
13 int32_t slcam_get_scene(HSLcam cam, int32_t *flag);
```

消反光

```
1  // 描述
2  // 设置当前消反光值
3  // 参数
4  // HSLcam cam 相机句柄
5  // int32_t value 消反光值，范围0-127
6  int32_t slcam_set_drc(HSLcam cam, int32_t value);
7
8  // 描述
9  // 获取当前消反光值
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *value 获取的消反光值，范围0-127
13 int32_t slcam_get_drc(HSLcam cam, int32_t *value);
```

去紫边

```
1  // 描述
2  // 设置当前去紫边状态
3  // 参数
4  // HSLcam cam 相机句柄
5  // int32_t enable 去紫边使能状态，0为关闭，1为开启
6  int32_t slcam_set_cac(HSLcam cam, int32_t enable);
7
8  // 描述
9  // 获取当前去紫边状态
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *enable 获取的去紫边使能状态，0为关闭，1为开启
13 int32_t slcam_get_cac(HSLcam cam, int32_t *enable);
```

对比度增强

```
1 // 描述
2 // 设置当前对比度增强状态
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t enable 对比度增强使能状态，0为关闭，1为开启
6 int32_t slcam_set_ldci(HSLcam cam, int32_t enable);
7
8 // 描述
9 // 获取当前对比度增强状态
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *enable 获取的对比度增强使能状态，0为关闭，1为开启
13 int32_t slcam_get_ldci(HSLcam cam, int32_t *enable);
```

弱纹理增强

```
1 // 描述
2 // 设置当前弱纹理增强状态
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t enable 弱纹理增强使能状态，0为关闭，1为开启
6 int32_t slcam_set_bayer_shp(HSLcam cam, int32_t enable);
7
8 // 描述
9 // 获取当前弱纹理增强状态
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *enable 获取的弱纹理增强使能状态，0为关闭，1为开启
13 int32_t slcam_get_bayer_shp(HSLcam cam, int32_t *enable);
```

灯

```
1 // 描述
2 // 设置当前灯的亮度
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t ledIndex 灯的序号
6 // int32_t partIndex 对应灯的分区号
7 // int32_t level 亮度
8 int32_t slcam_set_led(HSLcam cam, int32_t ledIndex, int32_t partIndex,
9 int32_t level);
10
11 // 描述
12 // 获取当前灯的亮度
13 // 参数
14 // HSLcam cam 相机句柄
15 // int32_t ledIndex 灯的序号
16 // int32_t partIndex 对应灯的分区号
17 // int32_t *level 获取的亮度
18 int32_t slcam_get_led(HSLcam cam, int32_t ledIndex, int32_t partIndex,
19 int32_t *level);
```

```

19 // 描述
20 // 获取当前灯的信息
21 // 参数
22 // HSLcam cam 相机句柄
23 // int32_t *ledNumber 灯的数量
24 // int32_t *partNumber 每个灯的分区数量
25 // int32_t *minLevel 最低亮度
26 // int32_t *maxLevel 最高亮度
27 // int32_t *defLevel 默认亮度
28 // int32_t *stepLevel 亮度调节步长
29 int32_t slcam_get_led_info(HSLcam cam, int32_t *ledNumber, int32_t
    *partNumber, int32_t *minLevel, int32_t *maxLevel, int32_t *defLevel,
    int32_t *stepLevel);
30
31 // 描述
32 // 设置A3xx相机所有灯光亮度
33 // 参数
34 // HSLcam cam 相机句柄
35 // int32_t up 上分区灯光亮度
36 // int32_t left 左分区灯光亮度
37 // int32_t right 右分区灯光亮度
38 // int32_t down 下分区灯光亮度
39 // 注: 亦可使用slcam_set_led来设置灯光, 灯序号为0, 分区号对应 0-up, 1-left, 2-right,
    3-down
40 int32_t slcam_set_a3xx_all_led(HSLcam cam, int32_t up, int32_t left, int32_t
    right, int32_t down);
41
42 // 描述
43 // 获取A3xx相机所有灯光亮度
44 // 参数
45 // HSLcam cam 相机句柄
46 // int32_t *up 上分区灯光亮度
47 // int32_t *left 左分区灯光亮度
48 // int32_t *right 右分区灯光亮度
49 // int32_t *down 下分区灯光亮度
50 // 注: 亦可使用slcam_get_led来获取灯光亮度, 灯序号为0, 分区号对应 0-up, 1-left, 2-
    right, 3-down
51 int32_t slcam_get_a3xx_all_led(HSLcam cam, int32_t *up, int32_t *left,
    int32_t *right, int32_t *down);
52
53 // 描述
54 // 设置A3xx相机灯光使能状态及亮度 (A3xx 1.2.0版本后可用)
55 // 参数
56 // HSLcam cam 相机句柄
57 // int32_t partIndex 分区号
58 // int32_t enable 使能状态, 0-关闭, 1-开启
59 // int32_t level 亮度
60 // 注: 分区号对应 0-up, 1-left, 2-right, 3-down
61 int32_t slcam_set_a3xx_led(HSLcam cam, int32_t partIndex, int32_t enable,
    int32_t level);
62
63 // 描述
64 // 获取A3xx相机灯光使能状态及亮度 (A3xx 1.2.0版本后可用)
65 // 参数
66 // HSLcam cam 相机句柄
67 // int32_t partIndex 分区号

```

```

68 // int32_t *enable      使能状态，0-关闭，1-开启
69 // int32_t *level      亮度
70 // 注：分区号对应 0-up, 1-left, 2-right, 3-down
71 int32_t slcam_get_a3xx_led(HSLcam cam, int32_t partIndex, int32_t *enable,
    int32_t *level);

```

除雾

```

1 // 描述
2 // 设置当前除雾状态
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t enable 除雾使能状态，0为关闭，1为开启
6 int32_t slcam_set_defog(HSLcam cam, int32_t enable);
7
8 // 描述
9 // 获取当前除雾状态
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *enable 获取的除雾使能状态，0为关闭，1为开启
13 int32_t slcam_get_defog(HSLcam cam, int32_t *enable);

```

用户ID

```

1 // 描述
2 // 设置当前用户ID
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t userId 用户ID，范围(0x0-0x3FFFFFFF)
6 int32_t slcam_set_user_id(HSLcam cam, int32_t userId);
7
8 // 描述
9 // 获取当前用户ID
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t userId 用户ID，范围(0x0-0x3FFFFFFF)
13 int32_t slcam_get_user_id(HSLcam cam, int32_t *userId);

```

Windows系统版本

```

1 // 描述
2 // 通知相机当前windows版本（仅在windows系统下可用）
3 // 当相机设置为win7版本时，将会重启切换至USB2.0，以兼容此系统
4 // 其他系统均为USB3.0
5 // 参数
6 // HSLcam cam 相机句柄
7 // int32_t windowsVersion windows系统版本，7对应windows7，10对应windows10，11对应
    windows11
8 int32_t slcam_set_windows_version(HSLcam cam, int32_t windowsVersion);

```

型号后缀

```
1 // 描述
2 // 获取相机型号后缀
3 // 参数
4 // HSLcam cam 相机句柄
5 // char *suffix 型号后缀，例如U205A的后缀为A，U205无后缀，为空字符
6 // 仅有单个字符，大写
7 int32_t slcam_get_model_suffix(HSLcam cam, char *suffix);
```

升级

```
1 // 描述
2 // 相机固件升级（仅U408可用）
3 // 参数
4 // HSLcam cam 相机句柄
5 // const char *filePath 升级文件路径
6 // SLcamProgressCallback callback 升级进度回调函数
7 // void *ctx 用户自定义ctx
8 int32_t slcam_upgrade(HSLcam cam, const char *filePath, SLcamProgressCallback callback, void *ctx);
```

设备名称

```
1 // 描述
2 // 设置设备名称（仅U408可用）
3 // 参数
4 // HSLcam cam 相机句柄
5 // const char *name 设备名称，字符数需小于32
6 int32_t slcam_set_device_name(HSLcam cam, const char *name);
```

唯一ID

```
1 // 描述
2 // 获取相机唯一ID
3 // 参数
4 // HSLcam cam 相机句柄
5 // uint8_t *uniqueId 唯一ID，最大长度60字节
6 // int32_t len 传入unique_id的长度
7 // int32_t *readLen 读取到的唯一ID长度
8 int32_t slcam_get_unique_id(HSLcam cam, uint8_t *uniqueId, int32_t len, int32_t *readLen);
```

伽马模式

```
1 // 描述
2 // 设置相机的伽马模式
3 // 参数
4 // HSLcam cam 相机句柄
5 // int32_t mode 伽马模式，对应SLcamGammaMode
6 int32_t slcam_set_gamma_mode(HSLcam cam, int32_t mode);
7
8 // 描述
9 // 获取当前相机的伽马模式
10 // 参数
11 // HSLcam cam 相机句柄
12 // int32_t *mode 返回当前的伽马模式，对应SLcamGammaMode
13 int32_t slcam_get_gamma_mode(HSLcam cam, int32_t *mode);
```

伽马贝塞尔曲线

```
1 // 描述
2 // 设置相机的伽马贝塞尔曲线，仅处于曲线模式下有效
3 // 参数
4 // HSLcam cam 相机句柄
5 // const SLPoint *points 存储伽马贝塞尔曲线点的数组（由用户提供）
6 // int32_t size 曲线点的数量
7 int32_t slcam_set_gamma_bezier_curve(HSLcam cam, const SLPoint *points,
8   int32_t size);
9
10 // 描述
11 // 获取相机的伽马贝塞尔曲线
12 // 参数
13 // HSLcam cam 相机句柄
14 // SLPoint *points 存储贝塞尔曲线点的数组（由用户分配），返回贝塞尔曲线的控制点
15 // int32_t size 传入数组的最大容量
16 // int32_t *readSize 返回实际获取的贝塞尔曲线点的数量
17 int32_t slcam_get_gamma_bezier_curve(HSLcam cam, SLPoint *points, int32_t
18   size, int32_t *readSize);
```

ROI(感兴趣区域)

```
1 // 设置当前ROI区域，仅支持在 set_capture_context 成功后调用
2 // 可设置的ROI区域，以当前采集的视频分辨率为准（例如在视频分辨率1920x1080下，ROI只能设置
   在此范围内）
3 // 相机重新开启，ROI会保持原样，但是视频分辨率变化后，ROI将默认关闭
4 // 参数
5 // HSLcam cam 相机句柄
6 // int32_t enable ROI裁剪是否启用
7 // int32_t x ROI区域起点x轴坐标，必须为偶数
8 // int32_t y ROI区域起点y轴坐标，必须为偶数
9 // int32_t width ROI区域宽度，必须为8的倍数
10 // int32_t height ROI区域高度，必须为8的倍数
11 SLCAM_API int32_t slcam_set_roi_region(HSLcam cam, int32_t enable, int32_t
   x, int32_t y, int32_t width, int32_t height);
```



```

12
13 // 获取当前ROI区域，仅支持在 set_capture_context 成功后调用
14 // 参数
15 // HSLcam cam      相机句柄
16 // int32_t *enable  获取到的ROI裁剪是否启用
17 // int32_t *x       获取到的ROI区域起点x轴坐标
18 // int32_t *y       获取到的ROI区域起点y轴坐标
19 // int32_t *width   获取到的ROI区域宽度
20 // int32_t *height  获取到的ROI区域高度
21 SLCAM_API int32_t slcam_get_roi_region(HSLcam cam, int32_t *enable, int32_t
*x, int32_t *y, int32_t *width, int32_t *height);

```

锐度增强

```

1 // 设置锐度增强功能
2 // 锐度增强可以提高图像的清晰度和边缘定义
3 // 参数
4 // HSLcam cam      相机句柄
5 // int32_t enable  锐度增强是否启用，0 禁用，1 启用
6 SLCAM_API int32_t slcam_set_sharpness_enhancement(HSLcam cam, int32_t
enable);
7
8 // 获取当前锐度增强功能状态
9 // 参数
10 // HSLcam cam      相机句柄
11 // int32_t *enable  获取到的锐度增强是否启用，0 禁用，1 启用
12 SLCAM_API int32_t slcam_get_sharpness_enhancement(HSLcam cam, int32_t
*enable);

```

高性能模式

```

1 // 设置高性能模式
2 // 高性能模式下，相机输出帧率会提升，但部分电脑存在兼容问题，需关闭
3 // 切换模式后，相机将会重启
4 // 参数
5 // HSLcam cam      相机句柄
6 // int32_t enable  高性能模式状态，0 关闭，1 开启
7 SLCAM_API int32_t slcam_set_high_performance_mode(HSLcam cam, int32_t
enable);
8
9 // 获取当前高性能模式状态
10 // 参数
11 // HSLcam cam      相机句柄
12 // int32_t *enable  返回当前高性能模式状态，0 关闭，1 开启
13 SLCAM_API int32_t slcam_get_high_performance_mode(HSLcam cam, int32_t
*enable);

```

附录

U系列场景切换参数配置

型号/模式	生物模式	工业模式	金相模式
U106	使用SDK场景切换至生物场景，同时PC端下发如下设置： 亮度：150 饱和度：128 锐度：0 色度：50 对比度：40 伽马：40 消反光：0	使用SDK场景切换至工业场景，同时PC端下发如下设置： 亮度：50 饱和度：128 锐度：100 色度：50 对比度：55 伽马：110 消反光：0	使用SDK场景切换至金相场景，同时PC端下发如下设置： 亮度：50 饱和度：140 锐度：170 色度：50 对比度：50 伽马：100 消反光：0
U108T	使用SDK场景切换至生物场景，同时PC端下发如下设置： 亮度：150 饱和度：128 锐度：0 色度：50 对比度：40 伽马：40 消反光：0	使用SDK场景切换至工业场景，同时PC端下发如下设置： 亮度：50 饱和度：128 锐度：100 色度：50 对比度：55 伽马：110 消反光：0	使用SDK场景切换至金相场景，同时PC端下发如下设置： 亮度：50 饱和度：150 锐度：180 色度：50 对比度：50 伽马：100 消反光：0
U112/U112T	使用SDK场景切换至生物场景，同时PC端下发如下设置： 亮度：150 饱和度：128 锐度：0 色度：50 对比度：40 伽马：40 消反光：0	使用SDK场景切换至工业场景，同时PC端下发如下设置： 亮度：50 饱和度：128 锐度：100 色度：50 对比度：55 伽马：110 消反光：0	使用SDK场景切换至金相场景，同时PC端下发如下设置： 亮度：50 饱和度：145 锐度：200 色度：50 对比度：50 伽马：100 消反光：0
U120/U120T	使用SDK场景切换至生物场景，同时PC端下发如下设置： 亮度：150 饱和度：128 锐度：0 色度：50 对比度：40 伽马：40 消反光：0	使用SDK场景切换至工业场景，同时PC端下发如下设置： 亮度：50 饱和度：128 锐度：100 色度：50 对比度：55 伽马：110 消反光：0	使用SDK场景切换至金相场景，同时PC端下发如下设置： 亮度：50 饱和度：140 锐度：180 色度：50 对比度：50 伽马：100 消反光：0
U306	使用SDK场景切换至生物场景，同时PC端下发如下设置： 亮度：150 饱和度：128 锐度：0 色度：50 对比度：40 伽马：40 消反光：0	使用SDK场景切换至工业场景，同时PC端下发如下设置： 亮度：50 饱和度：128 锐度：100 色度：50 对比度：55 伽马：110 消反光：0	使用SDK场景切换至金相场景，同时PC端下发如下设置： 亮度：50 饱和度：140 锐度：200 色度：50 对比度：50 伽马：100 消反光：0
U405	使用SDK场景切换至生物场景，同时PC端下发如下设置： 亮度：150 饱和度：128 锐度：0 色度：50 对比度：40 伽马：40 消反光：0	使用SDK场景切换至工业场景，同时PC端下发如下设置： 亮度：50 饱和度：128 锐度：100 色度：50 对比度：55 伽马：110 消反光：0	使用SDK场景切换至金相场景，同时PC端下发如下设置： 亮度：50 饱和度：145 锐度：220 色度：50 对比度：50 伽马：100 消反光：0
U406	使用SDK场景切换至生物场景，同时PC端下发如下设置： 亮度：150 饱和度：128 锐度：0 色度：50 对比度：40 伽马：40 消反光：0	使用SDK场景切换至工业场景，同时PC端下发如下设置： 亮度：50 饱和度：128 锐度：100 色度：50 对比度：55 伽马：110 消反光：0	使用SDK场景切换至金相场景，同时PC端下发如下设置： 亮度：50 饱和度：140 锐度：200 色度：50 对比度：50 伽马：100 消反光：0

功能列表

0408	60FPS#8384#2160 (RJPCB) 30FPS#1520#1080 (RW12) 10FPS#3072#1728 (RW12) 60FPS#8384#2160 (RW12)	×	×	×	×	×	×	✓	0°100	0.25ms~40ms	0~36	✓	21000° 7500	0°4096	✓	0°100	0°100	0°100	×	0°100	×	×	×	✓	×	0°100	×	✓	×	×	×	×	×	×
A311W (USB)	60FPS#1280#720 (RJPCB) 60FPS#1520#1080 (RJPCB) 60FPS#1280#720 (RW12) 60FPS#1520#1080 (RW12)	✓	✓	0~400	✓	✓	支持自定义	✓	0°250	0°16.50ms (66/4)	0°30	✓	×	1°4096	✓	0°100	0°255	0°100	×	0°15	✓	0.1°4 5	✓	✓	×	1°170	✓	✓	✓	×	×	×	×	×
A312W (USB)	60FPS#1280#720 (RJPCB) 60FPS#1520#1080 (RW12) 60FPS#1280#720 (RW12)	✓	✓	0~400	✓	✓	支持自定义	✓	0°250	0°16.50ms (66/4)	0°30	✓	×	1°4096	✓	0°100	0°255	0°100	×	0°15	✓	0.1°4 8	✓	✓	×	1°170	0°255	✓	✓	×	×	×	×	×
A321 (USB)	60FPS#1280#720 (RJPCB) 60FPS#1520#1080 (RJPCB) 60FPS#1280#720 (RW12) 60FPS#1520#1080 (RW12)	×	×	×	×	×	×	✓	0°250	0°16.50ms (66/4)	0°30	✓	×	1°4096	✓	0°100	0°255	0°100	×	0°15	✓	0.1°4 5	✓	✓	×	1°170	✓	✓	✓	×	×	×	×	×
A322 (USB)	60FPS#1280#720 (RW12) 60FPS#1520#1080 (RW12)	×	×	×	×	×	×	✓	0°250	0°16.50ms (66/4)	0°30	✓	×	1°4096	✓	0°100	0°255	0°100	×	0°15	✓	0.1°4 5	✓	✓	×	1°170	✓	✓	✓	×	×	×	×	×
A313 (USB)	30FPS#8384#2160 (a.jpg) 60FPS#1520#1080 (a.jpg) 60FPS#1080#720 (a.jpg) 15FPS#8384#2160 (rw12) 60FPS#1520#1080 (rw12) 60FPS#1080#720 (rw12)	✓	✓	0~400	✓	✓	支持自定义	✓	0°250	0°16.50ms (66/4)	0°30	✓	×	1°4096	✓	0°100	0°255	0°100	×	0°15	✓	0.1°4 8	✓	✓	×	1°170	0°255	✓	✓	×	×	×	×	×
A314 (USB)	30FPS#8396#3000 (a.jpg) 30FPS#8384#2160 (a.jpg) 30FPS#1520#1080 (a.jpg) 10FPS#8396#3000 (rw12) 15FPS#8384#2160 (rw12) 30FPS#1520#1080 (rw12)	✓	✓	0~400	✓	✓	支持自定义	✓	0°250	0°33ms	0°30	✓	×	1°4096	✓	0°100	0°255	0°100	×	0°15	✓	0.1°4 8	✓	✓	×	1°170	0°255	✓	✓	×	×	×	×	×