SV1-25 具身立体双目相机

使用手册 V1.0



Unitree

本产品为民用机器人产品，请各位用户不要危险性改造和使用机器人。

请访问宇树科技官网了解更多产品相关条款与政策，请遵守各地区法律法规。

SV1-25 具身立体双目相机

简介

SV1-25具身立体双目相机搭载1280×800像素的高质量传感器，采用1/4英寸光学格式，提供卓越的图像捕捉能力。采用全局快门技术，确保动态画面清晰无畸变。拥有超广220°视角和定焦镜头，适合宽视野应用。凭借25mm基线距离，提升立体视觉精度。通过USB2.0接口，支持MJPEG格式输出，提供1856x800@30FPS高清视频，也可选择928x400分辨率下的60FPS高速拍摄。采用低功耗设计，工作温度覆盖-10℃至60℃，适应多种环境，是高性能双目视觉应用的优选。





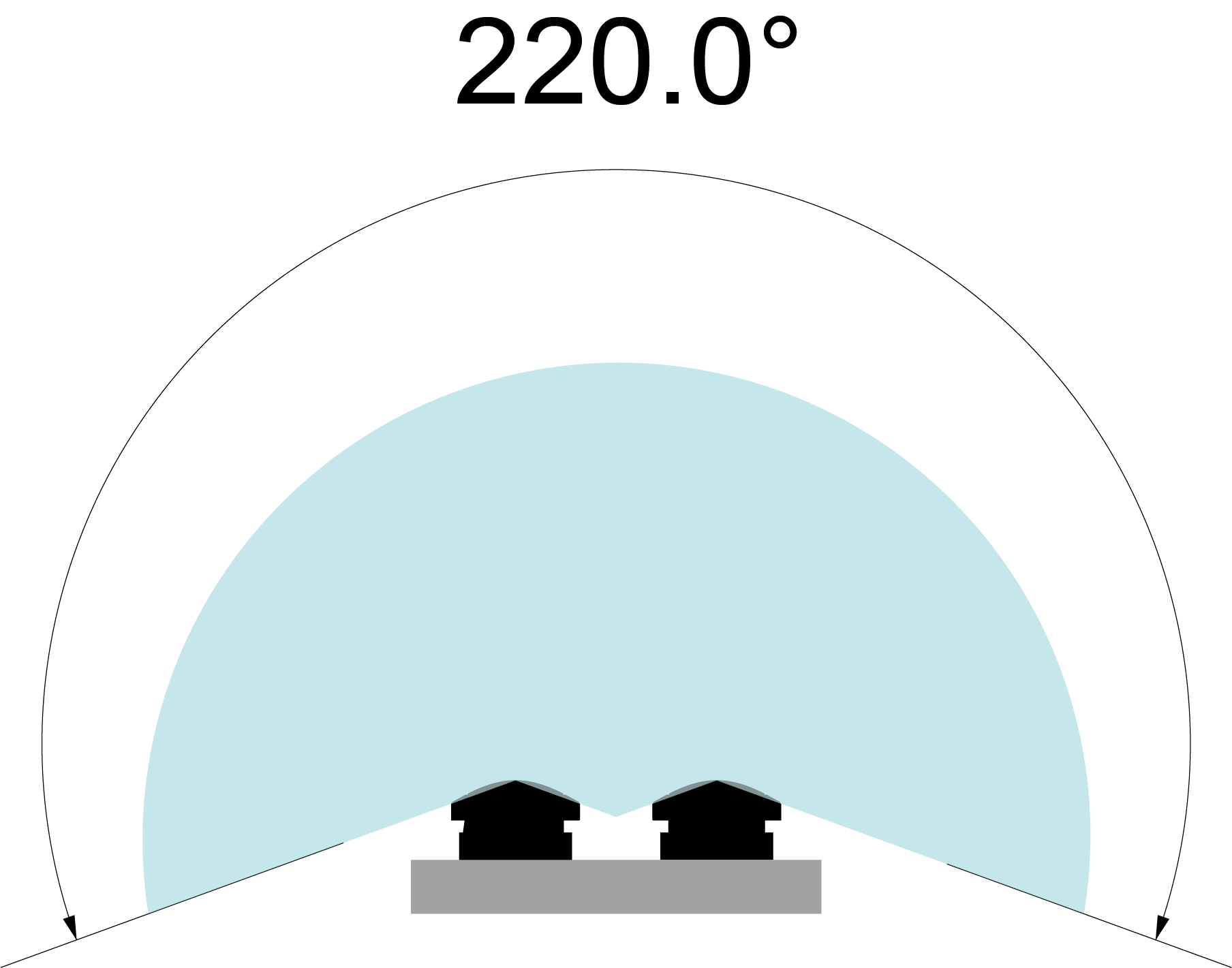
物品清单

|  |  |
| --- | --- |
|  |  |
| 双目相机 | USBA-4pin GH1.25线缆1m |

技术规格

|  |  |
| --- | --- |
| 参数 | 规格 |
| 芯片参数 | 1280(H) x 800(V) = 1.0 Mp |
| 光学参数 | 1/4-inch(4.5 mm) |
| 快门类型 | 彩色全局快门 |
| 视角 | V = 220°(Y = 1.35mm)  H = 220°(Y = 1.80mm)  D = 220°(Y = 2.29mm) |
| 镜头类型 | 定焦 |
| 基线 | 25mm(左镜头中线到右镜头中线距离) |
| 接口类型 | USB2.0 |
| 输出图像格式 | MJPEG |
| 支持的分辨率和帧率 | 1856x800@30FPS  928x400@30FPS /60FPS |
| 供电要求 | 5V /200mA |
| 滤光片截至频率 | T = 50% at 650±10nm |
| 工作温度 | -10℃ ~ 60℃ |

**视场角示意:**



接口说明



5V

D+

D-

GND

调试连接

您可以使用自己的PC USB口连接SV1-25具身立体双目相机的GH1.24-4pin接口，来建立User PC和双目相机之间的通讯。



双目相机

User PC

USBA-4pin GH1.25线缆

API接口说明

SDK简介

Unitree为SV1-25具身立体双目相机提供两个版本的SDK，版本1是依赖于OpenCV，OpenGL，GLUT，X11等外部库函数，可以实现拉流，鱼眼矫正等功能，用于使用相机前，必须安装相关依赖，否则无法正常使用。版本2是去除外部依赖的轻量级相机SDK，满足有开发能力的linux用户灵活自主的相机开发需求，同时也提供一些加速用户开发的接口。

|  |  |
| --- | --- |
| P1160C1T39#yIS1 | 注意：因为相机硬件特性，在进行相机像素配置的时候双目情况下的分辨率支持1856x800、928x400两种，也推荐用户配置成这两种像素格式。 |

版本1

依赖于OpenCV，OpenGL，GLUT，X11等外部库函数，可以实现拉流，鱼眼矫正等功能，用于使用相机前，必须安装相关依赖，否则无法正常使用。



1. StereoCamera(void)

【描述】

初始化sensor，包括设置输入图像分辨率、帧率。

【定义】

StereoCamera(void)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| void | 无 | 无参数 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 无 | 无 |

1. StereoCamera(int deviceNode)

【描述】

初始化sensor，包括设置输入图像分辨率、帧率。

【定义】

StereoCamera(void)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| deviceNode | int | sensor的标号，对应于/dev/videoX |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 无 | 无 |

1. StereoCamera(std::string fileName)

【描述】

初始化sensor，包括设置输入图像分辨率、帧率。

【定义】

StereoCamera(std::string fileName)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| fileName | string | fileName对应的配置文件初始化相机 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 无 |  |

1. isOpened(void)

【描述】

获取相机的运行状态，必须在相机初始化后调用。

【定义】

bool isOpened(void)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| void | 无 | 无 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| true | 相机正常运行初始化 |
| false | 相机状态异常 |

1. setLogLevel(int level)

【描述】

设置日志等级

【定义】

bool setLogLevel(int level)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| level | int | 设置的日志等级 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| true | 设置成功 |
| false | 设置失败 |

1. getLogLevel(void)

【描述】

获取日志等级

【定义】

int getLogLevel(int level)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| void | 无 | 无 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| int | 日志等级 |

1. getRawFrame(cv::Mat &frame, std::chrono::microseconds &timeStamp)

【描述】

相机取流函数，同时保存左目右右目帧数据

【定义】

bool StereoCamera::getRawFrame(cv::Mat &frame, std::chrono::microseconds &timeStamp)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| frame | cv::Mat | 保存帧数据 |
| timeStamp | std::chrono::microseconds | 保存帧对应时间戳 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| true | 传输成功 |
| false | 传输失败 |

1. getStereoFrame(cv::Mat&left,cv::Mat&right,std::chrono::microseconds &timeStamp)

【描述】

相机取流函数，左目数据与右目数据分开保存

【定义】

bool StereoCamera::getStereoFrame(cv::Mat &left, cv::Mat &right, std::chrono::microseconds &timeStamp)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| left | cv::Mat | 保存左帧数据 |
| right | cv::Mat | 保存右帧数据 |
| timeStamp | std::chrono::microseconds | 保存帧对应时间戳 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| true | 传输成功 |
| false | 传输失败 |

1. getRectStereoFrame(cv::Mat &left, cv::Mat &right)

【描述】

相机取流函数，并通过自定义矩阵变换和线性差值把双目帧数据分别保存左目数据与右目数据

【定义】

bool StereoCamera::getStereoFrame(cv::Mat &left, cv::Mat &right, std::chrono::microseconds &timeStamp)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| left | cv::Mat | 保存左帧数据 |
| right | cv::Mat | 保存右帧数据 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| true | 传输成功 |
| false | 传输失败 |

1. setRawFrameRate(int frameRate)

【描述】

设置相机的帧率，如果输出的分辨率是1856x800，帧率是30，如果输出的分辨率是928x400，输出的帧率是30到60。

【定义】

bool StereoCamera::setRawFrameRate(int frameRate)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| frameRate | int | 设置相机的帧率 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| true | 设置成功 |
| false | 设置失败 |

1. getRawFrameRate(int frameRate)

【描述】

获取设置的帧率数值。

【定义】

float StereoCamera::getRawFrameRate(void)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| void | 无 | 无 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| float | 返回设置的帧率 |
| false | 设置失败 |

1. setRawFrameSize(cv::Size frameSize)

【描述】

设置相机的分辨率，分辨率有两种，一种是1856x800，一种是928x400

【定义】

bool StereoCamera::setRawFrameSize(cv::Size frameSize)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| frameSize | cv::Size | 设置图像的分辨率 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| true | 设置成功 |
| false | 设置失败 |

1. getRawFrameSize(void)

【描述】

获取数据帧的大小

【定义】

cv::Size StereoCamera::getRawFrameSize(void)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| void | 无 | 无 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| cv::Size | 数据帧的分辨率 |

1. setRectFrameSize(cv::Size frameSize)

【描述】

设置矫正后的图像分辨率

【定义】

bool StereoCamera::setRectFrameSize(cv::Size frameSize)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| frameSize | cv::Size | 矫正后的分辨率 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| true | 设置成功 |
| false | 设置失败 |

1. startCapture(bool udpFlag, bool shmFlag)

【描述】

相机启动数据流传输，并配置是否需要进行UDP传输，是否使用共享内存

【定义】

bool StereoCamera::startCapture(bool udpFlag, bool shmFlag)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| udpFlag | bool | 是否udp传输 |
| shmFlag | bool | 是否使用共享内存 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| true | 设置成功 |
| false | 设置失败 |

1. stopCapture(void)

【描述】

相机停止数据传输，该接口与startCapture配合使用

【定义】

bool StereoCamera::stopCapture(void)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| Void | 无 | 无 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| true | 设置成功 |
| false | 设置失败 |

版本2

去除外部依赖的轻量级相机SDK，满足有开发能力的linux用户灵活自主的相机开发需求，同时也提供一些加速用户开发的接口。

轻量级SDK提供了两组接口：自主开发模式接口和快速开发模式接口。自主开发模式接口，在相机使用前，先调用UnitreeOpenCamKey函数，解锁相机使用模式，然后用户可以定制化开发自己的功能，实现自己的相机开发逻辑。如果用户采用快速开发模式接口，无需调用UnitreeOpenCamKey函数，下文详细介绍了相关的API接口。

**自主开发模式**

1. UnitreeOpenCamKey(void)

【描述】

打开相机的camel模式，这样相机才可以正常使用，该接口与非自主开发模式是互斥使用的，使用该接口表示相机进入自主开发模式。

【定义】

int UnitreeOpenCamKey(void)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| 无 | 无 | 无参数 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 表示接口调用成功 |
| 其他 | 接口调用失败 |

1. UnitreeCloseCamKey(void)

【描述】

关闭相机的自主开发模式。

【定义】

int UnitreeCloseCamKey(void)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| void | 无 | 无参数 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 表示接口调用成功 |
| 其他 | 接口调用失败 |

**快速开发模式**

1. UnitreeInitSdkEnv(void)

【描述】

去初始化相机的启动环境

【定义】

int UnitreeInitSdkEnv(void)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| 无 | void | 无参数 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 接口调用成功 |
| 其他 | 接口调用失败 |

1. UnitreeDeInitSdkEnv(void)

【描述】

去初始化相机的启动环境

【定义】

int UnitreeDeInitSdkEnv(void)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| 无 | void | 无参数 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 接口调用成功 |
| 其他 | 接口调用失败 |

1. UnitreeInitCamera(int CamId)

【描述】

初始化相机配置

【定义】

int InitCamera(int CamId)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| CamId | int | sensor的标号，对应于/dev/videoX |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 接口调用成功 |
| 其他 | 接口调用失败 |

1. UnitreeDeInitCamera(int CamId)

【描述】

去初始化相机配置

【定义】

int InitCamera(int CamId)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| CamId | int | sensor的标号，对应于/dev/videoX |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 接口调用成功 |
| 其他 | 接口调用失败 |

1. UnitreeSetCamParam(int CamId, ut\_cam\_ctx \*pCamCtx)

【描述】

初始化相机配置，配置相机的像素格式，相机图像的宽高等。

【定义】

int UnitreeSetCamParam(int CamId, ut\_cam\_ctx \*pCamCtx)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| CamId | int | sensor的标号，对应于/dev/videoX |
| pCamCtx | ut\_cam\_ctx | 相机配置的上下文结构 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 接口调用成功 |
| 其他 | 接口调用失败 |

1. UnitreeGetCamParam(int CamId, ut\_cam\_param \*pCamParam)

【描述】

获取相机的配置参数，该接口必须在UnitreeSetCamParam调用之后，才可以得到正确的数值。

【定义】

int UnitreeGetCamParam(int CamId, ut\_cam\_param \*pCamParam)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| CamId | int | sensor的标号，对应于/dev/videoX |
| pCamParam | ut\_cam\_param | 相机配置的上下文 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 接口调用成功 |
| 其他 | 接口调用失败 |

1. UnitreeGetCamerbuffer(int CamId, unsigned int buf\_num)

【描述】

相机运转起来，需要的buf个数，一般配置为4

【定义】

int UnitreeGetCamerbuffer(int CamId, unsigned int buf\_num)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| CamId | int | sensor的标号，对应于/dev/videoX |
| buf\_num | unsigned int | 相机取帧时用到的buf个数 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 接口调用成功 |
| 其他 | 接口调用失败 |

1. UnitreeReleaseCamerBuffer(int CamId)

【描述】

在最后需要释放相机的buf，防止内存泄漏

【定义】

int UnitreeReleaseCamerBuffer(int CamId)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| CamId | int | sensor的标号，对应于/dev/videoX |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 接口调用成功 |
| 其他 | 接口调用失败 |

1. UnitreeStreamStart(int CamId)

【描述】

启动相机

【定义】

int UnitreeStreamStart(int CamId)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| CamId | int | sensor的标号，对应于/dev/videoX |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 接口调用成功 |
| 其他 | 接口调用失败 |

1. UnitreeStreamStop(int CamId)

【描述】

停止相机

【定义】

int UnitreeStreamStop(int CamId)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| CamId | int | sensor的标号，对应于/dev/videoX |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 接口调用成功 |
| 其他 | 接口调用失败 |

1. UnitreeGetAndReleaseframe(int CamId)

【描述】

每调用一次该接口，保存一帧.mjpg的数据，接口内自动申请和释放帧存数据。

【定义】

int UnitreeGetReleaseframe(int CamId)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| CamId | int | sensor的标号，对应于/dev/videoX |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 接口调用成功 |
| 其他 | 接口调用失败 |

1. UnitreeGetOneFrame(int CamId, void \*\*Framebuf, unsigned int \*index, unsigned int \*Bytesused)

【描述】

每调用一次该接口，帧数据会保存在一个一维指针里面，通过这个会二维指针Framebuf记录保存帧

数据的指针的指针的位置，在相机内部会给该帧进行编号，编号记录在index里面，Bytesused是帧的大小。Framebuf，index，Bytesused都是输出接口

【定义】

int UnitreeGetOneFrame(int CamId, void \*\*Framebuf, unsigned int \*index, unsigned int \*Bytesused)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| CamId | int | sensor的标号，对应于/dev/videoX |
| Framebuf | void\*\* | 二维指针，指向帧缓冲区指针的指针 |
| index | int \* | 相机采集图片的帧编号 |
| Bytesused | unsigned int \* | 记录了帧存的大小 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 接口调用成功 |
| 其他 | 接口调用失败 |

1. UnitreeReleaseOneFrame(int CamId, unsigned int index, unsigned int Bytesused)

【描述】

该接口与UnitreeGetOneFrame是成对出现的，是一个帧缓冲器释放接口。

【定义】

int UnitreeReleaseOneFrame(int CamId, unsigned int index, unsigned int Bytesused)

【参数】

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| CamId | int | sensor的标号，对应于/dev/videoX |
| index | unsigned int | 需要释放的帧的标号 |

【返回值】

|  |  |
| --- | --- |
| **返回值** | **描述** |
| 0 | 接口调用成功 |
| 其他 | 接口调用失败 |

**使用示例**

#include "unitree\_mkp.h"

int main(int argc, char \*argv[]){

int i = 0;

char buffer[256];

unsigned int index;

unsigned int Bytesused;

ut\_cam\_param pCamParam;

ut\_cam\_param GetCamParam;

pCamParam.high = 400;

pCamParam.width = 928;

pCamParam.pix = PIX\_FMT\_MJPEG;

/\* 初始化相机上下文 \*/

UnitreeInitSdkEnv();

/\* 初始化0号相机 \*/

UnitreeInitCamera(0);

/\* 获取0号相机的功能参数（比如mjpeg格式等信息） \*/

PrintCamCapabilities(0);

/\* 设置相机的分辨率，像素格式等信息 \*/

UnitreeSetCamParam(0, &pCamParam);

/\* 获取配置的参数信息，这个信息是相机实际使用的信息，注意，因为硬件限制部分用户配置的分辨率与实际相机使用的分辨率不一致，见章节一描述\*/

UnitreeGetCamParam(0, &GetCamParam);

/\* 配置相机使用的buf数值，建议配置为4 \*/

UnitreeGetCamerbuffer(0, 4);

/\* 相机开始起流 \*/

UnitreeStreamStart(0);

for (i = 0;i < 5;i++){

/\* 该接口每次调用一次就会保存一帧图像，这里连续保存4帧 \*/

UnitreeGetReleaseframe(0);

}

/\* 释放帧 \*/

UnitreeReleaseCamerBuffer(0);

/\* 0相机去初始化 \*/

UnitreeDeInitCamera(0);

/\* 相机上下文环境关闭 \*/

UnitreeDeInitSdkEnv();

return 0;

}



Unitree 技术支持

Unitree Support

**https://www.unitree.com**

内容如有更新，恕不另外通知



微信扫一扫，关注Unitree公众号

您可以在Unitree 官方网站获取更多文档

https://www.unitree.com/download

If you have any questions or suggestions about the manual, please contact us at the following

E-mail address: [support@unitree.cc](mailto:support@unitree.cc)

© 2024 宇树科技 版权所有