

00A0 2203Ξ2200∀2286⊆2713x27FA↔221A√221B√2295⊕2297⊗UTF8gbsn

---

# **MYNT® EYE S SDK Documentation**

**发布 2.3.9**

**MYNTAI**

**2019 年 07 月 12 日**



---

## Contents

---

<b>1 产品</b>	<b>1</b>
1.1 简介	1
1.2 外观	2
1.3 规格	3
1.4 IMU 坐标系统	6
1.5 附件	7
<b>2 SDK</b>	<b>9</b>
2.1 SDK 说明	9
2.2 SDK 安装	10
2.3 SDK 数据样例	20
2.4 SDK 控制样例	34
2.5 SDK 工具	45
2.6 SDK 工程引用	51
2.7 更新日志	61
<b>3 固件</b>	<b>63</b>
3.1 固件说明	63
3.2 固件升级	64
<b>4 使用工具支持</b>	<b>69</b>
4.1 标定工具的使用	69
<b>5 开源项目支持</b>	<b>75</b>
5.1 VINS-Mono 如何整合	75
5.2 VINS-Fusion 如何整合	76
5.3 ORB_SLAM2 如何整合	77
5.4 OKVIS 如何整合	78
5.5 VIORB 如何整合	80
5.6 Maplab 如何整合	81
<b>6 API DOCS</b>	<b>83</b>
6.1 API	83
6.2 Device	87
6.3 Enums	91
6.4 Types	96
6.5 Utils	100

<b>7 技术支持</b>	<b>101</b>
7.1 常见问题 . . . . .	101
7.2 联系我们 . . . . .	101
<b>索引</b>	<b>103</b>

### 1.1 简介

小觅双目摄像头标准版系列包括小觅双目摄像头标准版（MYNT EYE S），小觅双目摄像头标准入门版（MYNT EYE SE），小觅双目摄像头标准彩色版（MYNT EYE SC），小觅双目摄像头（MYNT® EYE）标准版系列采用的“双目 +IMU”的惯性导航方案，可为 vSLAM 的应用提供精准的六轴互补数据，并且相较其他单一方案拥有更高精度和鲁棒性。

结合自研的帧同步、自动曝光及白平衡控制等摄像头技术，小觅双目摄像头（MYNT® EYE）标准版系列提供基于 CUDA 的 GPU 实时加速方案，可以输出高精度同步的图像源，帮助降低算法研发难度，加快算法研发效率。同时，小觅双目摄像头标准版配备六轴传感器（IMU）和红外主动光探测器（IR）。其中，六轴传感器（IMU）可为视觉定位算法的研究提供数据的互补和校正，适用于视觉惯性里程计（VIO）的算法研究，帮助提升定位精度；红外主动光探测器（IR）可以帮助解决室内白墙和无纹理物体的识别难题，提升图像源的识别精度。小觅双目摄像头标准入门版和标准版的区别仅在于入门版不带 IR，为客户提供更低成本的硬件。小觅双目摄像头标准彩色版（MYNT EYE SC）提供 8cm/12cm 可选基线方案，超广角 146°FOV，提供更广阔的深度识别范围和精度水平，产品配备彩色镜头、全新升级 BMI088 六轴 IMU、IR 主动光、I2C 时间同步芯片、全局快门等领先的硬件方案，分辨率最高可高达 2560x800@30fps，精度可达厘米级。此外，小觅双目摄像头标准版系列产品还提供丰富的 SDK 接口和 VSLAM 开源项目支持，可以帮助客户迅速进行方案集成，加速实现产品研发进程，实现方案的快速产品化和落地。

小觅双目摄像头标准版系列可广泛应用于视觉定位导航（vSLAM）领域，包括：无人车和机器人的视觉实时定位导航系统、无人机视觉定位系统、无人驾驶避障导航系统、增强现实（AR）、虚拟现实（VR）等；双目也可应用于视觉识别领域，包括：立体人脸识别、三维物体识别、空间运动追踪、三维手势与体感识别等；应用于测量领域，包括：辅助驾驶系统（ADAS）、双目体积计算、工业视觉筛选等。目前，小觅智能已与国内外 500 余家企事业单位开展了服务与合作。

为保证摄像头产品输出数据质量，产品出厂时，我们已对双目进行标定。同时，产品通过富士康实验室的高温高湿持续工作、高温高湿持续操作、低温动态老化、高温工作、低温存储、整机冷热冲击、正弦振动、随机振动等多项产品质量测试，保证品质的稳定和可靠。除了产品和技术的研发，亦可直接应用于产品量产，加速从研发到产品化的过程。

## 1.2 外观

### 1.2.1 S1030 尺寸与结构

外壳 (mm)	PCBA 板 (mm)
165x31.5x29.6	149x24

## 1.3 规格

### 1.3.1 S1030-120/Mono

#### 产品规格

型号	S1030-120/Mono
尺寸	165x31.5x31.23mm
帧率	10/15/20/25/30/35/40/45/50/55/60FPS
分辨率	752*480; 376*240
深度分辨率	Based on CPU/GPU Up to 752*480@60FPS
像素尺寸	6.0*6.0µm
基线	120.0mm
视角	D:146° H:122° V:76°
焦距	2.1mm
滤镜	Dual Pass Filter
支持 IR	No
IR 可探测距离	-
色彩模式	Monochrome
深度工作距离	0.8-5m+
曝光方式	Global Shutter
功耗	1W@5V DC from USB
同步精度	<1ms (up to 0.05ms)
IMU 频率	100/200/250/333/500Hz
输出数据格式	Raw data
接口	USB3.0
重量	160g
UVC MODE	Yes

#### 软件

支持操作系统	Windows 10、Ubuntu 14.04/16.04/18.04、ROS indigo/kinetic/melodic、Android 7.0+
SDK 地址	<a href="http://www.myntai.com/dev/mynteye">http://www.myntai.com/dev/mynteye</a>
开发者支持	SDK
开源项目支持	ORB_SLAM2、OKVIS、Vins-Mono、Vins-Fusion、VIORB

#### 环境

运行温度	10°C~50°C
存储温度	-20°C~60°C
湿度	10% to 90% non-condensing

#### 包装

包装内容	MYNT EYE x1 USB Micro-B Cable x1
------	----------------------------------

## 保修

产品保修	12 Months Limited Manufacturer's Warranty
------	---

## 精度

深度测量精度	误差不超过 4%
--------	----------

## 1.3.2 S1030-IR-120/Mono

### 产品规格

型号	S1030-IR-120/Mono
尺寸	165x31.5x31.23mm
帧率	10/15/20/25/30/35/40/45/50/55/60FPS
分辨率	752*480; 376*240
深度分辨率	Based on CPU/GPU Up to 752*480@60FPS
像素尺寸	6.0*6.0µm
基线	120.0mm
视角	D:146° H:122° V:76°
焦距	2.1mm
滤镜	Dual Pass Filter
支持 IR	Yes
IR 可探测距离	Up to 3m
色彩模式	Monochrome
深度工作距离	0.8-5m+
曝光方式	Global Shutter
功耗	1~2.7W@5V DC from USB
同步精度	<1ms (up to 0.05ms)
IMU 频率	100/200/250/333/500Hz
输出数据格式	Raw data
接口	USB3.0
重量	184g
UVC MODE	Yes

## 软件

支持操作系统	Windows 10、Ubuntu 14.04/16.04/18.04、ROS indigo/kinetic/melodic、Android 7.0+
SDK 地址	<a href="http://www.myntai.com/dev/mynteye">http://www.myntai.com/dev/mynteye</a>
开发者支持	SDK
开源项目支持	ORB_SLAM2、OKVIS、Vins-Mono、Vins-Fusion、VIORB

## 环境

运行温度	10°C~50°C
存储温度	-20°C~60°C
湿度	10% to 90% non-condensing

## 包装

包装内容	MYNT EYE x1 USB Micro-B Cable x1
------	----------------------------------

## 保修

产品保修	12 Months Limited Manufacturer's Warranty
------	---

## 精度

深度测量精度	误差不超过 4%
--------	----------

## 1.3.3 S2100-146/Color

### 产品规格

型号	S2100-146/Color
尺寸	125x47x26.6mm
帧率	1280x400@10/20/30/60fps 2560x800@10/20/30fps
分辨率	1280x400; 2560x800;
深度分辨率	Based on CPU/GPU Up to 1280*400@60FPS
像素尺寸	3.0*3.0μm
基线	80.0mm
视角	D:141° H:124° V:87°
焦距	0.95mm
支持 IR	NO
色彩模式	Color
深度工作距离	0.26-3m+
快门类型	Global Shutter
功耗	1.1W@5V DC from USB
同步精度	<1ms (up to 0.02ms)
IMU 频率	200Hz
输出数据格式	YUYV
接口	USB3.0
时间同步接口	DF50A
重量	62g
UVC MODE	Yes

## 软件

支持操作系统	Windows 10、Ubuntu 14.04/16.04/18.04、ROS indigo/kinetic/melodic、Android 7.0+
SDK 地址	<a href="http://www.myntai.com/dev/mynteye">http://www.myntai.com/dev/mynteye</a>
开发者支持	SDK
开源项目支持	ORB_SLAM2、OKVIS、Vins-Mono、Vins-Fusion、VIORB

## 环境

运行温度	-15°C~55°C
存储温度	-20°C~75°C
湿度	0% to 95% non-condensing

## 包装

包装内容	MYNT EYE x1 USB Micro-B Cable x1
------	----------------------------------

## 保修

产品保修	12 Months Limited Manufacturer's Warranty
------	---

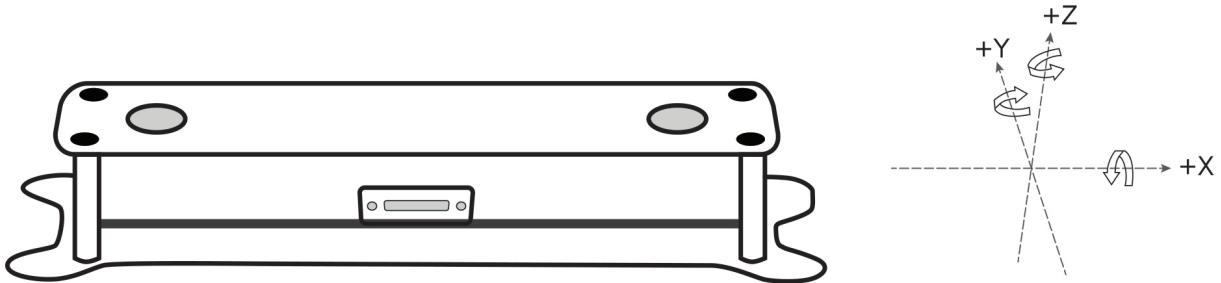
## 精度

深度测量精度	误差不超过 4%
--------	----------

## 1.4 IMU 坐标系统

### 1.4.1 S1030 IMU 坐标系统

IMU 坐标系统为右手系，坐标轴方向如下：



## 1.5 附件

产品附件包含：双目 \*1、数据线 \*1

为了更好的支持您的开发，我们还提供双目支架，您可以在我们的天猫旗舰店 [mynt 小觅旗舰店](#) 进行购买。



## 2.1 SDK 说明

### 2.1.1 支持平台

SDK 是基于 CMake 构建的，用以跨 Linux, Windows 等多个平台。SDK 提供两种安装方式：下载安装以及源码编译安装。

已测试可用的平台有：

- Windows 10
- Ubuntu 18.04.1 / 16.04.6 / 14.04.5
- Jetson TX1/TX2 / Xavier
- firefly RK3399

**警告：**由于硬件传输速率要求，务必使用 USB 3.0 接口。另外，虚拟机因其大多存在 USB 驱动兼容性问题，不建议使用。

### 2.1.2 OpenCV 说明

SDK 提供了三层接口，其 OpenCV 依赖情况如下：

- api，上层接口，依赖 OpenCV 。
- device，中间层接口，不依赖 OpenCV 。
- uvc，底层接口，不依赖 OpenCV 。

如果不想使用 OpenCV，你可编辑 `<sdk>/cmake/Option.cmake` 里的 `WITH_API` 选项，设为 OFF 就能关闭 api 层代码编译：

```
option(WITH_API "Build with API layer, need OpenCV" ON)
```

device 层接口使用样例, 请见 device/camera.cc。

## 2.2 SDK 安装

### 2.2.1 Ubuntu PPA 安装

Ubuntu 14.04	Ubuntu 16.04	Ubuntu 18.04
✓	✓	✓

#### x64 PPA 安装

```
$ sudo add-apt-repository ppa:slichtech/mynt-eye-s-sdk
$ sudo apt-get update
$ sudo apt-get install mynt-eye-s-sdk
```

#### armv8 PPA 安装

```
$ sudo add-apt-repository ppa:slichtech/mynt-eye-s-sdk-arm
$ sudo apt-get update
$ sudo apt-get install mynt-eye-s-sdk
```

#### 运行样例

---

**小技巧:** samples 路径: /opt/mynt-eye-s-sdk/samples; tools 路径: /opt/mynt-eye-s-sdk/tools

---

```
$ cd /opt/mynt-eye-s-sdk/samples
$ ./api/camera_a
```

### 2.2.2 Ubuntu 源码安装

Ubuntu 14.04	Ubuntu 16.04	Ubuntu 18.04
✓	✓	✓

---

**小技巧:** 安装 SDK 前尽量保证系统纯净, 其他第三方软件可能会造成库的冲突。如果是其他 Linux 发行版, 不是用的 apt-get 包管理工具, 那你准备依赖时不能 make init 自动安装, 得自己手动安装了。必要安装项如下:

Linux	How to install required packages
Debian based	sudo apt-get install build-essential cmake git libv4l-dev
Red Hat based	sudo yum install make gcc gcc-c++ kernel-devel cmake git libv4l-devel
Arch Linux	sudo pacman -S base-devel cmake git v4l-utils

## 获取代码

```
sudo apt-get install git
git clone https://github.com/slightech/MYNT-EYE-S-SDK.git
```

## 准备依赖

```
cd <sdk> # <sdk> 是指 sdk 路径
make init
```

- OpenCV

**小技巧:** 如果需要安装 ROS, 可以不用安装 OpenCV/PCL, 以防兼容性问题。OpenCV 如何编译安装, 请见官方文档 [Installation in Linux](#)。或参考如下命令:

```
[compiler] sudo apt-get install build-essential
[required] sudo apt-get install cmake git libgtk2.0-dev pkg-config libavcodec-dev
           ↳ libavformat-dev libswscale-dev
[optional] sudo apt-get install python-dev python-numpy libtbb2 libtbb-dev libjpeg-
           ↳ dev libpng-dev libtiff-dev libjasper-dev libdc1394-22-dev

$ git clone https://github.com/opencv/opencv.git
$ cd opencv/
$ git checkout tags/3.4.1

$ mkdir _build
$ cd _build/

$ cmake \
-DCMAKE_BUILD_TYPE=RELEASE \
-DCMAKE_INSTALL_PREFIX=/usr/local \
\
-DBUILD_DOCS=OFF \
\
-DBUILD_EXAMPLES=OFF \
-DBUILD_TESTS=OFF \
-DBUILD_PERF_TESTS=OFF \
..

$ make -j4
$ sudo make install
```

## 编译代码

---

**小技巧:** 如果 OpenCV 安装到了自定义目录或想指定某一版本, 编译前可如下设置路径:

```
# OpenCV_DIR is the directory where your OpenCVConfig.cmake exists  
export OpenCV_DIR=~/opencv
```

---

不然, CMake 会提示找不到 OpenCV。如果不依賴 OpenCV, 请阅读[OpenCV 说明](#)。

---

编译并安装:

```
cd <sdk>  
make install
```

最终, 默认会安装在 /usr/local 目录。

## 编译样例

```
cd <sdk>  
make samples
```

运行样例:

```
./samples/_output/bin/api/camera_a
```

教程样例, 请阅读[SDK 工程引用](#) 和[SDK 控制样例](#)。

## 编译工具

```
cd <sdk>  
make tools
```

安装脚本依赖:

```
cd <sdk>/tools/  
sudo pip install -r requirements.txt
```

工具和脚本的使用, 后续会有介绍。

## 结语

工程要引入 SDK 的话, CMake 可参考 samples/CMakeLists.txt 里的配置。不然, 就是直接引入安装目录里的头文件和动态库。

### 2.2.3 Windows EXE 安装

Windows 10
✓

## 下载并安装 SDK

**小技巧:** 下载地址: mynteye-s-x.x.x-win-x64-opencv-3.4.3.exe [Google Drive 百度网盘 \(提取码:rj4k\)](#)。

安装完 SDK 的 exe 安装包后, 桌面会生成 SDK 根目录的快捷方式。

**小技巧:** <SDK\_ROOT\_DIR> 是指 exe 包安装路径

进入 <SDK\_ROOT\_DIR>\bin\samples\tutorials 目录, 双击 get\_stereo.exe 运行, 即可看到双目实时画面。

如果样例没有运行成功, 请先检查一下系统变量 PATH 中是否成功添加了 <SDK\_ROOT\_DIR>\3rdparty\opencv\build, <SDK\_ROOT\_DIR>\bin 路径, 如果没有需要手动添加一下。

## 生成样例工程

首先, 安装好 Visual Studio 2017 和 CMake。

接着, 进入 <SDK\_ROOT\_DIR>\samples 目录, 双击 generate.bat 即可生成样例工程。

**小技巧:** 运行样例需要先右键样例, 设为启动项目, 然后使用 Release x64 运行

p.s. 样例教程, 可见 [SDK](#) 主页给出的 Guide 文档。

## 如何于 Visual Studio 2017 下使用 SDK

进入 <SDK\_ROOT\_DIR>\projects\vs2017, 见 README.md 说明。

## 2.2.4 Windows 源码安装

Windows 10
✓

**小技巧:** Windows 不直接提供 Visual Studio \*.sln 工程文件, 需要用 CMake 来构建生成。一是 CMake 跨平台、易配置、可持续维护, 二是第三方代码 (glog, OpenCV) 也都是用的 CMake 构建。

## 前提条件

### CMake (提供构建)

- CMake, 用于构建编译 (必要)。
- Git, 用于获取代码 (可选)。
- Doxygen, 用于生成文档 (可选)。

安装好上述工具后，在命令提示符（Command Prompt）里确认可运行这些命令：

```
>cmake --version  
cmake version 3.10.1  
  
>git --version  
git version 2.11.1.windows.1  
  
>doxygen --version  
1.8.13
```

### Visual Studio (提供编译)

- Visual Studio
  - Visual Studio 2017
  - Visual Studio 2015
- Windows 10 SDK

安装好 Visual Studio 后，在其 Visual Studio Command Prompt 里确认可运行如下命令：

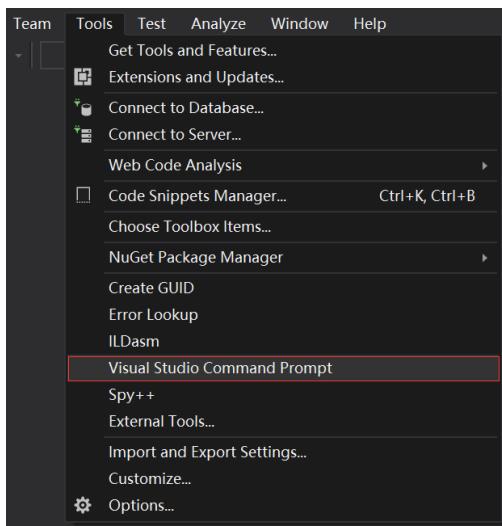
```
>cl  
Microsoft (R) C/C++ Optimizing Compiler Version 19.14.26429.4 for x86  
  
>msbuild  
Microsoft (R) 生成引擎版本 15.7.179.6572
```

---

**小技巧：**Visual Studio Command Prompt 可以从开始菜单打开，



也可以从 Visual Studio 的工具菜单里打开，



但如 Visual Studio 2015 工具菜单里可能没有，可以自己添加个。

打开 Tools 的 External Tools…，然后 Add 如下内容：

Field	Value
Title	Visual Studio Command Prompt
Command	C:\Windows\System32\cmd.exe
Arguments	/k "C:\Program Files (x86)\Microsoft Visual Studio 14.0\Common7\Tools\VsDevCmd.bat"
Initial Directory	\$(SolutionDir)

Visual Studio Command Prompt 里就可以用编译命令 cl link lib msbuild 等 (需要先完成“MSYS2”和“获取代码”步骤)，

The screenshot shows a Windows Command Prompt window with the following text:

```
c:\Program Files (x86)\Microsoft Visual Studio\2017\Community\Common7\Tools>cl  
Microsoft (R) C/C++ Optimizing Compiler Version 19.14.26429.4 for x86  
Copyright (C) Microsoft Corporation. All rights reserved.  
usage: cl [ option... ] filename... [ /link linkoption... ]  
c:\Program Files (x86)\Microsoft Visual Studio\2017\Community\Common7\Tools>msbuild  
用于 .NET Framework 的 Microsoft (R) 生成引擎版本 15.7.179.6572  
版权所有 (C) Microsoft Corporation. 保留所有权利。  
MSBUILD : error MSB1003: 请指定项目或解决方案文件。当前工作目录中未包含项目或解决方案文件。  
c:\Program Files (x86)\Microsoft Visual Studio\2017\Community\Common7\Tools>cd %USERPROFILE%  
C:\Users\John>cd Workspace\Slightech\mynt-eye-sdk-2  
C:\Users\John\Workspace\Slightech\mynt-eye-sdk-2>make host  
Make host  
HOST_OS: Win  
HOST_ARCH: x64  
HOST_NAME: MSYS  
SH: /bin/bash  
ECHO: echo -e  
FIND: C:/msys64/usr/bin/find  
CC: cl  
CXX: cl  
MAKE: make  
BUILD: msbuild.exe ALL_BUILD.vcxproj /property:Configuration=Release  
LDD: ldd  
CMAKE: cmake -DCMAKE_BUILD_TYPE=Release -DCMAKE_C_COMPILER=cl -DCMAKE_CXX_COMPILER=cl -G Visual Studio 15 2017 Win64  
C:\Users\John\Workspace\Slightech\mynt-eye-sdk-2>
```

## MSYS2 (提供 Linux 命令)

- MSYS2
  - 国内镜像
  - pacman

安装后，确认系统环境变量 PATH 里添加了如下路径：

```
C:\msys64\usr\bin
```

然后，打开 MSYS2 MSYS，执行更新并安装 make：

```
$ pacman -Syu  
$ pacman -S make
```

最终，命令提示符（Command Prompt）里可以运行如下命令：

```
>make --version  
GNU Make 4.2.1
```

## 获取代码

```
git clone https://github.com/slightech/MYNT-EYE-S-SDK.git
```

## 准备依赖

```
>cd <sdk> # <sdk> 是指 sdk 路径
>make init
Make init
Init deps
Install cmd: pacman -S
Install deps: git clang-format
pacman -S clang-format (not exists)
error: target not found: clang-format
pip install --upgrade autopep8 cpplint pylint requests
...
Init git hooks
ERROR: clang-format-diff is not installed!
Expect cmake version >= 3.0
cmake version 3.10.1
```

- OpenCV

**小技巧:** OpenCV 官方提供了 exe 进行安装。如果想从源码编译, 请见官方文档 [Installation in Windows](#)。或参考如下命令:

```
>git clone https://github.com/opencv/opencv.git
>cd opencv
>git checkout tags/3.4.1

>cd opencv
>mkdir _build
>cd _build

>cmake ^
-D CMAKE_BUILD_TYPE=RELEASE ^
-D CMAKE_INSTALL_PREFIX=C:/opencv ^
-D WITH_CUDA=OFF ^
-D BUILD_DOCS=OFF ^
-D BUILD_EXAMPLES=OFF ^
-D BUILD_TESTS=OFF ^
-D BUILD_PERF_TESTS=OFF ^
-G "Visual Studio 15 2017 Win64" ^
..

>msbuild ALL_BUILD.vcxproj /property:Configuration=Release
>msbuild INSTALL.vcxproj /property:Configuration=Release

msbuild 路径可参考 ``C:\Program Files (x86)\Microsoft Visual Studio\2017\Community\MSBuild\15.0\Bin\MSBuild.exe``
```

## 编译代码

**小技巧:** 如果 OpenCV 安装到了自定义目录或想指定某一版本, 编译前可如下设置路径:

```
# OpenCV_DIR 为 OpenCVConfig.cmake 所在目录
set OpenCV_DIR=C:\opencv\_build
```

不然， CMake 会提示找不到 OpenCV 。如果不想要依赖 OpenCV ，请阅读 *OpenCV* 说明。

---

编译并安装：

```
cd <sdk>
make install
```

最终，默认会安装在 `<sdk>/_install` 目录。

## 编译样例

```
cd <sdk>
make samples
```

运行样例：

```
.\samples\_output\bin\api\camera_a.bat
```

教程样例，请阅读 [SDK 工程引用](#) 和 [SDK 控制样例](#)。

---

**小技巧：**所有编译出的样例程序 `exe` 都会有个相应的 `bat`。`bat` 会临时设定下系统环境变量，然后再运行 `exe`。所以建议执行 `bat` 运行程序。

如果直接运行 `exe` 的话，可能会报 `dll` 找不到。说明你需要将 `<sdk>\_install\bin %OPENCV_DIR%\bin` 加入到系统环境变量 `PATH` 里。

OpenCV 如何设定环境变量，可见官方文档 [Set the OpenCV environment variable and add it to the systems path](#)。

---

## 编译工具

```
cd <sdk>
make tools
```

工具和脚本的使用，后续会有介绍。

---

**小技巧：**脚本为 Python 实现，需要先安装 Python 及其包管理工具 pip，然后再如下安装依赖：

```
cd <sdk>\tools
pip install -r requirements.txt
```

注：MSYS2 里也带了 Python，但测试未能安装上 matplotlib。

---

## 结语

工程要引入 SDK 的话，CMake 可参考 `samples/CMakeLists.txt` 里的配置。不然，就是直接引入安装目录里的头文件和动态库。

## 2.2.5 ROS Wrapper 安装

ROS Melodic	ROS Kinetic	ROS Indigo
✓	✓	✓

### 环境准备

- ROS

#### ROS Melodic (Ubuntu 18.04)

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
sudo apt-key adv --keyserver hkp://ha.pool.sks-keyserver.net:80 --recv-key
421C365BD9FF1F717815A3895523BAEEB01FA116
sudo apt update
sudo apt install ros-melodic-desktop-full
sudo rosdep init
rosdep update
```

#### ROS Kinetic (Ubuntu 16.04)

```
wget https://raw.githubusercontent.com/oroca/oroca-ros-pkg/master/ros_install.sh && \
chmod 755 ./ros_install.sh && bash ./ros_install.sh catkin_ws kinetic
```

#### ROS Indigo (Ubuntu 14.04)

```
wget https://raw.githubusercontent.com/oroca/oroca-ros-pkg/master/ros_install.sh && \
chmod 755 ./ros_install.sh && bash ./ros_install.sh catkin_ws indigo
```

### 编译代码

```
cd <sdk>
make ros
```

### 运行节点

```
source wrappers/ros-devel/setup.bash
roslaunch mynt_eye_ros_wrapper mynteye.launch # 这个节点没有图像显示
```

运行节点，同时打开 RViz 预览：

```
source wrappers/ros-devel/setup.bash
roslaunch mynt_eye_ros_wrapper display.launch
```

## 测试服务

运行节点，有提供获取设备信息服务，如下测试：

```
$ source wrappers/ros-devel/setup.bash
$ rosrun mynt_eye_ros_wrapper get_device_info.py
LENS_TYPE: 0000
SPEC_VERSION: 1.0
NOMINAL_BASELINE: 120
HARDWARE_VERSION: 2.0
IMU_TYPE: 0000
SERIAL_NUMBER: 0610243700090720
FIRMWARE_VERSION: 2.0
DEVICE_NAME: MYNT-EYE-S1000
```

## 常见问题 - ROS Indigo

### make ros 时 libopencv 找不到

```
make[3]: *** No rule to make target `/usr/lib/x86_64-linux-gnu/libopencv_videostab.so.
˓→2.4.8', needed by `/home/john/Workspace/MYNT-EYE-S-SDK/wrappers/ros-devel/lib/
˓→libmynteye_wrapper.so'. Stop.
```

**Solution 1)** 安装 OpenCV 2:

```
sudo apt-get update
sudo apt-get install libcv-dev
```

**Solution 2)** 安装 OpenCV 3 并重编 cv\_bridge:

```
sudo apt-get install ros-indigo-opencv3

git clone https://github.com/ros-perception/vision_opencv.git
mv vision_opencv/cv_bridge/ MYNT-EYE-S-SDK/wrappers/ros/src/
```

然后，重新 make ros。

## 结语

关于如何使用，请阅读 wrapper\_ros。

## 2.3 SDK 数据样例

### 2.3.1 获取设备信息

通过 API 的 `GetInfo()` 函数，就可以获取当前打开设备的各类信息值。

参考代码片段：

```
auto &&api = API::Create(argc, argv);

LOG(INFO) << "Device name: " << api->GetInfo(Info::DEVICE_NAME);
LOG(INFO) << "Serial number: " << api->GetInfo(Info::SERIAL_NUMBER);
LOG(INFO) << "Firmware version: " << api->GetInfo(Info::FIRMWARE_VERSION);
LOG(INFO) << "Hardware version: " << api->GetInfo(Info::HARDWARE_VERSION);
LOG(INFO) << "Spec version: " << api->GetInfo(Info::SPEC_VERSION);
LOG(INFO) << "Lens type: " << api->GetInfo(Info::LENS_TYPE);
LOG(INFO) << "IMU type: " << api->GetInfo(Info::IMU_TYPE);
LOG(INFO) << "Nominal baseline: " << api->GetInfo(Info::NOMINAL_BASELINE);
```

参考运行结果，于 Linux 上：

```
$ ./samples/_output/bin/tutorials/get_device_info
I0503 16:40:21.109391 32106 utils.cc:13] Detecting MYNT EYE devices
I0503 16:40:21.604116 32106 utils.cc:20] MYNT EYE devices:
I0503 16:40:21.604127 32106 utils.cc:24] index: 0, name: MYNT-EYE-S1000
I0503 16:40:21.604142 32106 utils.cc:30] Only one MYNT EYE device, select index: 0
I0503 16:40:21.615054 32106 get_device_info.cc:10] Device name: MYNT-EYE-S1000
I0503 16:40:21.615113 32106 get_device_info.cc:11] Serial number: 0610243700090720
I0503 16:40:21.615129 32106 get_device_info.cc:12] Firmware version: 2.0
I0503 16:40:21.615139 32106 get_device_info.cc:13] Hardware version: 2.0
I0503 16:40:21.615146 32106 get_device_info.cc:14] Spec version: 1.0
I0503 16:40:21.615155 32106 get_device_info.cc:15] Lens type: 0000
I0503 16:40:21.615164 32106 get_device_info.cc:16] IMU type: 0000
I0503 16:40:21.615171 32106 get_device_info.cc:17] Nominal baseline: 120
```

完整代码样例，请见 [get\\_device\\_info.cc](#)。

### 2.3.2 获取图像标定参数

通过 API 的 `GetIntrinsics()` `GetExtrinsics()` 函数，就可以获取当前打开设备的图像标定参数和相机使用的模型。

**小技巧：**参数模版可以参考 `tools/writer/config` 下的参数文件，其中 S2100/S210A 对应的相机参数在 `tools/writer/config/S210A` `S1030` 对应的相机参数在 `tools/writer/config/S1030` `equidistant` 表示等距模型，`pinhole` 表示针孔模型

**注解：** 相机内参 `Intrinsics`: `k` 表示等距畸变系数，`mu,mv` 对应焦距，`v,u` 对应主点坐标。相机外参 `Extrinsics`(从右目往左目): `rotation` 表示旋转矩阵，`translation` 表示平移矩阵。`D`、`K`、`R`、`P` 分别为畸变参数，内参矩阵，矫正矩阵，投影矩阵。参考链接: [ros CameraInfo](#)

参考代码片段：

```
auto &&api = API::Create(argc, argv);

LOG(INFO) << "Intrinsics left: {" << *api->GetIntrinsicsBase(Stream::LEFT)
    << "}";
LOG(INFO) << "Intrinsics right: {" << *api->GetIntrinsicsBase(Stream::RIGHT)
    << "}";
LOG(INFO) << "Extrinsics right to left: {"
    << api->GetExtrinsics(Stream::RIGHT, Stream::LEFT) << "};"
```

参考运行结果，于 Linux 上：

```
$ ./samples/_output/bin/tutorials/get_img_params
I/utils.cc:48 MYNT EYE devices:
I/utils.cc:51   index: 0, name: MYNT-EYE-S1030, sn: 4B4C192400090712, firmware: 2.4
I/utils.cc:60 Only one MYNT EYE device, select index: 0
I/synthetic.cc:59 camera calib model: kannala_brandt
I/utils.cc:93 MYNT EYE requests:
I/utils.cc:96   index: 0, request: width: 752, height: 480, format: Format::YUYV, ↵
  ↵fps: 60
I/utils.cc:96   index: 1, request: width: 376, height: 240, format: Format::YUYV, ↵
  ↵fps: 60
I/utils.cc:107 There are 2 stream requests, select index:
0
I/get_img_params.cc:44 Intrinsics left: {equidistant, width: 752, height: 480, k2: 0.
  ↵00986113697985857, k3: -0.11937208025856659, k4: 0.19092250072175385, k5: -0.
  ↵10168315832257743, mu: 356.41566867259672335, mv: 356.31078130432149464, u0: 375.
  ↵76739787805968263, v0: 246.20025492033516912}
I/get_img_params.cc:45 Intrinsics right: {equidistant, width: 752, height: 480, k2: -0.02246312175999786, k3: 0.01303393297719630, k4: -0.01735983686524734, k5: 0.
  ↵00675132874903371, mu: 357.96820061652590539, mv: 357.76889287108474491, u0: 397.
  ↵09281703352422710, v0: 258.93978588846073308}
I/get_img_params.cc:46 Extrinsics right to left: {rotation: [0.99997489222742053, 0.
  ↵00041828202737396, -0.00707389248605010, -0.00042920419615213, 0.99999871813992847, ↵
  ↵-0.00154256353448567, 0.00707323819170721, 0.00154556094848940, 0.
  ↵99997378992793495], translation: [-120.01607586757218371, 0.34488126401045993, 0.
  ↵64552185106557303]}
ROSMsgInfoPair:
left:
width: 752, height: 480
distortion_model: KANNALA_BRANDT
D: 0.00986114,-0.119372,0.190923,-0.101683,0,
K: 356.416,0,375.767,0,356.311,246.2,0,0,1,
R: 0.999919,-0.00246361,-0.0124477,0.00245407,0.999997,-0.000781093,0.0124496,0.
  ↵000750482,0.999922,
P: 357.04,0,511.114,0,0,357.04,311.965,0,0,0,1,0,
right:
width: 752, height: 480
distortion_model: KANNALA_BRANDT
D: -0.0224631,0.0130339,-0.0173598,0.00675133,0,
K: 357.968,0,397.093,0,357.769,258.94,0,0,1,
R: 0.999981,-0.00287357,-0.00537853,0.00287782,0.999996,0.000781842,0.00537626,-0.
  ↵000797306,0.999985,
P: 357.04,0,511.114,-42851.3,0,357.04,311.965,0,0,0,1,0,
```

完整代码样例，请见 [get\\_img\\_params.cc](#)。

### 2.3.3 获取 IMU 标定参数

通过 API 的 GetMotionIntrinsics() GetMotionExtrinsics() 函数，就可以获取当前打开设备的 IMU 标定参数。

参考代码片段：

```
auto &&api = API::Create(argc, argv);
```

(下页继续)

(续上页)

```
LOG(INFO) << "Motion intrinsics: {" << api->GetMotionIntrinsics() << "}";
LOG(INFO) << "Motion extrinsics left to imu: {" << api->GetMotionExtrinsics(Stream::LEFT) << "};
```

完整代码样例, 请见 [get\\_imu\\_params.cc](#)。

### 2.3.4 获取双目原始图像

API 提供了 Start() Stop() 函数, 用于开始或停止捕获数据。如果只捕获图像数据的话, 参数用 Source::VIDEO\_STREAMING 即可。

开始捕获数据后, 首先调用 WaitForStreams() 函数, 等待捕获到数据。接着, 通过 GetStreamData() 函数, 就能获取想要的数据了。

参考代码片段:

```
auto &&api = API::Create(argc, argv);

api->Start(Source::VIDEO_STREAMING);

cv::namedWindow("frame");

while (true) {
    api->WaitForStreams();

    auto &&left_data = api->GetStreamData(Stream::LEFT);
    auto &&right_data = api->GetStreamData(Stream::RIGHT);

    cv::Mat img;
    cv::hconcat(left_data.frame, right_data.frame, img);
    cv::imshow("frame", img);

    char key = static_cast<char>(cv::waitKey(1));
    if (key == 27 || key == 'q' || key == 'Q') { // ESC/Q
        break;
    }
}

api->Stop(Source::VIDEO_STREAMING);
```

上述代码, 用了 OpenCV 来显示图像。选中显示窗口时, 按 ESC/Q 就会结束程序。

完整代码样例, 请见 [get\\_stereo.cc](#)。

### 2.3.5 获取双目纠正图像

API 提供的 GetStreamData() 默认仅能获取到硬件的原始数据, 例如双目原始图像。

而双目纠正图像, 属于上层合成数据。此类数据, 需要事先 EnableStreamData() 启用, 然后 GetStreamData() 才能获取到。

另外, WaitForStreams() 等待的是关键原始数据。刚开始时, 合成数据可能还在处理, 取出的是空值, 所以需要判断下不为空。

---

**小技巧:** 如果想要合成数据一生成就立即获取到, 请参阅[从回调接口获取数据](#)。

---

参考代码片段:

```
auto &&api = API::Create(argc, argv);

api->EnableStreamData(Stream::LEFT_RECTIFIED);
api->EnableStreamData(Stream::RIGHT_RECTIFIED);

api->Start(Source::VIDEO_STREAMING);

cv::namedWindow("frame");

while (true) {
    api->WaitForStreams();

    auto &&left_data = api->GetStreamData(Stream::LEFT_RECTIFIED);
    auto &&right_data = api->GetStreamData(Stream::RIGHT_RECTIFIED);

    if (!left_data.frame.empty() && !right_data.frame.empty()) {
        cv::Mat img;
        cv::hconcat(left_data.frame, right_data.frame, img);
        cv::imshow("frame", img);
    }

    char key = static_cast<char>(cv::waitKey(1));
    if (key == 27 || key == 'q' || key == 'Q') { // ESC/Q
        break;
    }
}

api->Stop(Source::VIDEO_STREAMING);
```

上述代码, 用了 OpenCV 来显示图像。选中显示窗口时, 按 ESC/Q 就会结束程序。

完整代码样例, 请见 [get\\_stereo\\_rectified.cc](#)。

## 2.3.6 获取视差图像

视差图像, 属于上层合成数据。需要事先 `EnableStreamData()` 启用, 然后 `GetStreamData()` 获取。另外, 判断不为空后再使用。

详细流程说明, 请参阅[获取双目原始图像](#) [获取双目纠正图像](#)。

另外, 推荐使用插件计算深度: 深度图效果会更好, 并且运算速度更快。具体请参阅[使用插件获取数据](#)。

---

**小技巧:** 其中 `SetDisparityComputingMethodType` 方法用于改变视差计算方式, 目前有 BM/SGBM 两种方式可供选择。

---

参考代码片段:

```
auto &&api = API::Create(argc, argv);

// api->EnableStreamData(Stream::DISPARITY);
```

(下页继续)

(续上页)

```

api->EnableStreamData(Stream::DISPARITY_NORMALIZED);

api->SetDisparityComputingMethodType(DisparityComputingMethod::BM);

api->Start(Source::VIDEO_STREAMING);

cv::namedWindow("frame");
// cv::namedWindow("disparity");
cv::namedWindow("disparity_normalized");

while (true) {
    api->WaitForStreams();

    auto &&left_data = api->GetStreamData(Stream::LEFT);
    auto &&right_data = api->GetStreamData(Stream::RIGHT);

    cv::Mat img;
    cv::hconcat(left_data.frame, right_data.frame, img);
    cv::imshow("frame", img);

    // auto &&disp_data = api->GetStreamData(Stream::DISPARITY);
    // if (!disp_data.frame.empty()) {
    //     cv::imshow("disparity", disp_data.frame);
    // }

    auto &&disp_norm_data = api->GetStreamData(Stream::DISPARITY_NORMALIZED);
    if (!disp_norm_data.frame.empty()) {
        cv::imshow("disparity_normalized", disp_norm_data.frame); // CV_8UC1
    }

    char key = static_cast<char>(cv::waitKey(1));
    if (key == 27 || key == 'q' || key == 'Q') { // ESC/Q
        break;
    }
}

api->Stop(Source::VIDEO_STREAMING);

```

上述代码，用了 OpenCV 来显示图像。选中显示窗口时，按 ESC/Q 就会结束程序。

完整代码样例，请见 [get\\_disparity.cc](#)。

### 2.3.7 获取深度图像

深度图像，属于上层合成数据。需要事先 EnableStreamData() 启用，然后 GetStreamData() 获取。深度数据的格式为 CV\_16UC1。另外，判断不为空后再使用。

详细流程说明，请参阅[获取双目原始图像](#) [获取双目纠正图像](#)。

另外，推荐使用插件计算深度：深度图效果会更好，并且运算速度更快。具体请参阅[使用插件获取数据](#)。

参考代码片段：

```

auto &&api = API::Create(argc, argv);

api->EnableStreamData(Stream::DEPTH);

```

(下页继续)

(续上页)

```

api->Start(Source::VIDEO_STREAMING);

cv::namedWindow("frame");
cv::namedWindow("depth");

while (true) {
    api->WaitForStreams();

    auto &&left_data = api->GetStreamData(Stream::LEFT);
    auto &&right_data = api->GetStreamData(Stream::RIGHT);

    cv::Mat img;
    cv::hconcat(left_data.frame, right_data.frame, img);
    cv::imshow("frame", img);

    auto &&depth_data = api->GetStreamData(Stream::DEPTH);
    if (!depth_data.frame.empty()) {
        cv::imshow("depth", depth_data.frame); // CV_16UC1
    }

    char key = static_cast<char>(cv::waitKey(1));
    if (key == 27 || key == 'q' || key == 'Q') { // ESC/Q
        break;
    }
}

api->Stop(Source::VIDEO_STREAMING);

```

上述代码，用了 OpenCV 来显示图像。选中显示窗口时，按 ESC/Q 就会结束程序。

完整代码样例，请见 [get\\_depth.cc](#)。

---

**小技巧：**预览深度图某区域的值，请见 [get\\_depth\\_with\\_region.cc](#)。

---

## 2.3.8 获取点云图像

点云图像，属于上层合成数据。需要事先 `EnableStreamData()` 启用，然后 `GetStreamData()` 获取。另外，判断不为空后再使用。

详细流程说明，请参阅[获取双目原始图像](#) [获取双目纠正图像](#)。

另外，推荐使用插件计算深度：深度图效果会更好，并且运算速度更快。具体请参阅[使用插件获取数据](#)。

参考代码片段：

```

auto &&api = API::Create(argc, argv);

api->EnableStreamData(Stream::POINTS);

api->Start(Source::VIDEO_STREAMING);

cv::namedWindow("frame");
PCViewer pcviewer;

```

(下页继续)

(续上页)

```

while (true) {
    api->WaitForStreams();

    auto &&left_data = api->GetStreamData(Stream::LEFT);
    auto &&right_data = api->GetStreamData(Stream::RIGHT);

    cv::Mat img;
    cv::hconcat(left_data.frame, right_data.frame, img);
    cv::imshow("frame", img);

    auto &&points_data = api->GetStreamData(Stream::POINTS);
    if (!points_data.frame.empty()) {
        pcviewer.Update(points_data.frame);
    }

    char key = static_cast<char>(cv::waitKey(1));
    if (key == 27 || key == 'q' || key == 'Q') { // ESC/Q
        break;
    }
    if (pcviewer.WasStopped()) {
        break;
    }
}

api->Stop(Source::VIDEO_STREAMING);

```

上述代码，用了 PCL 来显示点云。关闭点云窗口时，也会结束程序。

完整代码样例，请见 [get\\_points.cc](#)。

**注意：**准备好了 PCL 库，编译教程样例时才会有此例子。如果 PCL 库安装到了自定义目录，那么请打开 [tutorials/CMakeLists.txt](#)，找到 `find_package(PCL)`，把 `PCLConfig.cmake` 所在目录添加进 `CMAKE_PREFIX_PATH`。

### 2.3.9 获取 IMU 数据

API 提供了 `Start()` `Stop()` 函数，用于开始或停止捕获数据。要捕获 IMU 数据的话，参数用 `Source::MOTION_TRACKING`。或者 `Source::ALL` 同时捕获图像和 IMU 数据。

开始捕获数据后，需要 `EnableMotionDatas()` 启用缓存，才能通过 `GetMotionDatas()` 函数获取到 IMU 数据。否则，只能通过回调接口得到 IMU 数据，请参阅[从回调接口获取数据](#)。

参考代码片段：

```

auto &&api = API::Create(argc, argv);

// Enable this will cache the motion datas until you get them.
api->EnableMotionDatas();

api->Start(Source::ALL);

CVPainter painter;

cv::namedWindow("frame");

```

(下页继续)

(续上页)

```

while (true) {
    api->WaitForStreams();

    auto &&left_data = api->GetStreamData(Stream::LEFT);
    auto &&right_data = api->GetStreamData(Stream::RIGHT);

    cv::Mat img;
    cv::hconcat(left_data.frame, right_data.frame, img);

    auto &&motion_datas = api->GetMotionDatas();
    /*
    for (auto &&data : motion_datas) {
        LOG(INFO) << "Imu frame_id: " << data imu->frame_id
            << ", timestamp: " << data imu->timestamp
            << ", accel_x: " << data imu->accel[0]
            << ", accel_y: " << data imu->accel[1]
            << ", accel_z: " << data imu->accel[2]
            << ", gyro_x: " << data imu->gyro[0]
            << ", gyro_y: " << data imu->gyro[1]
            << ", gyro_z: " << data imu->gyro[2]
            << ", temperature: " << data imu->temperature;
    }
    */

    painter.DrawImgData(img, *left_data.img);
    if (!motion_datas.empty()) {
        painter.DrawImuData(img, *motion_datas[0].imu);
    }

    cv::imshow("frame", img);

    char key = static_cast<char>(cv::waitKey(1));
    if (key == 27 || key == 'q' || key == 'Q') { // ESC/Q
        break;
    }
}

api->Stop(Source::ALL);

```

上述代码，用了 OpenCV 来显示图像和数据。选中显示窗口时，按 ESC/Q 就会结束程序。

完整代码样例，请见 [get\\_imu.cc](#)。

### 2.3.10 获取时间戳对应的 IMU 数据

如果想让获取到的图像的时间戳，在获取到的 IMU 数据的时间戳中间，保持对应关系，可以通过 API 提供的 `EnableTimestampCorrespondence()` 函数，启用此功能。

参考代码片段：

```

auto &&api = API::Create(argc, argv);

// Enable motion datas with timestamp correspondence of some stream
api->EnableTimestampCorrespondence(Stream::LEFT);

```

(下页继续)

(续上页)

```

api->Start(Source::ALL);

cv::namedWindow("frame");

while (true) {
    api->WaitForStreams();

    auto &&left_data = api->GetStreamData(Stream::LEFT);
    auto &&right_data = api->GetStreamData(Stream::RIGHT);

    auto img_stamp = left_data.img->timestamp;
    LOG(INFO) << "Img timestamp: " << img_stamp
        << ", diff_prev=" << (img_stamp - prev_img_stamp);
    prev_img_stamp = img_stamp;

    cv::Mat img;
    cv::hconcat(left_data.frame, right_data.frame, img);

    auto &&motion_datas = api->GetMotionDatas();
    LOG(INFO) << "Imu count: " << motion_datas.size();
    for (auto &&data : motion_datas) {
        auto imu_stamp = data imu->timestamp;
        LOG(INFO) << "Imu timestamp: " << imu_stamp
            << ", diff_prev=" << (imu_stamp - prev_imu_stamp)
            << ", diff_img=" << (1.f + imu_stamp - img_stamp);
        prev_imu_stamp = imu_stamp;
    }
    LOG(INFO);

    cv::imshow("frame", img);

    char key = static_cast<char>(cv::waitKey(1));
    if (key == 27 || key == 'q' || key == 'Q') { // ESC/Q
        break;
    }
}

api->Stop(Source::ALL);

```

参考运行结果，于 Linux 上：

```

$ ./samples/_output/bin/tutorials/get_imu_correspondence
I/utils.cc:30 Detecting MYNT EYE devices
I/utils.cc:40 MYNT EYE devices:
I/utils.cc:43 index: 0, name: MYNT-EYE-S1030, sn: 0281351000090807
I/utils.cc:51 Only one MYNT EYE device, select index: 0
I/synthetic.cc:126 camera calib model: kannala_brandt
I/utils.cc:79 MYNT EYE devices:
I/utils.cc:82 index: 0, request: width: 752, height: 480, format: Format::YUYV,
    ↵fps: 60
I/utils.cc:87 Only one stream request, select index: 0
I/get_imu_correspondence.cc:50 Img timestamp: 171323050, diff_prev=39990
I/get_imu_correspondence.cc:58 Imu count: 13
I/get_imu_correspondence.cc:61 Imu timestamp: 171318710, diff_prev=171318710, diff_
    ↵img=-4352
I/get_imu_correspondence.cc:61 Imu timestamp: 171320730, diff_prev=2020, diff_img=-
    ↵2320

```

(下页继续)

(续上页)

```
I/get_imu_correspondence.cc:61 Imu timestamp: 171322750, diff_prev=2020, diff_img=-304
I/get_imu_correspondence.cc:61 Imu timestamp: 171324770, diff_prev=2020, diff_img=1712
I/get_imu_correspondence.cc:61 Imu timestamp: 171326790, diff_prev=2020, diff_img=3728
I/get_imu_correspondence.cc:61 Imu timestamp: 171328800, diff_prev=2010, diff_img=5744
I/get_imu_correspondence.cc:61 Imu timestamp: 171330810, diff_prev=2010, diff_img=7760
I/get_imu_correspondence.cc:61 Imu timestamp: 171332840, diff_prev=2030, diff_img=9776
I/get_imu_correspondence.cc:61 Imu timestamp: 171334860, diff_prev=2020, diff_
↪img=11808
I/get_imu_correspondence.cc:61 Imu timestamp: 171336880, diff_prev=2020, diff_
↪img=13824
I/get_imu_correspondence.cc:61 Imu timestamp: 171338900, diff_prev=2020, diff_
↪img=15840
I/get_imu_correspondence.cc:61 Imu timestamp: 171340920, diff_prev=2020, diff_
↪img=17872
I/get_imu_correspondence.cc:61 Imu timestamp: 171342930, diff_prev=2010, diff_
↪img=19872
I/get_imu_correspondence.cc:66
I/get_imu_correspondence.cc:50 Img timestamp: 171403040, diff_prev=79990
I/get_imu_correspondence.cc:58 Imu count: 20
I/get_imu_correspondence.cc:61 Imu timestamp: 171383310, diff_prev=40380, diff_img=-
↪19728
I/get_imu_correspondence.cc:61 Imu timestamp: 171385330, diff_prev=2020, diff_img=-
↪17712
I/get_imu_correspondence.cc:61 Imu timestamp: 171387350, diff_prev=2020, diff_img=-
↪15696
I/get_imu_correspondence.cc:61 Imu timestamp: 171389370, diff_prev=2020, diff_img=-
↪13664
I/get_imu_correspondence.cc:61 Imu timestamp: 171391380, diff_prev=2010, diff_img=-
↪11664
I/get_imu_correspondence.cc:61 Imu timestamp: 171393390, diff_prev=2010, diff_img=-
↪9648
I/get_imu_correspondence.cc:61 Imu timestamp: 171395420, diff_prev=2030, diff_img=-
↪7616
I/get_imu_correspondence.cc:61 Imu timestamp: 171397440, diff_prev=2020, diff_img=-
↪5600
I/get_imu_correspondence.cc:61 Imu timestamp: 171399460, diff_prev=2020, diff_img=-
↪3584
I/get_imu_correspondence.cc:61 Imu timestamp: 171401480, diff_prev=2020, diff_img=-
↪1568
I/get_imu_correspondence.cc:61 Imu timestamp: 171403500, diff_prev=2020, diff_img=464
I/get_imu_correspondence.cc:61 Imu timestamp: 171405510, diff_prev=2010, diff_img=2464
I/get_imu_correspondence.cc:61 Imu timestamp: 171407520, diff_prev=2010, diff_img=4480
I/get_imu_correspondence.cc:61 Imu timestamp: 171409540, diff_prev=2020, diff_img=6496
I/get_imu_correspondence.cc:61 Imu timestamp: 171411570, diff_prev=2030, diff_img=8528
I/get_imu_correspondence.cc:61 Imu timestamp: 171413590, diff_prev=2020, diff_
↪img=10544
I/get_imu_correspondence.cc:61 Imu timestamp: 171415610, diff_prev=2020, diff_
↪img=12576
I/get_imu_correspondence.cc:61 Imu timestamp: 171417630, diff_prev=2020, diff_
↪img=14592
I/get_imu_correspondence.cc:61 Imu timestamp: 171419650, diff_prev=2020, diff_
↪img=16608
I/get_imu_correspondence.cc:61 Imu timestamp: 171421660, diff_prev=2010, diff_
↪img=18624
```

完整代码样例, 请见 `get_imu_correspondence.cc`。

### 2.3.11 从回调接口获取数据

API 提供了 SetStreamCallback() SetMotionCallback() 函数，来设定各类数据的回调。

**注意：**一定不要阻塞回调。如果需要长时间处理数据，请将回调作为数据生产者。

参考代码片段：

```

auto &&api = API::Create(argc, argv);

// Attention: must not block the callbacks.

// Get left image from callback
std::atomic_uint left_count(0);
api->SetStreamCallback(
    Stream::LEFT, [&left_count] (const api::StreamData &data) {
        CHECK_NOTNULL(data.img);
        ++left_count;
    });
}

// Get depth image from callback
api->EnableStreamData(Stream::DEPTH);
std::atomic_uint depth_count(0);
cv::Mat depth;
std::mutex depth_mtx;
api->SetStreamCallback(
    Stream::DEPTH,
    [&depth_count, &depth, &depth_mtx] (const api::StreamData &data) {
        UNUSED(data)
        ++depth_count;
    {
        std::lock_guard<std::mutex> _(depth_mtx);
        depth = data.frame;
    }
});
}

// Get motion data from callback
std::atomic_uint imu_count(0);
std::shared_ptr<mynteye::ImuData> imu;
std::mutex imu_mtx;
api->SetMotionCallback(
    [&imu_count, &imu, &imu_mtx] (const api::MotionData &data) {
        CHECK_NOTNULL(data.imu);
        ++imu_count;
    {
        std::lock_guard<std::mutex> _(imu_mtx);
        imu = data.imu;
    }
});
}

api->Start(Source::ALL);

CVPainter painter;

cv::namedWindow("frame");
cv::namedWindow("depth");

```

(下页继续)

(续上页)

```

unsigned int depth_num = 0;
while (true) {
    api->WaitForStreams();

    auto &&left_data = api->GetStreamData(Stream::LEFT);
    auto &&right_data = api->GetStreamData(Stream::RIGHT);

    // Concat left and right as img
    cv::Mat img;
    cv::hconcat(left_data.frame, right_data.frame, img);

    // Draw img data and size
    painter.DrawImgData(img, *left_data.img);

    // Draw imu data
    if (imu) {
        std::lock_guard<std::mutex> _(imu_mtx);
        painter.DrawImuData(img, *imu);
    }

    // Draw counts
    std::ostringstream ss;
    ss << "left: " << left_count << ", depth: " << depth_count
    << ", imu: " << imu_count;
    painter.DrawText(img, ss.str(), CVPainter::BOTTOM_RIGHT);

    // Show img
    cv::imshow("frame", img);

    // Show depth
    if (!depth.empty()) {
        // Is the depth a new one?
        if (depth_num != depth_count || depth_num == 0) {
            std::lock_guard<std::mutex> _(depth_mtx);
            depth_num = depth_count;
            // LOG(INFO) << "depth_num: " << depth_num;
            ss.str("");
            ss.clear();
            ss << "depth: " << depth_count;
            painter.DrawText(depth, ss.str());
            cv::imshow("depth", depth); // CV_16UC1
        }
    }

    char key = static_cast<char>(cv::waitKey(1));
    if (key == 27 || key == 'q' || key == 'Q') { // ESC/Q
        break;
    }
}

api->Stop(Source::ALL);

```

上述代码，用了 OpenCV 来显示图像和数据。选中显示窗口时，按 ESC/Q 就会结束程序。

完整代码样例，请见 [get\\_from\\_callbacks.cc](#)。

### 2.3.12 使用插件获取数据

API 提供了 `EnablePlugin()` 函数，以启用某路径下的插件。

官方目前提供了些计算双目视差的插件，在 `MYNTEYE_BOX` 的 `Plugins` 目录内。

```
Plugins/
├─linux-x86_64/
| ├─libplugin_b_ocl1.2_opencv3.4.0.so
| ├─libplugin_g_cuda9.1_opencv2.4.13.5.so
| ├─libplugin_g_cuda9.1_opencv3.3.1.so
| └─libplugin_g_cuda9.1_opencv3.4.0.so
└─tegra-armv8/
└─win-x86_64/
```

- 目录 `linux-x86_64` 表明了系统和架构。
  - 可从系统信息或 `uname -a` 得知你的 CPU 架构。
- 库名 `libplugin_*` 表明了插件标识和第三方依赖。
  - `b g` 是插件标识，说明用了不同算法。
  - `ocl1.2` 表明依赖了 OpenCL 1.2，如果存在。
  - `cuda9.1` 表明依赖了 CUDA 9.1，如果存在。
  - `opencv3.4.0` 表明依赖了 OpenCV 3.4.0，如果存在。
  - `mynteye2.0.0` 表明依赖了 MYNT EYE SDK 2.0.0，如果存在。

首先，根据具体情况，选择你想测试使用的插件。如果依赖了第三方，那么请安装一致的版本。

然后，参考如下代码启用插件：

```
auto &&api = API::Create(argc, argv);

api->EnablePlugin("plugins/linux-x86_64/libplugin_g_cuda9.1_opencv3.4.0.so");
```

路径可以是绝对路径，也可以是相对路径（相对于当前工作目录）。

最终，和之前一样调用 API 获取数据就行了。

---

**小技巧：**如果没有启用插件的话，`api->Start(Source::VIDEO_STREAMING)`；时会自动在 `<sdk>/plugins/<platform>` 目录里找合适的插件去加载。

换句话说，可以把当前平台的插件目录整个搬进 `<sdk>/plugins` 目录内。安装好对应的 CUDA OpenCV 等插件依赖后重编译，此后运行 API 层接口程序，就会自动加载官方插件了。

---

运行前，请执行如下命令，以确保能搜索到插件的依赖库：

```
# Linux
export LD_LIBRARY_PATH=/usr/local/lib:$LD_LIBRARY_PATH
# /usr/local/lib 指依赖库所在路径

# macOS
export DYLD_LIBRARY_PATH=/usr/local/lib:$DYLD_LIBRARY_PATH
# /usr/local/lib 指依赖库所在路径

# Windows
```

(下页继续)

(续上页)

```
set PATH=C:\opencv\x64\vc14\bin;%PATH%
# 或者, 添加进系统环境变量 Path 里。
```

此外, 可执行如下命令, 检查是否能搜索到插件的依赖库:

```
# Linux
ldd *.so
# *.so 指具体插件路径

# macOS
otool -L *.dylib
# *.dylib 指具体插件路径

# Windows
# 请下载如 Dependency Walker , 打开 DLL 。
```

如果找不到插件的依赖库, 加载时将会报错 “Open plugin failed”。

完整代码样例, 请见 [get\\_with\\_plugin.cc](#)。

**小技巧:** Linux 上也可以把依赖库路径加入系统环境, 编译出的程序就可以直接运行了 (不需要于终端里 export LD\_LIBRARY\_PATH 再运行)。

- 新建 /etc/ld.so.conf.d/libmynteye.conf 文件, 写入依赖库路径。
- 终端里执行 sudo /sbin/ldconfig 命令, 刷新缓存。

## 2.4 SDK 控制样例

### 2.4.1 设定图像帧率和 IMU 频率

通过 API 的 SetOptionValue() 函数, 就可以设定当前打开设备的各类控制值。

以 s1030 为例, 设定图像帧率和 IMU 频率, 就是设定 Option::FRAME\_RATE 和 Option::IMU\_FREQUENCY。

**注意:**

- 图像帧率有效值: 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60。
- IMU 频率有效值: 100, 200, 250, 333, 500。

以 s2100/s210a 为例, 图像帧率需要在运行样例时选择, 帧率和分辨率选择如下:

```
index: 0, request: width: 1280, height: 400, format: Format::BGR888, fps: 10
index: 1, request: width: 1280, height: 400, format: Format::BGR888, fps: 20
index: 2, request: width: 1280, height: 400, format: Format::BGR888, fps: 30
index: 3, request: width: 1280, height: 400, format: Format::BGR888, fps: 60
index: 4, request: width: 2560, height: 800, format: Format::BGR888, fps: 10
index: 5, request: width: 2560, height: 800, format: Format::BGR888, fps: 20
index: 6, request: width: 2560, height: 800, format: Format::BGR888, fps: 30
```

参考代码片段：

s1030:

```
auto &&api = API::Create(argc, argv);

// Attention: must set FRAME_RATE and IMU_FREQUENCY together, otherwise won't
// succeed.

// FRAME_RATE values: 10, 15, 20, 25, 30, 35, 40, 45, 50, 55
api->SetOptionValue(Option::FRAME_RATE, 25);
// IMU_FREQUENCY values: 100, 200, 250, 333, 500
api->SetOptionValue(Option::IMU_FREQUENCY, 500);

LOG(INFO) << "Set FRAME_RATE to " << api->GetOptionValue(Option::FRAME_RATE);
LOG(INFO) << "Set IMU_FREQUENCY to "
    << api->GetOptionValue(Option::IMU_FREQUENCY);
```

s2100/s210a:

```
auto &&api = API::Create(argc, argv);
if (!api) return 1;

bool ok;
auto &&request = api->SelectStreamRequest(&ok);
if (!ok) return 1;
api->ConfigStreamRequest(request);

LOG(INFO) << "Please set frame rate by 'SelectStreamRequest()'";
```

参考运行结果，于 Linux 上：

s1030:

```
$ ./samples/_output/bin/tutorials/ctrl_framerate
I0513 14:05:57.218222 31813 utils.cc:26] Detecting MYNT EYE devices
I0513 14:05:57.899404 31813 utils.cc:33] MYNT EYE devices:
I0513 14:05:57.899430 31813 utils.cc:37] index: 0, name: MYNT-EYE-S1000
I0513 14:05:57.899435 31813 utils.cc:43] Only one MYNT EYE device, select index: 0
I0513 14:05:58.076257 31813 framerate.cc:36] Set FRAME_RATE to 25
I0513 14:05:58.076836 31813 framerate.cc:37] Set IMU_FREQUENCY to 500
I0513 14:06:21.702361 31813 framerate.cc:82] Time beg: 2018-05-13 14:05:58.384967, ↵
    end: 2018-05-13 14:06:21.666115, cost: 23281.1ms
I0513 14:06:21.702388 31813 framerate.cc:85] Img count: 573, fps: 24.6122
I0513 14:06:21.702404 31813 framerate.cc:87] Imu count: 11509, hz: 494.348
```

s2100/s210a:

```
$ ./samples/_output/bin/tutorials/ctrl_framerate
I/utils.cc:30 Detecting MYNT EYE devices
I/utils.cc:40 MYNT EYE devices:
I/utils.cc:43 index: 0, name: MYNT-EYE-S210A, sn: 07C41A190009071F
I/utils.cc:51 Only one MYNT EYE device, select index: 0
I/utils.cc:79 MYNT EYE devices:
I/utils.cc:82 index: 0, request: width: 1280, height: 400, format: Format::BGR888, ↵
    fps: 10
I/utils.cc:82 index: 1, request: width: 1280, height: 400, format: Format::BGR888, ↵
    fps: 20
I/utils.cc:82 index: 2, request: width: 1280, height: 400, format: Format::BGR888, ↵
    fps: 30
```

(下页继续)

(续上页)

```
I/utils.cc:82    index: 3, request: width: 1280, height: 400, format: Format:::BGR888,_
↳fps: 60
I/utils.cc:82    index: 4, request: width: 2560, height: 800, format: Format:::BGR888,_
↳fps: 10
I/utils.cc:82    index: 5, request: width: 2560, height: 800, format: Format:::BGR888,_
↳fps: 20
I/utils.cc:82    index: 6, request: width: 2560, height: 800, format: Format:::BGR888,_
↳fps: 30
I/utils.cc:93 There are 7 stream requests, select index:
2
I/framerate.cc:54 Please set frame rate by 'SelectStreamRequest()'
I/framerate.cc:99 Time beg: 2018-12-29 10:05:08.203095, end: 2018-12-29 10:08:20.
↳074969, cost: 191872ms
I/framerate.cc:102 Img count: 5759, fps: 30.0148
I/framerate.cc:104 Imu count: 77163, hz: 402.159
```

样例程序按 ESC/Q 结束运行后，会输出计算得的图像帧率和 IMU 频率。

完整代码样例，请见 [framerate.cc](#)。

## 2.4.2 设定加速度计及陀螺仪的量程

通过 API 的 `SetOptionValue()` 函数，就可以设定当前打开设备的各类控制值。

设定 加速度计 及 陀螺仪 的 量程，就是 设定 `Option:::ACCELEROMETER_RANGE` 和 `Option:::GYROSCOPE_RANGE`。

**注意：**s1030 有效值：

- 加速度计量程有效值（单位：g）：4, 8, 16, 32。
- 陀螺仪量程有效值（单位：deg/s）：500, 1000, 2000, 4000。

s2100/s210a 有效值：

- 加速度计量程有效值（单位：g）：6, 12, 24, 48。
- 陀螺仪量程有效值（单位：deg/s）：250, 500, 1000, 2000, 4000。

参考代码片段：

s1030:

```
auto &&api = API::Create(argc, argv);
if (!api)
    return 1;

// ACCELEROMETER_RANGE values: 4, 8, 16, 32
api->SetOptionValue(Option:::ACCELEROMETER_RANGE, 8);
// GYROSCOPE_RANGE values: 500, 1000, 2000, 4000
api->SetOptionValue(Option:::GYROSCOPE_RANGE, 1000);

LOG(INFO) << "Set ACCELEROMETER_RANGE to "
    << api->GetOptionValue(Option:::ACCELEROMETER_RANGE);
LOG(INFO) << "Set GYROSCOPE_RANGE to "
    << api->GetOptionValue(Option:::GYROSCOPE_RANGE);
```

s2100/s210a:

```
auto &&api = API::Create(argc, argv);
if (!api) return 1;

bool ok;
auto &&request = api->SelectStreamRequest(&ok);
if (!ok) return 1;
api->ConfigStreamRequest(request);

// ACCELEROMETER_RANGE values: 6, 12, 24, 48
api->SetOptionValue(Option::ACCELEROMETER_RANGE, 6);
// GYROSCOPE_RANGE values: 250, 500, 1000, 2000, 4000
api->SetOptionValue(Option::GYROSCOPE_RANGE, 1000);

LOG(INFO) << "Set ACCELEROMETER_RANGE to "
      << api->GetOptionValue(Option::ACCELEROMETER_RANGE);
LOG(INFO) << "Set GYROSCOPE_RANGE to "
      << api->GetOptionValue(Option::GYROSCOPE_RANGE);
```

参考运行结果，于 Linux 上：

s1030:

```
$ ./samples/_output/bin/tutorials/ctrl_imu_range
I/utils.cc:28 Detecting MYNT EYE devices
I/utils.cc:38 MYNT EYE devices:
I/utils.cc:41 index: 0, name: MYNT-EYE-S1030, sn: 4B4C1F1100090712
I/utils.cc:49 Only one MYNT EYE device, select index: 0
I/imu_range.cc:34 Set ACCELEROMETER_RANGE to 8
I/imu_range.cc:36 Set GYROSCOPE_RANGE to 1000
I/imu_range.cc:81 Time beg: 2018-11-21 15:34:57.726428, end: 2018-11-21 15:35:12.
˓→190478, cost: 14464ms
I/imu_range.cc:84 Img count: 363, fps: 25.0967
I/imu_range.cc:86 Imu count: 2825, hz: 195.312
```

s2100/s210a:

```
$ ./samples/_output/bin/tutorials/ctrl_imu_range
I/utils.cc:30 Detecting MYNT EYE devices
I/utils.cc:40 MYNT EYE devices:
I/utils.cc:43 index: 0, name: MYNT-EYE-S210A, sn: 07C41A190009071F
I/utils.cc:51 Only one MYNT EYE device, select index: 0
I/utils.cc:79 MYNT EYE devices:
I/utils.cc:82 index: 0, request: width: 1280, height: 400, format: Format::BGR888, ˓→
˓→fps: 10
I/utils.cc:82 index: 1, request: width: 1280, height: 400, format: Format::BGR888, ˓→
˓→fps: 20
I/utils.cc:82 index: 2, request: width: 1280, height: 400, format: Format::BGR888, ˓→
˓→fps: 30
I/utils.cc:82 index: 3, request: width: 1280, height: 400, format: Format::BGR888, ˓→
˓→fps: 60
I/utils.cc:82 index: 4, request: width: 2560, height: 800, format: Format::BGR888, ˓→
˓→fps: 10
I/utils.cc:82 index: 5, request: width: 2560, height: 800, format: Format::BGR888, ˓→
˓→fps: 20
I/utils.cc:82 index: 6, request: width: 2560, height: 800, format: Format::BGR888, ˓→
˓→fps: 30
```

(下页继续)

(续上页)

```
I/utils.cc:93 There are 7 stream requests, select index:  
3  
I imu_range.cc:51 Set ACCELEROMETER_RANGE to 6  
I imu_range.cc:53 Set GYROSCOPE_RANGE to 1000  
I imu_range.cc:98 Time beg: 2018-12-29 10:03:10.706211, end: 2018-12-29 10:04:12.  
→497427, cost: 61791.2ms  
I imu_range.cc:101 Img count: 3706, fps: 59.9762  
I imu_range.cc:103 Imu count: 24873, hz: 402.533
```

样例程序按 ESC/Q 结束运行后，imu 量程设置完成。该结果将固化在硬件内部，不受掉电影响。

完整代码样例，请见 `imu_range.cc`。

### 2.4.3 启用自动曝光及其调节

通过 API 的 `SetOptionValue()` 函数，就可以设定当前打开设备的各类控制值。

启用自动曝光，就是设定 `Option::EXPOSURE_MODE` 为 0。

以 s1030 为例，自动曝光时，可调节的设定有：

- `Option::MAX_GAIN` 最大增益。
- `Option::MAX_EXPOSURE_TIME` 最大曝光时间。
- `Option::DESIRED_BRIGHTNESS` 期望亮度。

以 s2100/s210a 为例，自动曝光可调节的设定有：

- `Option::MAX_GAIN` 最大增益。
- `Option::MAX_EXPOSURE_TIME` 最大曝光时间。
- `Option::DESIRED_BRIGHTNESS` 期望亮度。
- `Option::MIN_EXPOSURE_TIME` 最小曝光时间。

参考代码片段：

s1030:

```
auto &&api = API::Create(argc, argv);

// auto-exposure: 0
api->SetOptionValue(Option::EXPOSURE_MODE, 0);

// max_gain: range [0,48], default 48
api->SetOptionValue(Option::MAX_GAIN, 48);
// max_exposure_time: range [0,240], default 240
api->SetOptionValue(Option::MAX_EXPOSURE_TIME, 240);
// desired_brightness: range [0,255], default 192
api->SetOptionValue(Option::DESIRED_BRIGHTNESS, 192);

LOG(INFO) << "Enable auto-exposure";
LOG(INFO) << "Set MAX_GAIN to " << api->GetOptionValue(Option::MAX_GAIN);
LOG(INFO) << "Set MAX_EXPOSURE_TIME to "
      << api->GetOptionValue(Option::MAX_EXPOSURE_TIME);
LOG(INFO) << "Set DESIRED_BRIGHTNESS to "
      << api->GetOptionValue(Option::DESIRED_BRIGHTNESS);
```

s2100/s210a:

```
auto &&api = API::Create(argc, argv);

bool ok;
auto &&request = api->SelectStreamRequest(&ok);
if (!ok) return 1;
api->ConfigStreamRequest(request);

// auto-exposure: 0
api->SetOptionValue(Option::EXPOSURE_MODE, 0);

// max_gain: range [0,255], default 8
api->SetOptionValue(Option::MAX_GAIN, 8);
// max_exposure_time: range [0,1000], default 333
api->SetOptionValue(Option::MAX_EXPOSURE_TIME, 333);
// desired_brightness: range [1,255], default 122
api->SetOptionValue(Option::DESIRED_BRIGHTNESS, 122);
// min_exposure_time: range [0,1000], default 0
api->SetOptionValue(Option::MIN_EXPOSURE_TIME, 0);

LOG(INFO) << "Enable auto-exposure";
LOG(INFO) << "Set EXPOSURE_MODE to "
    << api->GetOptionValue(Option::EXPOSURE_MODE);
LOG(INFO) << "Set MAX_GAIN to " << api->GetOptionValue(Option::MAX_GAIN);
LOG(INFO) << "Set MAX_EXPOSURE_TIME to "
    << api->GetOptionValue(Option::MAX_EXPOSURE_TIME);
LOG(INFO) << "Set DESIRED_BRIGHTNESS to "
    << api->GetOptionValue(Option::DESIRED_BRIGHTNESS);
LOG(INFO) << "Set MIN_EXPOSURE_TIME to "
    << api->GetOptionValue(Option::MIN_EXPOSURE_TIME);
```

参考运行结果，于 Linux 上：

s1030:

```
$ ./samples/_output/bin/tutorials/ctrl_auto_exposure
I0513 14:07:57.963943 31845 utils.cc:26] Detecting MYNT EYE devices
I0513 14:07:58.457536 31845 utils.cc:33] MYNT EYE devices:
I0513 14:07:58.457563 31845 utils.cc:37] index: 0, name: MYNT-EYE-S1000
I0513 14:07:58.457567 31845 utils.cc:43] Only one MYNT EYE device, select index: 0
I0513 14:07:58.474916 31845 auto_exposure.cc:37] Enable auto-exposure
I0513 14:07:58.491058 31845 auto_exposure.cc:38] Set MAX_GAIN to 48
I0513 14:07:58.505131 31845 auto_exposure.cc:39] Set MAX_EXPOSURE_TIME to 240
I0513 14:07:58.521375 31845 auto_exposure.cc:41] Set DESIRED_BRIGHTNESS to 192
```

s2100/s210a:

```
$ ./samples/_output/bin/tutorials/ctrl_auto_exposure
I/utils.cc:30 Detecting MYNT EYE devices
I/utils.cc:40 MYNT EYE devices:
I/utils.cc:43 index: 0, name: MYNT-EYE-S210A, sn: 07C41A190009071F
I/utils.cc:51 Only one MYNT EYE device, select index: 0
I/utils.cc:79 MYNT EYE devices:
I/utils.cc:82 index: 0, request: width: 1280, height: 400, format: Format::BGR888, ↵
    ↵fps: 10
I/utils.cc:82 index: 1, request: width: 1280, height: 400, format: Format::BGR888, ↵
    ↵fps: 20
```

(下页继续)

(续上页)

```
I/utils.cc:82    index: 2, request: width: 1280, height: 400, format: Format:::BGR888,_
↳fps: 30
I/utils.cc:82    index: 3, request: width: 1280, height: 400, format: Format:::BGR888,_
↳fps: 60
I/utils.cc:82    index: 4, request: width: 2560, height: 800, format: Format:::BGR888,_
↳fps: 10
I/utils.cc:82    index: 5, request: width: 2560, height: 800, format: Format:::BGR888,_
↳fps: 20
I/utils.cc:82    index: 6, request: width: 2560, height: 800, format: Format:::BGR888,_
↳fps: 30
I/utils.cc:93 There are 7 stream requests, select index:
3
I/auto_exposure.cc:72 Enable auto-exposure
I/auto_exposure.cc:73 Set EXPOSURE_MODE to 0
I/auto_exposure.cc:75 Set MAX_GAIN to 8
I/auto_exposure.cc:76 Set MAX_EXPOSURE_TIME to 333
I/auto_exposure.cc:78 Set DESIRED_BRIGHTNESS to 122
I/auto_exposure.cc:80 Set MIN_EXPOSURE_TIME to 0
```

样例程序会显示图像，左上角有真实曝光时间，单位毫秒。

完整代码样例，请见 [auto\\_exposure.cc](#)。

## 2.4.4 启用手动曝光及其调节

通过 API 的 `SetOptionValue()` 函数，就可以设定当前打开设备的各类控制值。

启用手动曝光，就是设定 `Option::EXPOSURE_MODE` 为 1。

以 s1030 为例，手动曝光时，可调节的设定有：

- `Option::GAIN` 增益。
- `Option::BRIGHTNESS` 亮度，或者说曝光时间。
- `Option::CONTRAST` 对比度，或者说黑电平校准。

以 s2100/s210a 为例，手动曝光时，可调节的设定有：

- `Option::BRIGHTNESS` 亮度，或者说曝光时间。

参考代码片段：

s1030:

```
auto &&api = API::Create(argc, argv);

// manual-exposure: 1
api->SetOptionValue(Option::EXPOSURE_MODE, 1);

// gain: range [0,48], default 24
api->SetOptionValue(Option::GAIN, 24);
// brightness/exposure_time: range [0,240], default 120
api->SetOptionValue(Option::BRIGHTNESS, 120);
// contrast/black_level_calibration: range [0,255], default 127
api->SetOptionValue(Option::CONTRAST, 127);

LOG(INFO) << "Enable manual-exposure";
LOG(INFO) << "Set GAIN to " << api->GetOptionValue(Option::GAIN);
```

(下页继续)

(续上页)

```
LOG(INFO) << "Set BRIGHTNESS to " << api->GetOptionValue(Option::BRIGHTNESS);
LOG(INFO) << "Set CONTRAST to " << api->GetOptionValue(Option::CONTRAST);
```

s2100/s210a:

```
auto &&api = API::Create(argc, argv);

bool ok;
auto &&request = api->SelectStreamRequest(&ok);
if (!ok) return 1;
api->ConfigStreamRequest(request);

// manual-exposure: 1
api->SetOptionValue(Option::EXPOSURE_MODE, 1);

// brightness/exposure_time: range [0,240], default 120
api->SetOptionValue(Option::BRIGHTNESS, 120);

LOG(INFO) << "Enable manual-exposure";
LOG(INFO) << "Set EXPOSURE_MODE to "
    << api->GetOptionValue(Option::EXPOSURE_MODE);
LOG(INFO) << "Set BRIGHTNESS to "
    << api->GetOptionValue(Option::BRIGHTNESS);
```

参考运行结果，于 Linux 上：

s1030:

```
$ ./samples/_output/bin/tutorials/ctrl_manual_exposure
I0513 14:09:17.104431 31908 utils.cc:26] Detecting MYNT EYE devices
I0513 14:09:17.501519 31908 utils.cc:33] MYNT EYE devices:
I0513 14:09:17.501551 31908 utils.cc:37] index: 0, name: MYNT-EYE-S1000
I0513 14:09:17.501562 31908 utils.cc:43] Only one MYNT EYE device, select index: 0
I0513 14:09:17.552918 31908 manual_exposure.cc:37] Enable manual-exposure
I0513 14:09:17.552953 31908 manual_exposure.cc:38] Set GAIN to 24
I0513 14:09:17.552958 31908 manual_exposure.cc:39] Set BRIGHTNESS to 120
I0513 14:09:17.552963 31908 manual_exposure.cc:40] Set CONTRAST to 127
```

s2100/s210a:

```
$ ./samples/_output/bin/tutorials/ctrl_manual_exposure
I/utils.cc:30 Detecting MYNT EYE devices
I/utils.cc:40 MYNT EYE devices:
I/utils.cc:43 index: 0, name: MYNT-EYE-S210A, sn: 07C41A190009071F
I/utils.cc:51 Only one MYNT EYE device, select index: 0
I/utils.cc:79 MYNT EYE devices:
I/utils.cc:82 index: 0, request: width: 1280, height: 400, format: Format::BGR888, ↵
    ↵fps: 10
I/utils.cc:82 index: 1, request: width: 1280, height: 400, format: Format::BGR888, ↵
    ↵fps: 20
I/utils.cc:82 index: 2, request: width: 1280, height: 400, format: Format::BGR888, ↵
    ↵fps: 30
I/utils.cc:82 index: 3, request: width: 1280, height: 400, format: Format::BGR888, ↵
    ↵fps: 60
I/utils.cc:82 index: 4, request: width: 2560, height: 800, format: Format::BGR888, ↵
    ↵fps: 10
I/utils.cc:82 index: 5, request: width: 2560, height: 800, format: Format::BGR888, ↵
    ↵fps: 20
```

(下页继续)

(续上页)

```
I/utils.cc:82    index: 6, request: width: 2560, height: 800, format: Format::BGR888,_
→fps: 30
I/utils.cc:93 There are 7 stream requests, select index:
3
I/manual_exposure.cc:62 Enable manual-exposure
I/manual_exposure.cc:63 Set EXPOSURE_MODE to 1
I/manual_exposure.cc:65 Set BRIGHTNESS to 120
```

样例程序会显示图像，左上角有真实曝光时间，单位毫秒。

完整代码样例，请见 [manual\\_exposure.cc](#)。

## 2.4.5 启用 IR 及其调节

通过 API 的 `SetOptionValue()` 函数，就可以设定当前打开设备的各类控制值。

启用 IR，就是设定 `Option::IR_CONTROL` 大于 0 的值。值越大，强度越高。

**注意:**

- s2100/s210a 不支持此功能

参考代码片段：

```
auto &&api = API::Create(argc, argv);

// Detect infrared add-ons
LOG(INFO) << "Support infrared: " << std::boolalpha
    << api->Supports(AddOns::INFRARED);
LOG(INFO) << "Support infrared2: " << std::boolalpha
    << api->Supports(AddOns::INFRARED2);

// Get infrared intensity range
auto &&info = api->GetOptionInfo(Option::IR_CONTROL);
LOG(INFO) << Option::IR_CONTROL << ":" << info << "}";

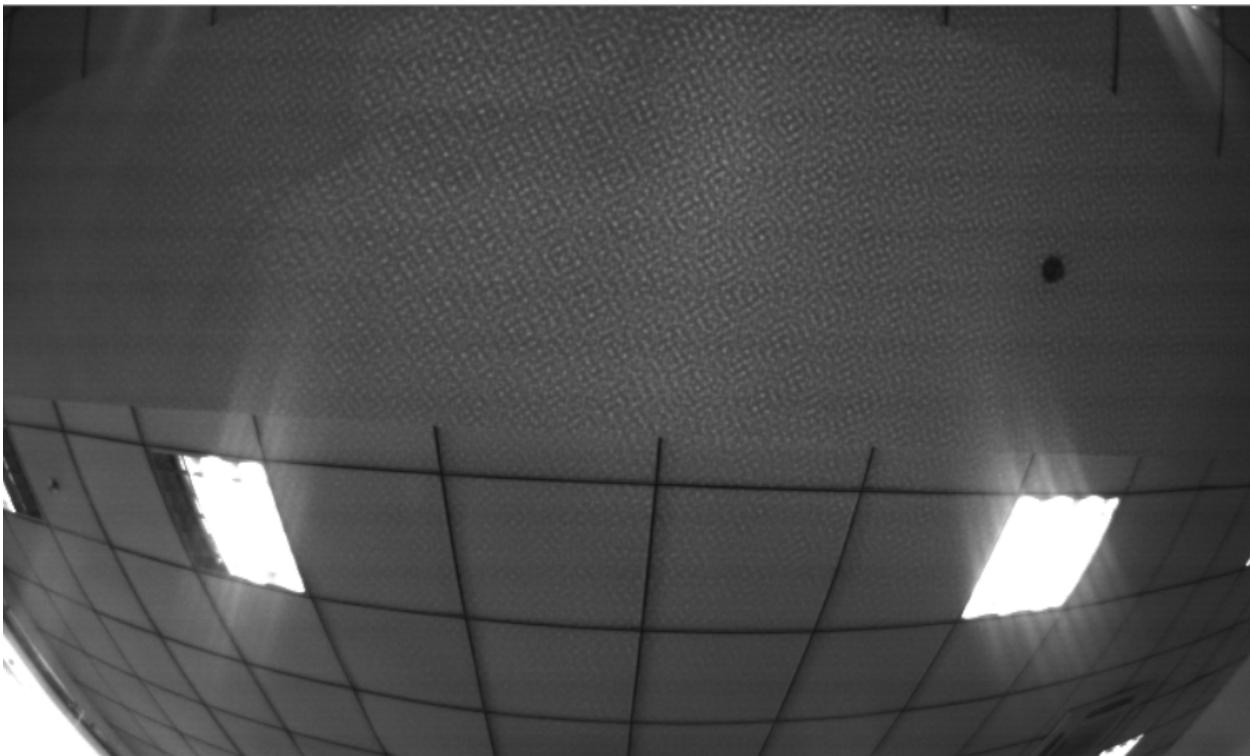
// Set infrared intensity value
api->SetOptionValue(Option::IR_CONTROL, 80);
```

参考运行结果，于 Linux 上：

```
$ ./samples/_output/bin/tutorials/ctrl_infrared
I0504 16:16:28.016624 25848 utils.cc:13] Detecting MYNT EYE devices
I0504 16:16:28.512462 25848 utils.cc:20] MYNT EYE devices:
I0504 16:16:28.512473 25848 utils.cc:24] index: 0, name: MYNT-EYE-S1000
I0504 16:16:28.512477 25848 utils.cc:30] Only one MYNT EYE device, select index: 0
I0504 16:16:28.520848 25848 infrared.cc:13] Support infrared: true
I0504 16:16:28.520869 25848 infrared.cc:15] Support infrared2: true
I0504 16:16:28.520889 25848 infrared.cc:20] Option::IR_CONTROL: {min: 0, max: 160,_
→def: 0}
```

此时，如果显示了图像，就能够看到图像上会有 IR 光斑，如下图：

 frame



**注意:** 硬件不会记忆 IR 值，断电会忘掉。如果需要保持启用 IR 的话，程序在打开设备后，一定要设定下 IR 值。

完整代码样例，请见 [infrared.cc](#)。

#### 2.4.6 低通滤波

通过 API 的 `SetOptionValue()` 函数，就可以设定当前打开设备的各类控制值。

设定 加速度计 低通滤波寄存器值 及 陀螺仪 低通滤波寄存器值，就是设定 `Option::ACCELEROMETER_LOW_PASS_FILTER` 和 `Option::GYROSCOPE_LOW_PASS_FILTER`。

**注意:**

- s1030 不支持此功能

参考代码片段：

```
auto &&api = API::Create(argc, argv);
if (!api) return 1;

bool ok;
auto &&request = api->SelectStreamRequest(&ok);
if (!ok) return 1;
api->ConfigStreamRequest(request);
```

(下页继续)

(续上页)

```
// ACCELEROMETER_RANGE values: 0, 1, 2
api->SetOptionValue(Option::ACCELEROMETER_LOW_PASS_FILTER, 2);
// GYROSCOPE_RANGE values: 23, 64
api->SetOptionValue(Option::GYROSCOPE_LOW_PASS_FILTER, 64);

LOG(INFO) << "Set ACCELEROMETER_LOW_PASS_FILTER to "
    << api->GetOptionValue(Option::ACCELEROMETER_LOW_PASS_FILTER);
LOG(INFO) << "Set GYROSCOPE_LOW_PASS_FILTER to "
    << api->GetOptionValue(Option::GYROSCOPE_LOW_PASS_FILTER);
```

参考运行结果，于 Linux 上：

```
$ ./samples/_output/bin/tutorials/ctrl_imu_low_pass_filter
I/utils.cc:30 Detecting MYNT EYE devices
I/utils.cc:40 MYNT EYE devices:
I/utils.cc:43 index: 0, name: MYNT-EYE-S210A, sn: 07C41A190009071F
I/utils.cc:51 Only one MYNT EYE device, select index: 0
I/utils.cc:79 MYNT EYE devices:
I/utils.cc:82 index: 0, request: width: 1280, height: 400, format: Format::BGR888,
    ↪fps: 10
I/utils.cc:82 index: 1, request: width: 1280, height: 400, format: Format::BGR888,
    ↪fps: 20
I/utils.cc:82 index: 2, request: width: 1280, height: 400, format: Format::BGR888,
    ↪fps: 30
I/utils.cc:82 index: 3, request: width: 1280, height: 400, format: Format::BGR888,
    ↪fps: 60
I/utils.cc:82 index: 4, request: width: 2560, height: 800, format: Format::BGR888,
    ↪fps: 10
I/utils.cc:82 index: 5, request: width: 2560, height: 800, format: Format::BGR888,
    ↪fps: 20
I/utils.cc:82 index: 6, request: width: 2560, height: 800, format: Format::BGR888,
    ↪fps: 30
I/utils.cc:93 There are 7 stream requests, select index:
1
I imu_low_pass_filter.cc:48 Set ACCELEROMETER_LOW_PASS_FILTER to 2
I imu_low_pass_filter.cc:50 Set GYROSCOPE_LOW_PASS_FILTER to 64
I imu_low_pass_filter.cc:96 Time beg: 2018-12-29 13:53:42.296299, end: 2018-12-29
    ↪14:06:33.295960, cost: 771000ms
I imu_low_pass_filter.cc:99 Img count: 15412, fps: 19.9896
I imu_low_pass_filter.cc:101 Imu count: 309891, hz: 401.934
```

样例程序按 ESC/Q 结束运行后，imu 低通滤波设置完成。该结果将固化在硬件内部，不受掉电影响。

完整代码样例，请见 [imu\\_low\\_pass\\_filter.cc](#)。

## 2.4.7 设定 IIC 地址

通过 API 的 `SetOptionValue()` 函数，就可以设定当前打开设备的各类控制值。

设定 IIC 地址，就是设定 `Option::IIC_ADDRESS_SETTING`。

**注意：**仅支持 S210A/S2100

参考代码片段：

s210a/s2100:

```
auto &&api = API::Create(argc, argv);
if (!api) return 1;
bool ok;
auto &&request = api->SelectStreamRequest(&ok);
if (!ok) return 1;
api->ConfigStreamRequest(request);
Model model = api->GetModel();
if (model == Model::STANDARD210A || model == Model::STANDARD2) {
    api->SetOptionValue(Option::IIC_ADDRESS_SETTING, 0x31);
    LOG(INFO) << "Set iic address to " << std::hex << "0x"
        << api->GetOptionValue(Option::IIC_ADDRESS_SETTING);
}
```

参考运行结果，于 Linux 上：

s210a/s2100:

```
$ ./samples/_output/bin/tutorials/ctrl_iic_adress
I/utils.cc:30 Detecting MYNT EYE devices
I/utils.cc:40 MYNT EYE devices:
I/utils.cc:43 index: 0, name: MYNT-EYE-S210A, sn: 07C41A190009071F
I/utils.cc:51 Only one MYNT EYE device, select index: 0
I/utils.cc:79 MYNT EYE devices:
I/utils.cc:82 index: 0, request: width: 1280, height: 400, format: Format::BGR888, ↵
    ↵fps: 10
I/utils.cc:82 index: 1, request: width: 1280, height: 400, format: Format::BGR888, ↵
    ↵fps: 20
I/utils.cc:82 index: 2, request: width: 1280, height: 400, format: Format::BGR888, ↵
    ↵fps: 30
I/utils.cc:82 index: 3, request: width: 1280, height: 400, format: Format::BGR888, ↵
    ↵fps: 60
I/utils.cc:82 index: 4, request: width: 2560, height: 800, format: Format::BGR888, ↵
    ↵fps: 10
I/utils.cc:82 index: 5, request: width: 2560, height: 800, format: Format::BGR888, ↵
    ↵fps: 20
I/utils.cc:82 index: 6, request: width: 2560, height: 800, format: Format::BGR888, ↵
    ↵fps: 30
I/utils.cc:93 There are 7 stream requests, select index:
3
I imu_range.cc:51 Set iic address to 0x31
```

样例程序按 ESC/Q 结束。

完整代码样例，请见[“iic\\_address.cc”](https://github.com/slightech/MYNT-EYE-S-SDK/blob/master/samples/tutorials/control/iic_address.cc)。

## 2.5 SDK 工具

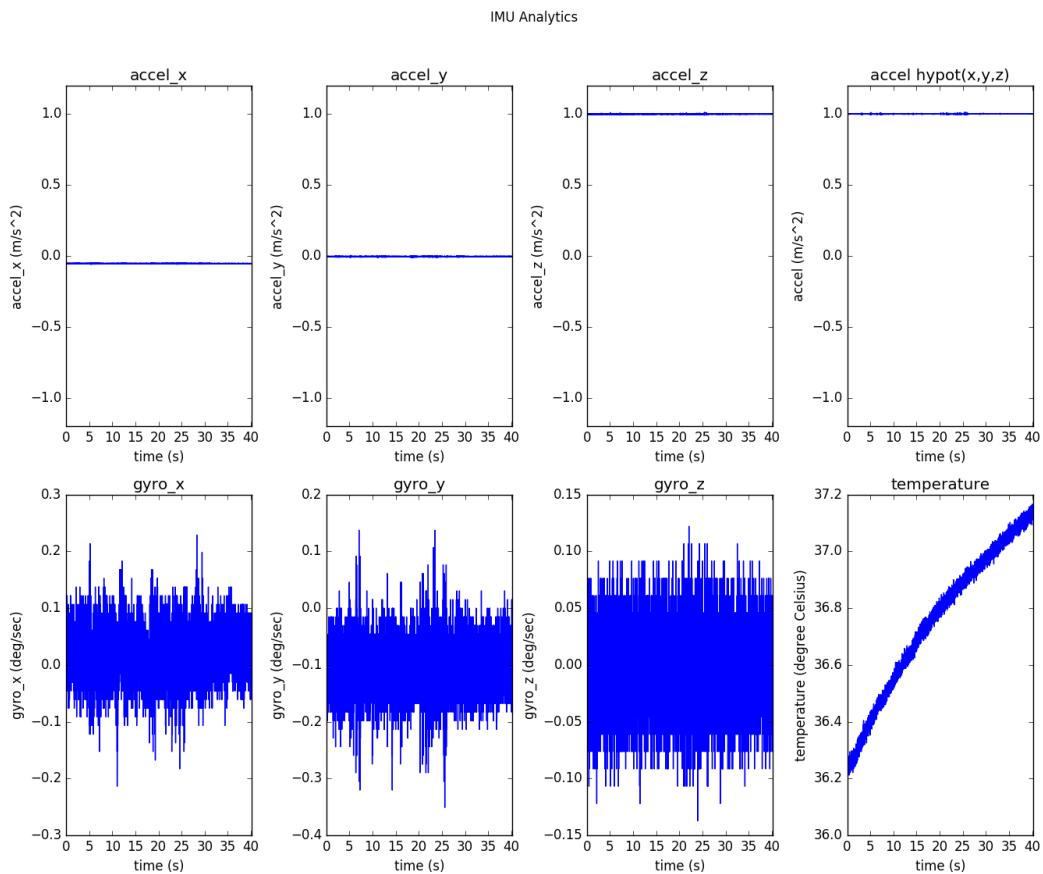
### 2.5.1 分析 IMU

SDK 提供了 IMU 分析的脚本 `imu_analytics.py`。工具详情可见 `tools/README.md`。

参考运行命令及结果，于 Linux 上：

```
$ python tools/analytics imu_analytics.py -i dataset -c tools/config/mynteye/mynteye_
→config.yaml -al=-1.2,1.2 -gl= -gdu=d -gsu=d -kl=
imu analytics ...
  input: dataset
  outdir: dataset
  gyro_limits: None
  accel_limits: [(-1.2, 1.2), (-1.2, 1.2), (-1.2, 1.2), (-1.2, 1.2)]
  time_unit: None
  time_limits: None
  auto: False
  gyro_show_unit: d
  gyro_data_unit: d
  temp_limits: None
open dataset ...
  imu: 20040, temp: 20040
  timebeg: 4.384450, timeend: 44.615550, duration: 40.231100
save figure to:
  dataset/imu_analytics.png
imu analytics done
```

分析结果图会保存进数据集目录，参考如下：



另外，脚本具体选项可执行 `-h` 了解：

```
$ python tools/analytics imu_analytics.py -h
```

## 2.5.2 分析时间戳

SDK 提供了时间戳分析的脚本 stamp\_analytics.py。工具详情可见 tools/README.md。

参考运行命令及结果，于 Linux 上：

```
$ python tools/analytics/stamp_analytics.py -i dataset -c tools/config/mynteye/
˓→mynteye_config.yaml
stamp analytics ...
    input: dataset
    outdir: dataset
open dataset ...
save to binary files ...
    binimg: dataset/stamp_analytics_img.bin
    binimu: dataset/stamp_analytics_imu.bin
    img: 1007, imu: 20040

rate (Hz)
    img: 25, imu: 500
sample period (s)
    img: 0.04, imu: 0.002

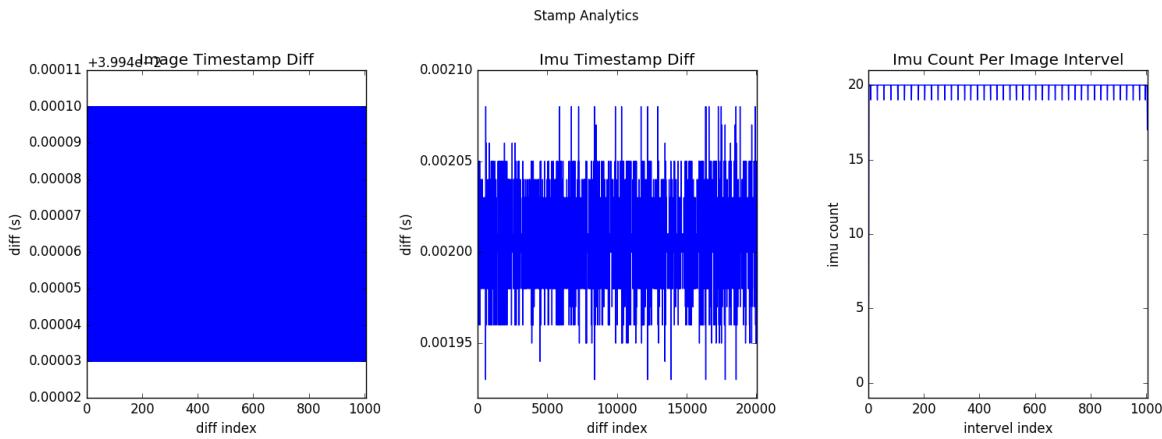
diff count
    imgs: 1007, imus: 20040
    imgs_t_diff: 1006, imus_t_diff: 20039

diff where (factor=0.1)
    imgs where diff > 0.04*1.1 (0)
    imgs where diff < 0.04*0.9 (0)
    imus where diff > 0.002*1.1 (0)
    imus where diff < 0.002*0.9 (0)

image timestamp duplicates: 0

save figure to:
    dataset/stamp_analytics.png
stamp analytics done
```

分析结果图会保存进数据集目录，参考如下：



另外，脚本具体选项可执行 -h 了解：

```
$ python tools/analytics/stamp_analytics.py -h
```

**小技巧：**录制数据集时建议 record.cc 里注释显示图像 cv::imshow()，dataset.cc 里注释存储图像 cv::imwrite()。因为此些操作都比较耗时，可能会导致丢弃图像。换句话说就是消费赶不上生产，所以丢弃了部分图像。record.cc 里用的 GetStreamDatas() 仅缓存最新的 4 张图像。

### 2.5.3 保存设备信息和参数

SDK 提供了保存信息和参数的工具 save\_all\_infos。工具详情可见 tools/README.md。

参考运行命令：

```
./tools/_output/bin/writer/save_all_infos  

# Windows  

.\tools\_output\bin\writer\save_all_infos.bat
```

参考运行结果，于 Linux 上：

```
$ ./tools/_output/bin/writer/save_all_infos  

I0512 21:40:08.687088 4092 utils.cc:26] Detecting MYNT EYE devices  

I0512 21:40:09.366693 4092 utils.cc:33] MYNT EYE devices:  

I0512 21:40:09.366734 4092 utils.cc:37] index: 0, name: MYNT-EYE-S1000  

I0512 21:40:09.366757 4092 utils.cc:43] Only one MYNT EYE device, select index: 0  

I0512 21:40:09.367609 4092 save_all_infos.cc:38] Save all infos to "config/  

→SN0610243700090720"
```

默认会保存进 <workdir>/config 目录。你也可以加参数，指定保存到其他目录。

保存内容如下：

```
<workdir>/  
└ config/  
  └ SN0610243700090720/  
    ├ device.info  
    ├ img.params  
    └ imu.params
```

## 2.5.4 写入图像标定参数

SDK 提供了写入图像标定参数的工具 `img_params_writer`。工具详情可见 `tools/README.md`。

有关如何获取, 请阅读 [获取图像标定参数](#)。此参数会用于计算纠正、视差等。

参考运行命令:

```
./tools/_output/bin/writer/img_params_writer tools/writer/config/S1030/img.params.  
→equidistant

# Windows
.\tools\_output\bin\writer\img_params_writer.bat tools\writer\config\s1030\img.params.  
→equidistant
```

**警告:** 请不要随意覆写参数。另外 `save_all_infos` 工具可帮你备份参数。

其中, `tools/writer/config/S1030/img.params.pinhole` 是 s1030 针孔模型参数文件路径。如果你自己标定了参数, 可以编辑此文件, 然后执行上述命令写入设备。

---

**小技巧:** S2100/S210A 对应的相机参数在 `tools/writer/config/S210A` S1030 对应的相机参数在 `tools/writer/config/S1030` 其中 `equidistant` 表示等距模型, `pinhole` 表示针孔模型

---

**小技巧:** 旧 SDK 提供的标定参数文件 `SN*.conf` 也可用此工具写入设备。

---

## 2.5.5 写入 IMU 标定参数

SDK 提供了写入 IMU 标定参数的工具 `imu_params_writer`。工具详情可见 `tools/README.md`。

有关如何获取, 请阅读 [IMU 标定参数](#)。

参考运行命令:

```
./tools/_output/bin/writer imu_params_writer tools/writer/config/S210A imu.params

# Windows
.\tools\_output\bin\writer\imu_params_writer.bat tools\writer\config\s210A\imu.params
```

其中, `tools/writer/config/imu.params` 是参数文件路径。如果你自己标定了参数, 可以编辑此文件, 然后执行上述命令写入设备。

**警告:** 请不要随意覆写参数。另外 `save_all_infos` 工具可帮你备份参数。

## 2.5.6 录制数据集

SDK 提供了录制数据集的工具 `record`。工具详情可见 `tools/README.md`。

参考运行命令:

```
./tools/_output/bin/dataset/record2

# Windows
.\tools\_output\bin\dataset\record2.bat
```

参考运行结果，于 Linux 上：

```
$ ./tools/_output/bin/dataset/record
I0513 21:28:57.128947 11487 utils.cc:26] Detecting MYNT EYE devices
I0513 21:28:57.807116 11487 utils.cc:33] MYNT EYE devices:
I0513 21:28:57.807155 11487 utils.cc:37] index: 0, name: MYNT-EYE-S1000
I0513 21:28:57.807163 11487 utils.cc:43] Only one MYNT EYE device, select index: 0
I0513 21:28:57.808437 11487 channels.cc:114] Option::GAIN: min=0, max=48, def=24, ↵
↪cur=24
I0513 21:28:57.809999 11487 channels.cc:114] Option::BRIGHTNESS: min=0, max=240, ↵
↪def=120, cur=120
I0513 21:28:57.818678 11487 channels.cc:114] Option::CONTRAST: min=0, max=255, ↵
↪def=127, cur=127
I0513 21:28:57.831529 11487 channels.cc:114] Option::FRAME_RATE: min=10, max=60, ↵
↪def=25, cur=25
I0513 21:28:57.848914 11487 channels.cc:114] Option::IMU_FREQUENCY: min=100, max=500, ↵
↪def=200, cur=500
I0513 21:28:57.865185 11487 channels.cc:114] Option::EXPOSURE_MODE: min=0, max=1, ↵
↪def=0, cur=0
I0513 21:28:57.881434 11487 channels.cc:114] Option::MAX_GAIN: min=0, max=48, def=48, ↵
↪cur=48
I0513 21:28:57.897598 11487 channels.cc:114] Option::MAX_EXPOSURE_TIME: min=0, ↵
↪max=240, def=240, cur=240
I0513 21:28:57.913918 11487 channels.cc:114] Option::DESIRED_BRIGHTNESS: min=0, ↵
↪max=255, def=192, cur=192
I0513 21:28:57.930177 11487 channels.cc:114] Option::IR_CONTROL: min=0, max=160, ↵
↪def=0, cur=0
I0513 21:28:57.946341 11487 channels.cc:114] Option::HDR_MODE: min=0, max=1, def=0, ↵
↪cur=0
Saved 1007 imgs, 20040 imus to ./dataset
I0513 21:29:38.608772 11487 record.cc:118] Time beg: 2018-05-13 21:28:58.255395, end: ↵
↪2018-05-13 21:29:38.578696, cost: 40323.3ms
I0513 21:29:38.608853 11487 record.cc:121] Img count: 1007, fps: 24.9732
I0513 21:29:38.608873 11487 record.cc:123] Imu count: 20040, hz: 496.983
```

默认录制进 <workdir>/dataset 目录。你也可以加参数，指定录制到其他目录。

录制内容如下：

```
<workdir>/
└dataset/
  └left/
    ├stream.txt # Image infomation
    ├000000.png # Image, index 0
    └...
  └right/
    ├stream.txt # Image information
    ├000000.png # Image, index 0
    └...
  motion.txt # IMU information
```

## 2.6 SDK 工程引用

### 2.6.1 CMake 如何使用 SDK

本教程将使用 CMake 创建一个项目来使用 SDK。

你可以在 <sdk>/platforms/projects/cmake 目录下找到工程样例。

#### 准备

- Windows: 安装 SDK 的 exe 包
- Linux: 使用源代码编译和 make install

#### 创建项目

添加 CMakeLists.txt 和 mynteye\_demo.cc 文件，

```
cmake_minimum_required(VERSION 3.0)

project(mynteyed_demo VERSION 1.0.0 LANGUAGES C CXX)

# flags

set(CMAKE_C_FLAGS "${CMAKE_C_FLAGS} -Wall -O3")
set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -Wall -O3")

set(CMAKE_C_FLAGS "${CMAKE_C_FLAGS} -std=c++11 -march=native")
set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11 -march=native")

## mynteye_demo

add_executable(mynteye_demo mynteye_demo.cc)
target_link_libraries(mynteye_demo mynteye ${OpenCV_LIBS})
```

#### 配置项目

增加 mynteye 和 OpenCV 到 CMakeLists.txt，

```
# packages

if(MSVC)
    set(SDK_ROOT "$ENV{MYNTEYES_SDK_ROOT}")
    if(SDK_ROOT)
        message STATUS "MYNTEYES_SDK_ROOT: ${SDK_ROOT}")
        list(APPEND CMAKE_PREFIX_PATH
            "${SDK_ROOT}/lib/cmake"
            "${SDK_ROOT}/3rdparty/opencv/build"
        )
    else()
        message(FATAL_ERROR "MYNTEYES_SDK_ROOT not found, please install SDK firstly")
    endif()
endif()
```

(下页继续)

(续上页)

```

## mynteye

find_package(mynteye REQUIRED)
message(STATUS "Found mynteye: ${mynteye_VERSION}")

# When SDK build with OpenCV, we can add WITH_OPENCV macro to enable some
# features depending on OpenCV, such as ToMat().
if(mynteye_WITH_OPENCV)
    add_definitions(-DWITH_OPENCV)
endif()

## OpenCV

# Set where to find OpenCV
#set(OpenCV_DIR "/usr/share/OpenCV")

# When SDK build with OpenCV, we must find the same version here.
find_package(OpenCV REQUIRED)
message(STATUS "Found OpenCV: ${OpenCV_VERSION}")

```

将 include\_directories 和 target\_link\_libraries 添加到 mynteye\_demo 目标,

```

# targets

include_directories(
    ${OpenCV_INCLUDE_DIRS}
)

## mynteye_demo

add_executable(mynteye_demo mynteye_demo.cc)
target_link_libraries(mynteye_demo mynteye ${OpenCV_LIBS})

```

## 使用 SDK

可以参考工程样例添加头文件和使用 API。

### Windows

可以参考 [Windows EXE 安装](#), 安装编译工具。

然后打开 “x64 Native Tools Command Prompt for VS 2017” 命令行来编译和运行,

```

mkdir _build
cd _build

cmake -G "Visual Studio 15 2017 Win64" ..

msbuild.exe ALL_BUILD.vcxproj /property:Configuration=Release

.\Release\mynteye_demo.exe

```

## Linux

打开命令行来编译和运行，

```
mkdir _build
cd _build/
cmake ..
make
./mynteye_demo
```

## 2.6.2 Visual Studio 2017 如何使用 SDK

本教程将使用 Visual Studio 2017 创建一个项目来使用 SDK。

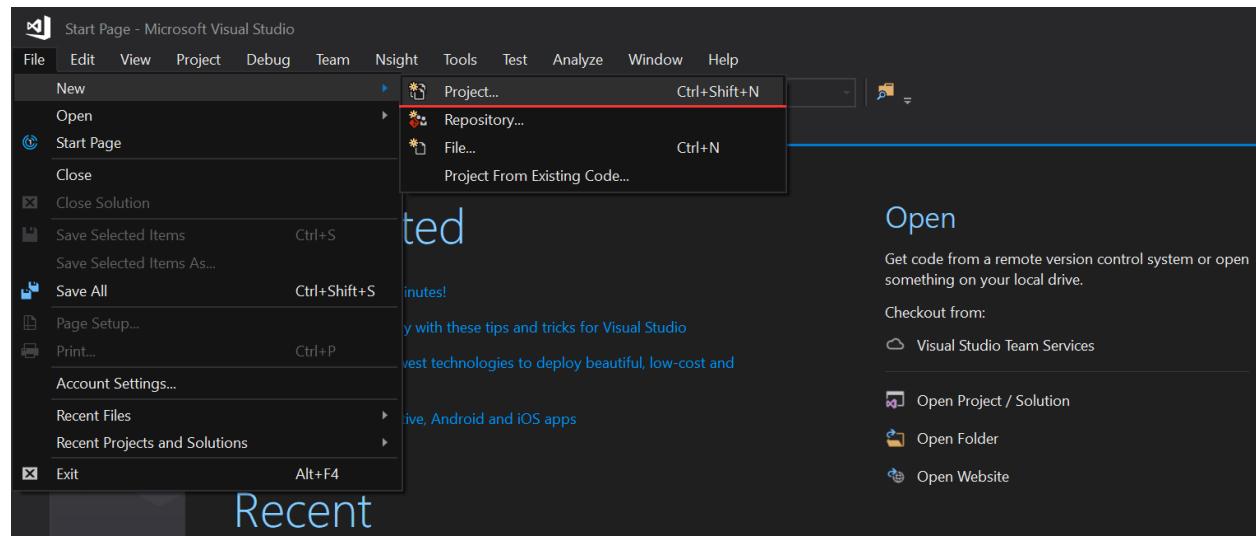
你可以在 <sdk>/platforms/projects/vs2017 目录下找到工程样例。

### 准备

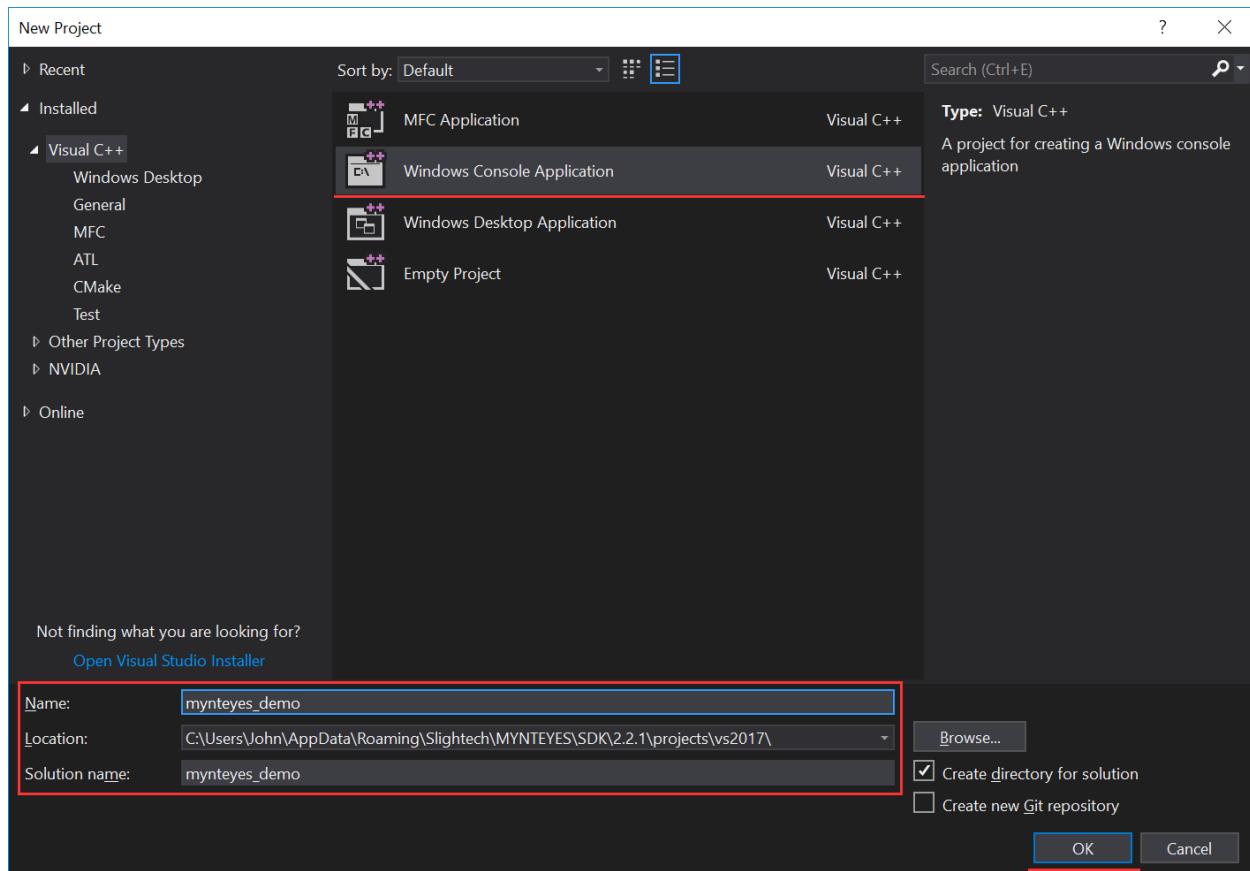
- Windows: 安装 SDK 的 exe 包

### 创建项目

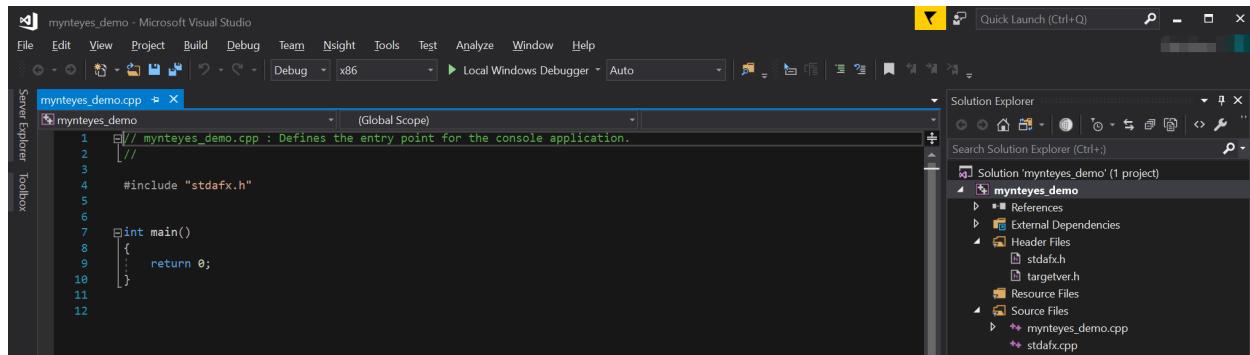
打开 Visual Studio 2017 , 然后 File > New > Project ,



选择 “Windows Console Application”，设置项目位置和名字，点击 “OK”，

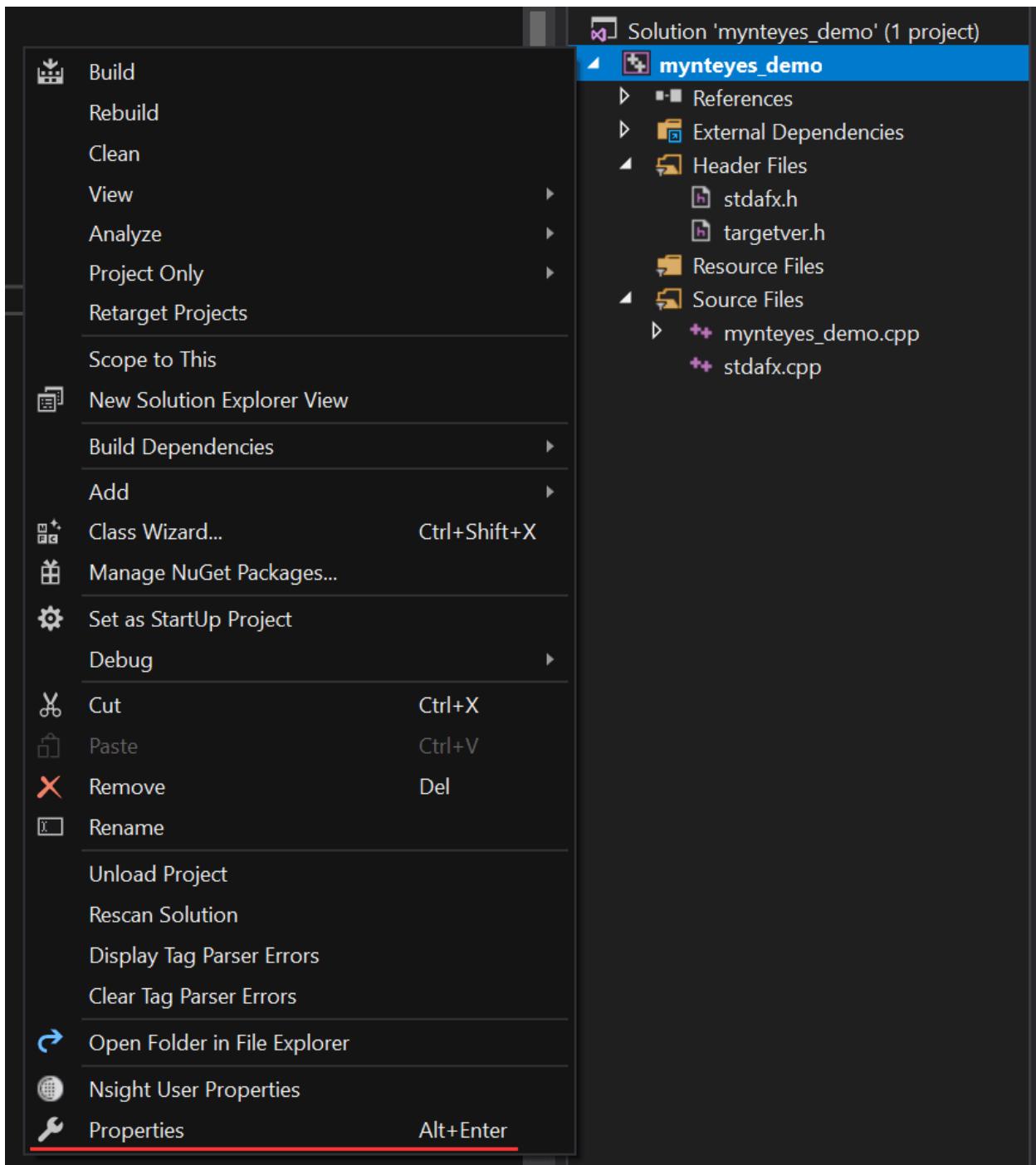


最后，你可以看到一个新的项目被创建，



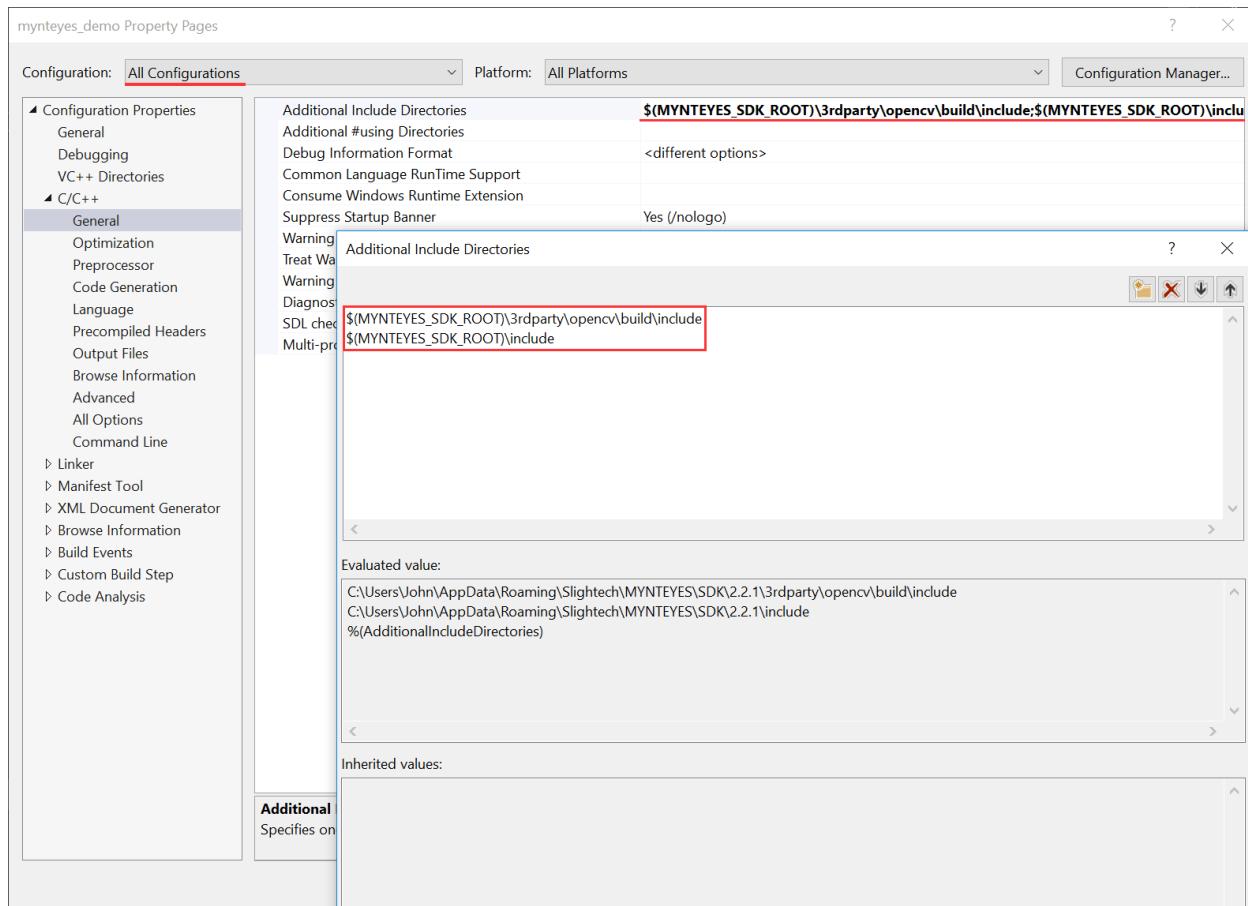
## 配置项目

右键点击该项目，打开“Properties”窗口，



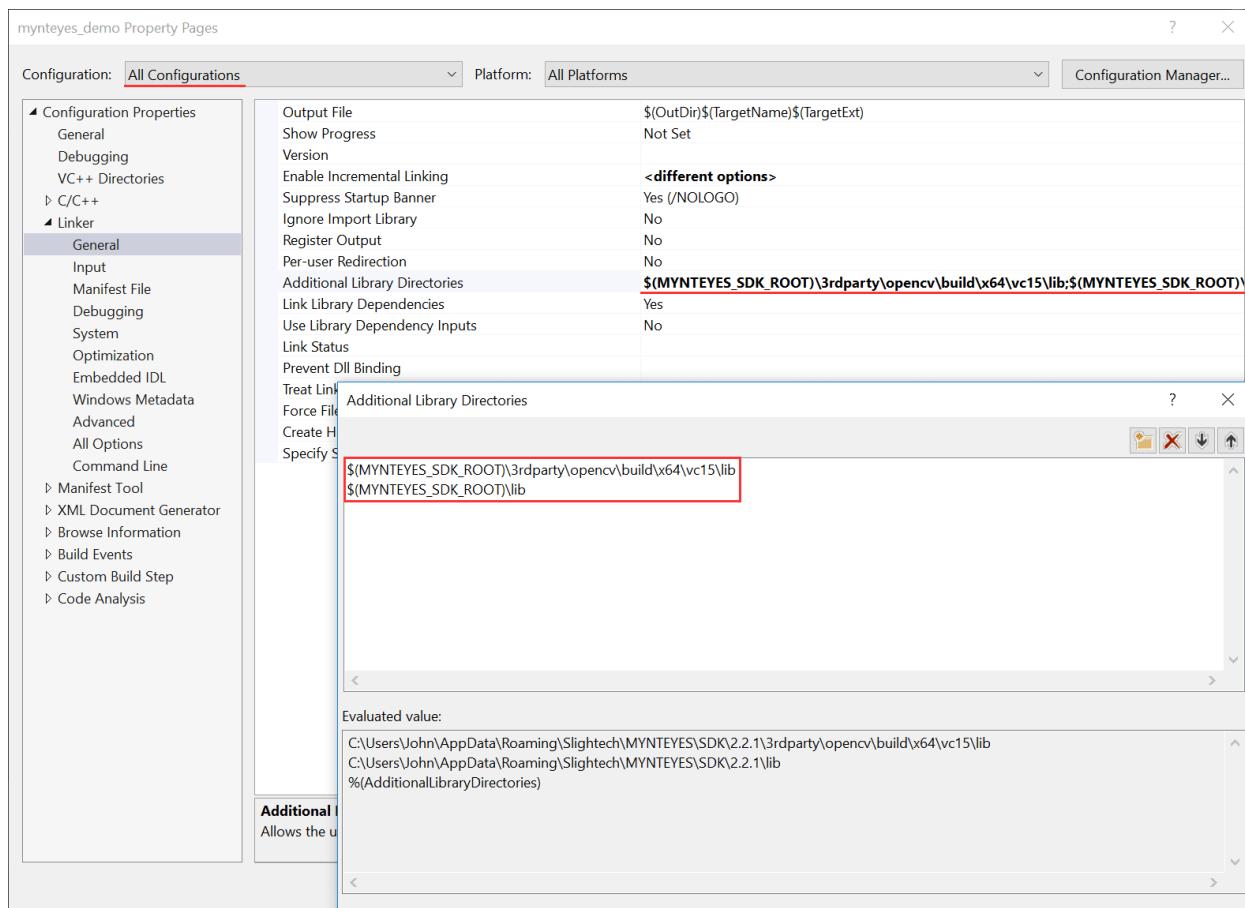
将“Configuration”更改为“All Configurations”，然后添加以下路径到“Additional Include Directories”，

```
$ (MYNTEYES_SDK_ROOT)\include
$ (MYNTEYES_SDK_ROOT)\3rdparty\opencv\build\include
```



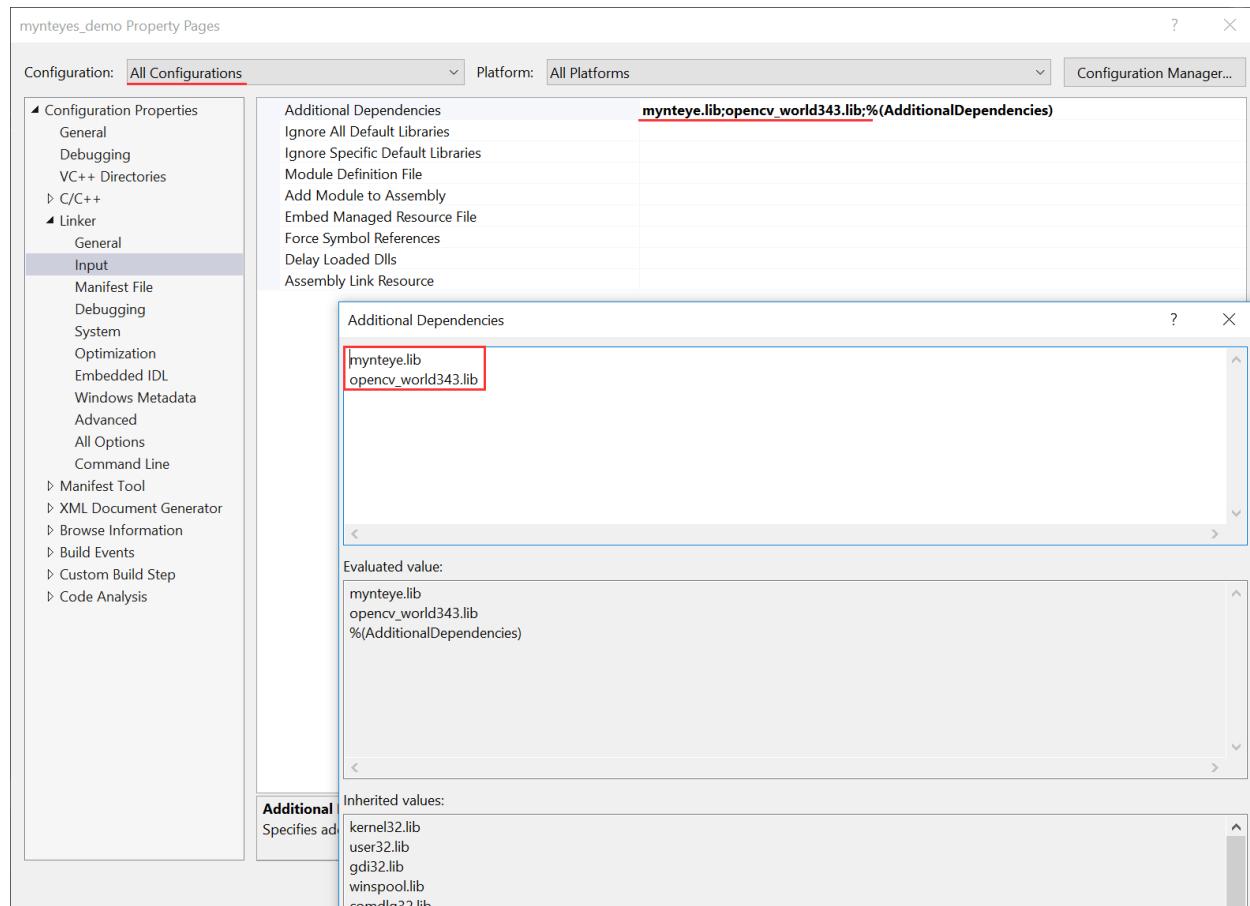
添加以下路径到“Additional Library Directories”，

```
$(MYNTEYES_SDK_ROOT)\1ib
$(MYNTEYES_SDK_ROOT)\3rdparty\opencv\build\x64\vc15\1ib
```



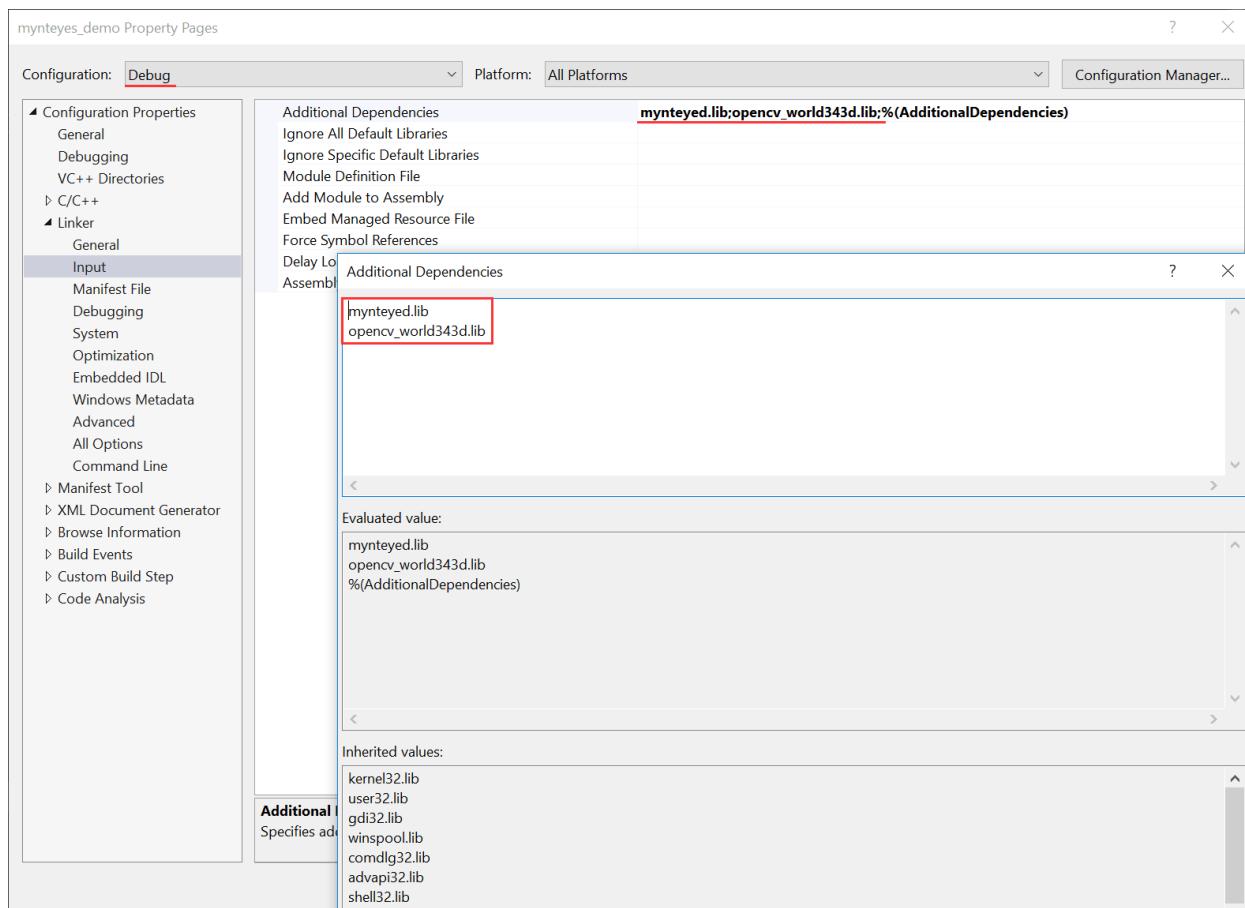
添加以下库到“Additional Dependencies”，

```
mynteye.lib
opencv_world343.lib
```



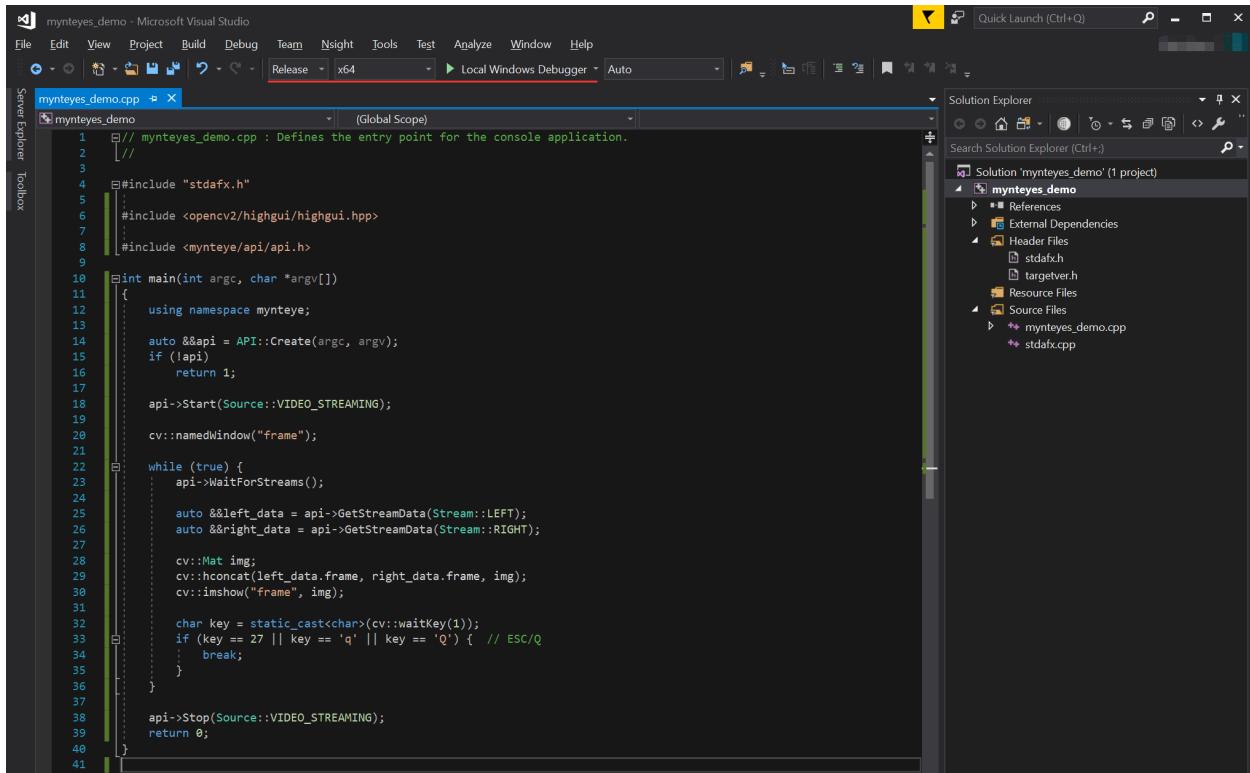
如果需要使用 debug 模式，可以改变“Configuration”到“Debug”添加下列 debug 的库：

```
mynteyed.lib  
opencv_world343d.lib
```

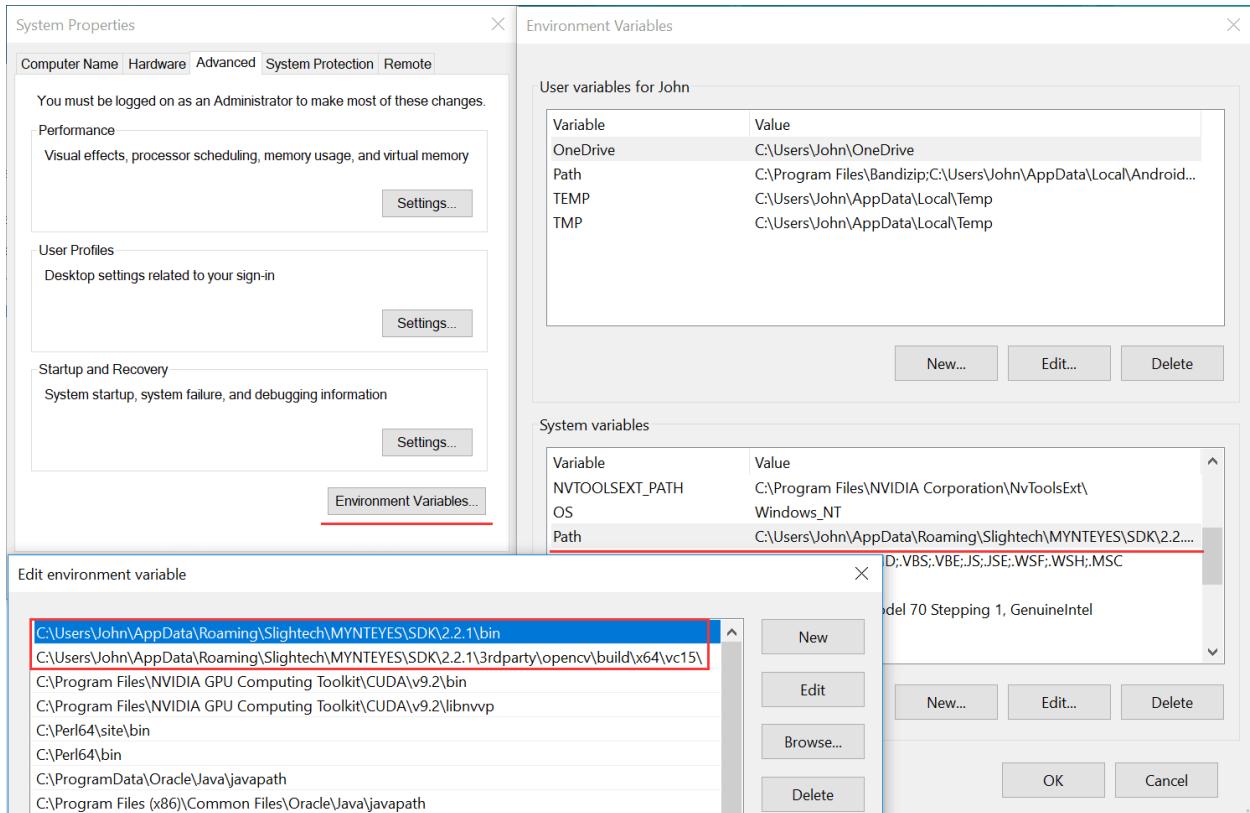


## 使用 SDK

添加头文件和使用 API ,



选择“Release x64”或者“Debug x64”来运行项目。



## 2.7 更新日志

### 2.7.1 2019-07-03(v2.3.9)

- 1、修复 ROS 时间戳换算
- 2、改进文档结构和内容

### 2.7.2 2019-05-20(v2.3.8)

- 1、改进对 VINS-Fusion 开源项目的支持
- 2、改进对 VINS-MONO 开源项目的支持
- 3、修复左右目纠正图像对应错误的问题

### 2.7.3 2019-04-19(2.3.7)

- 1、改进对 VINS-Fusion 开源项目的支持
- 2、改进对 ORB-SLAM2 开源项目的支持

### 2.7.4 2019-04-15(2.3.6)

- 1、修复 ROS 里 imu 对齐的问题
- 2、修复 Ubuntu14.04 的编译问题
- 3、增加 S2100 对 IIC 地址设置的支持

### 2.7.5 2019-04-01(2.3.5)

- 1、完善 camera info 信息
- 2、算法参数通过 yaml 文件输入
- 3、增加 ROS 同时打开多设备样例
- 4、增加 S210A IIC 地址设置 API
- 5、增加图像/imu 数据外部时间源标志位
- 6、提供适配 S1030 的 LaserScan 样例
- 7、修改 ros 点云默认朝向

### 2.7.6 2019-03-18(2.3.4)

- 1、增加获取协处理器芯片及 ISP 版本 API (依赖 S2100/S210A 1.1 固件 & 1.0 辅助芯片固件)。
- 2、修复算法点云残影问题。
- 3、增加对 S1030 376\*240 分辨率的支持 (依赖 S1030 固件版本 2.4.0)。
- 4、增加对 S2100/S210A imu 数据温漂等处理接口 (依赖对 imu 的标定)。

- 5、增加版本对比功能。
- 6、修复使用插件，订阅深度图崩溃问题。
- 7、文档优化。

# CHAPTER 3

## 固件

### 3.1 固件说明

#### 3.1.1 固件与 SDK 适配性

S1030 Firmwares	SDK Version
MYNTEYE_S_2.0.0_rc.img	2.0.0-rc (2.0.0-rc ~ 2.0.0-rc2)
MYNTEYE_S_2.0.0_rc2.img	2.0.0-rc2 (2.0.0-rc ~ 2.0.0-rc2)
MYNTEYE_S_2.0.0_rc1.img	2.0.0-rc1
MYNTEYE_S_2.0.0_rc0.img	2.0.0-rc0 (2.0.0-rc1 ~ 2.0.0-alpha1)
MYNTEYE_S_2.0.0_alpha1.1.img	2.0.0-alpha1 (2.0.0-rc1 ~ 2.0.0-alpha1)
MYNTEYE_S_2.0.0_alpha1.0.img	2.0.0-alpha1 (2.0.0-rc1 ~ 2.0.0-alpha1)
MYNTEYE_S_2.0.0_alpha0.img	2.0.0-alpha0
MYNTEYE_S_2.2.2.img	2.3.0 (2.2.2-rc1 ~ 2.3.0)
MYNTEYE_S_2.3.0.img	2.3.0 (2.2.2-rc1 ~ 2.3.3)
MYNTEYE_S_2.4.0.img	2.3.4 (2.3.4 ~ latest)

S2100 Firmwares	SDK Version
MYNTEYE_S2100_1.1.img	2.3.4
MYNTEYE_S2100_1.2.img	2.3.5(2.3.5 ~ latest)

**注意：**请先确认相机型号，然后使用对应的固件。

Firmwares 表明固件文件名称。其在 [MYNTEYE\\_BOX](#)(点此下载) 的 Firmwares 目录内。

SDK Version 表明该固件适配的 SDK 版本，括号内指可用版本范围。

## 3.2 固件升级

### 3.2.1 升级主处理芯片固件

主处理芯片固件升级，需要使用我们提供的固件升级程序：MYNT EYE TOOL。

固件及 MYNT EYE TOOL 的安装包，都在 [MYNTEYE\\_BOX](#)(点此下载) 的 Firmwares 目录内。文件结构如下：

```
Firmwares/
├─Checksum.txt          # File checksum
├─MYNTEYE_S_2.4.0.img    # S1030 firmware
├─MYNTEYE_S2100_1.2.img   # S2100 firmware
├─...
└─setup.zip              # MYNT EYE TOOL zip
```

固件升级程序，目前仅支持 Windows，所以需要你在 Windows 下进行操作。步骤如下：

#### 下载准备

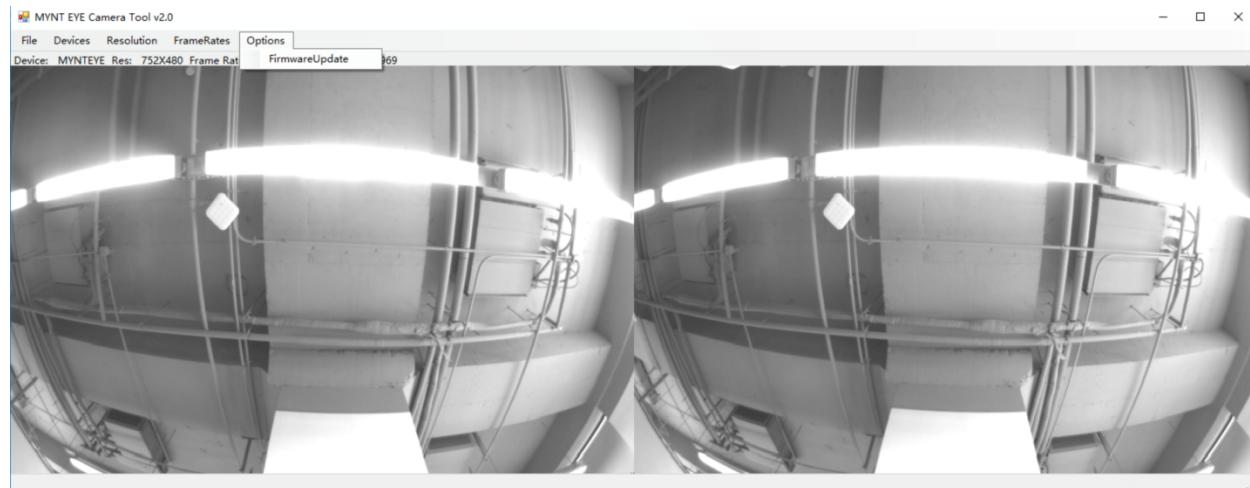
- 下载并解压 setup.zip。
- 找到固件，如 MYNTEYE\_S\_2.4.0.img 。
  - 请见 [固件与 SDK 适配性](#) 选择适合当前 SDK 版本的固件。

#### 安装 MYNT EYE TOOL

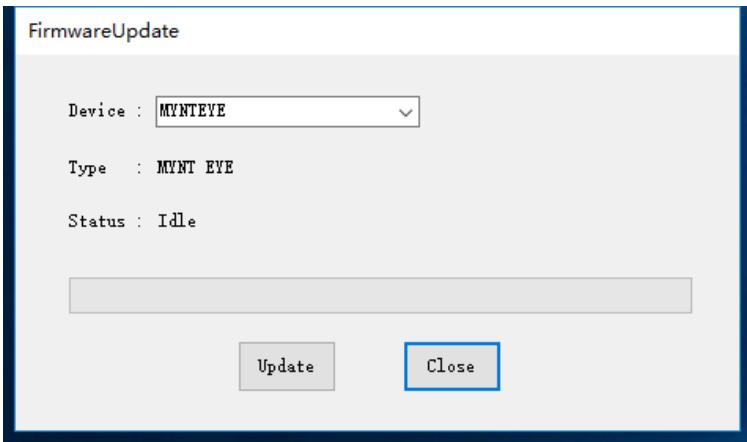
- 双击 setup.msi 安装固件升级程序。

#### 升级固件

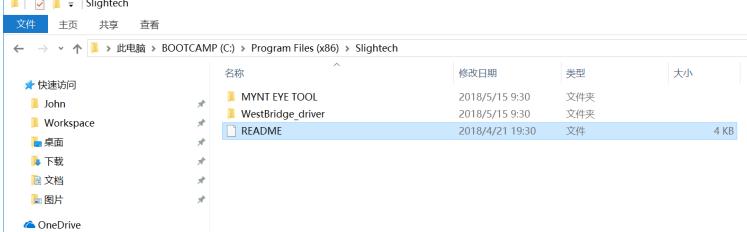
- USB3.0 口插上 MYNT® EYE 设备。
- 打开 MYNT EYE TOOL，选择 Options/FirmwareUpdate。



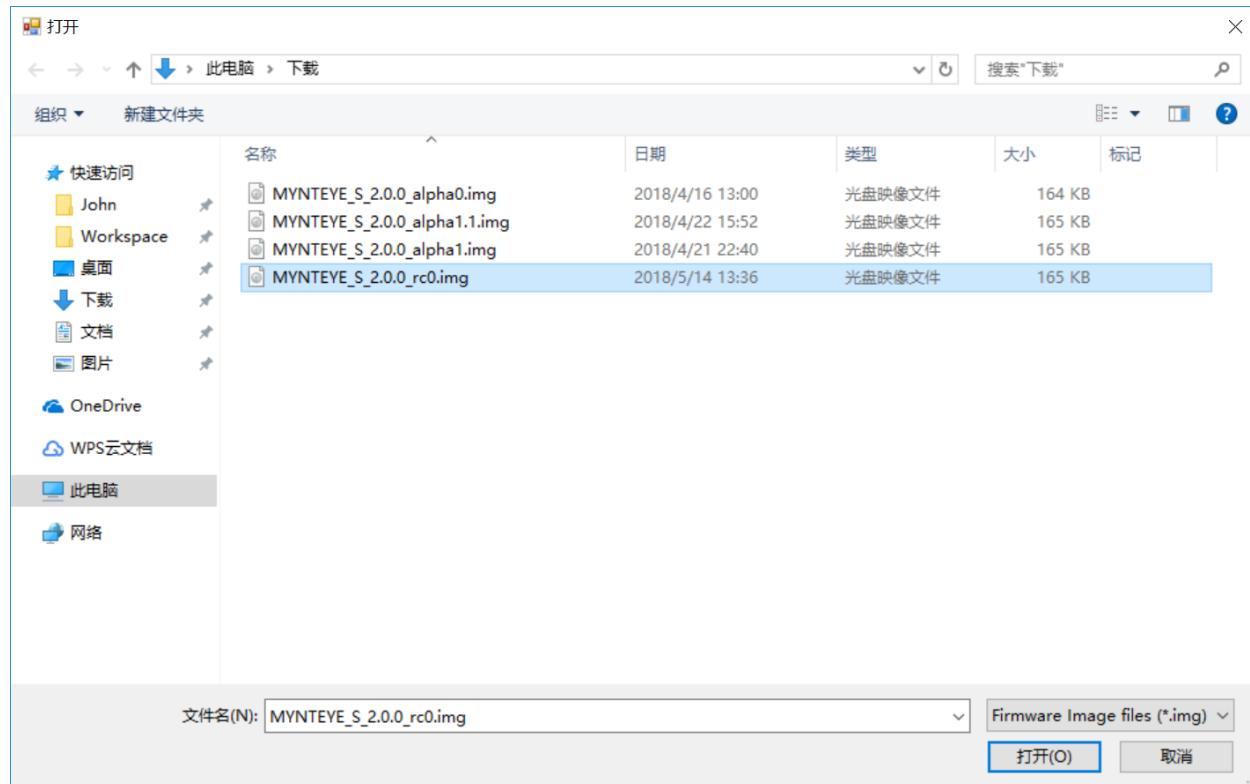
- 点击 Update。



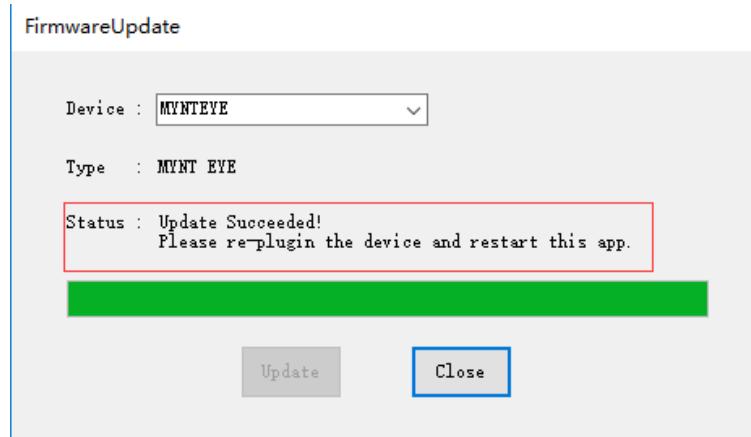
- 弹出警告对话框，直接确定即可。
  - 由于该操作会擦除固件，所以弹出警告。详情见 README。
    - \* 通常在升级过程中，MYNT EYE TOOL 会自动安装驱动。
    - \* 如果升级遇到问题，参考 README 解决。



- 在打开的文件选择框里，选择要升级的固件，开始升级。



- 升级完成后，状态变为 Succeeded。



- 关闭 MYNT EYE TOOL，结束。

**注意：**如果在设备管理器中同时找不到 MYNT 图像设备、WestBridge\_driver 以及 Cypress USB BootLoader 则尝试换一台电脑执行以上操作。如果还是不能升级成功，请及时联系我们。

### 手动更新驱动

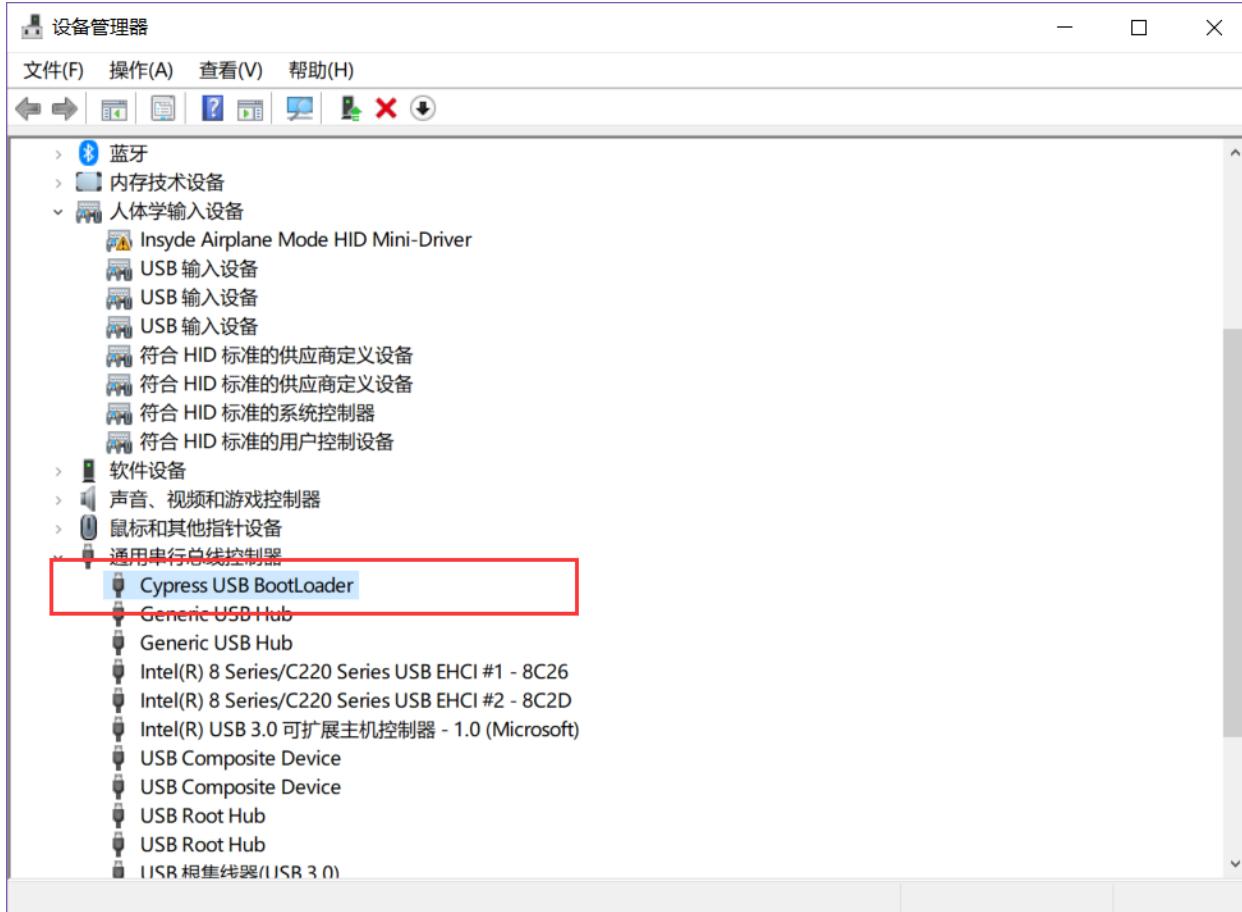
- 如果应用提示您升级失败，则可能是自动安装驱动失败，您可以尝试手动安装驱动然后重新升级。以下为手动安装驱动的步骤。

- 打开设备管理器，找到 WestBridge\_driver 设备，然后右键更新驱动，选择 [应用安装目录]\WestBridge\_driver\[对应系统文件夹](win7 以上选择 wlh)\[系统对应位数]。

## ✓ 其他设备

### WestBridge

- 以 win 10 64 位默认安装路径为例，需要选择的文件夹为 C:\Program Files (x86)\slightech\MYNT EYE TOOL 2.0\WestBridge\_driver\wlh\x64。
- 安装驱动成功之后，可以在设备管理器中找到 Cypress USB BootLoader 设备。



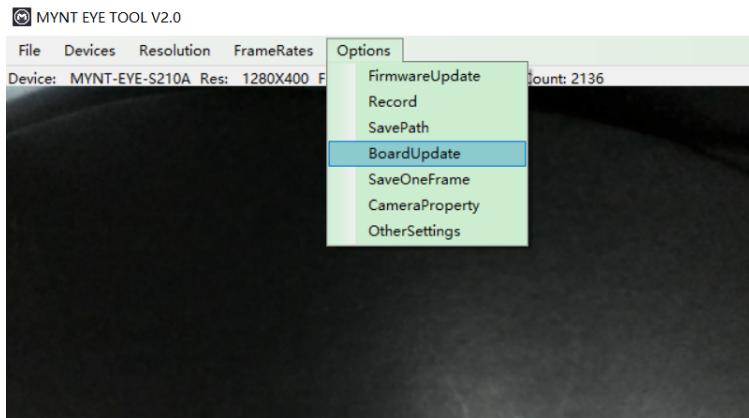
- 然后拔插摄像头，再次打开该应用进行升级。

**警告：**固件升级后，初次打开 MYNT® EYE 设备时，请静置 3 秒，其会有一个零漂补偿过程。或者，请主动调用控制接口 RunOptionAction (Option::ZERO\_DRIFT\_CALIBRATION) 来进行零漂补偿。

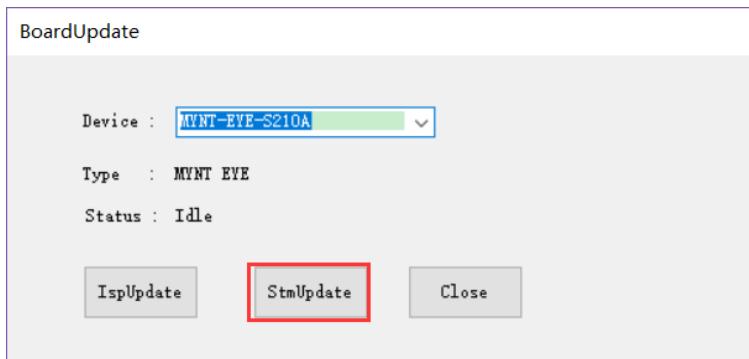
## 3.2.2 升级协处理芯片固件

### 升级协处理芯片 (仅支持 S2100/S210A)

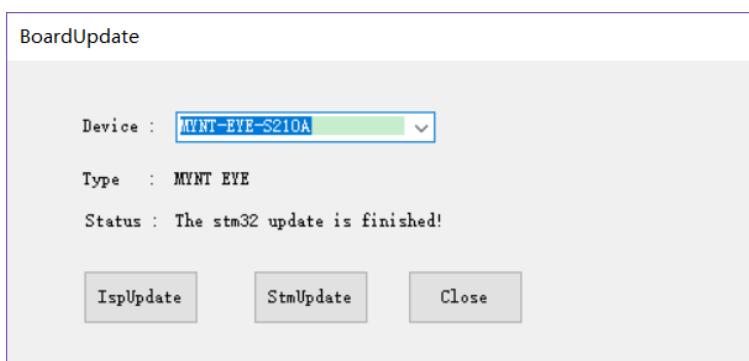
- USB3.0 口插上 MYNT® EYE 设备。
- 打开 MYNT EYE TOOL，选择 Options/BoardUpdate。



- 点击 StmUpdate .



- 在打开的文件选择框里，选择要升级的固件 MYNTEYE-S210x-auxiliary-chip-v1.3.bin，开始升级。
- 升级完成后，会显示升级结束.



# CHAPTER 4

---

## 使用工具支持

---

### 4.1 标定工具的使用

#### 4.1.1 介绍

#### 4.1.2 1.1 支持平台

目前标定工具只支持了 Ubuntu 16.04 LTS 上的发布。但有支持官方、ROS 多版 OpenCV 依赖。

平台	架构	不同依赖
Ubuntu 16.04 LTS	x64(amd64)	libopencv-dev
Ubuntu 16.04 LTS	x64(amd64)	ros-kinetic-opencv3

#### 4.1.3 1.2 工具包说明

Ubuntu 上提供的是 deb/ppa 安装包，名称上会区分架构、依赖和版本。如下：

- mynteye-s-calibrator-opencv-official-1.0.0\_amd64.deb
- mynteye-s-calibrator-opencv-ros-kinetic-1.0.0\_amd64.deb

其中，

依赖标识	依赖包名	详细说明
opencv-official	libopencv-dev	<a href="https://packages.ubuntu.com/xenial/libopencv-dev">https://packages.ubuntu.com/xenial/libopencv-dev</a>
opencv-ros-kinetic	ros-kinetic-opencv3	<a href="http://wiki.ros.org/opencv3">http://wiki.ros.org/opencv3</a>

#### 4.1.4 1.3 deb 工具包获取

获取方式	获取地址
百度网盘	<a href="https://pan.baidu.com/s/19rW0fPKUlQj6eldZpZFoAA">https://pan.baidu.com/s/19rW0fPKUlQj6eldZpZFoAA</a> 提取码: a6ps
Google Drive	<a href="https://drive.google.com/open?id=1RsV2WEKAsfxbn-Z5nGjk5g3ml1UDEsDc">https://drive.google.com/open?id=1RsV2WEKAsfxbn-Z5nGjk5g3ml1UDEsDc</a>

#### 4.1.5 安装

##### 4.1.6 2.1 安装准备

- Ubuntu 16.04 LTS 环境, x64 架构
- 标定工具的 deb 包, 按需选择 OpenCV 依赖 (PPA 安装不需要此步)

##### 4.1.7 2.2 安装 ppa 包

```
$ sudo add-apt-repository ppa:slichtech/mynt-eye-s-sdk
$ sudo apt-get update
$ sudo apt-get install mynteye-s-calibrator
$ sudo ln -sf /opt/myntai/mynteye-s-calibrator/mynteye-s-calibrator /usr/local/bin/_mynteye-s-calibrator
```

##### 4.1.8 2.3 安装 deb 包

sudo dpkg -i 即可安装 deb 包。如下:

```
$ sudo dpkg -i mynteye-s-calibrator-opencv-official-1.0.0_amd64.deb
...
(Reading database ... 359020 files and directories currently installed.)
Preparing to unpack mynteye-s-calibrator-opencv-official-1.0.0_amd64.deb ...
Unpacking mynteye-s-calibrator (1.0.0) over (1.0.0) ...
Setting up mynteye-s-calibrator (1.0.0) ...
```

如果遇到了依赖包未安装的错误, 例如:

```
$ sudo dpkg -i mynteye-s-calibrator-opencv-official-1.0.0_amd64.deb
Selecting previously unselected package mynteye-s-calibrator.
(Reading database ... 358987 files and directories currently installed.)
Preparing to unpack mynteye-s-calibrator-opencv-official-1.0.0_amd64.deb ...
Unpacking mynteye-s-calibrator (1.0.0) ...
dpkg: dependency problems prevent configuration of mynteye-s-calibrator:
mynteye-s-calibrator depends on libatlas-base-dev; however:
Package libatlas-base-dev is not installed.

dpkg: error processing package mynteye-s-calibrator (--install):
dependency problems - leaving unconfigured
Errors were encountered while processing:
mynteye-s-calibrator
```

可以继续执行 sudo apt-get -f install 完成安装,

```
$ sudo apt-get -f install
Reading package lists... Done
Building dependency tree
Reading state information... Done

Correcting dependencies... Done
The following additional packages will be installed:
libatlas-base-dev
Suggested packages:
libblas-doc liblapack-doc
The following NEW packages will be installed:
libatlas-base-dev
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
1 not fully installed or removed.
Need to get 3,596 kB of archives.
After this operation, 30.8 MB of additional disk space will be used.
Do you want to continue? [Y/n]
Get:1 http://cn.archive.ubuntu.com/ubuntu xenial/universe amd64 libatlas-base-dev...
  ↳amd64 3.10.2-9 [3,596 kB]
Fetched 3,596 kB in 3s (1,013 kB/s)
Selecting previously unselected package libatlas-base-dev.
(Reading database ... 358993 files and directories currently installed.)
Preparing to unpack .../libatlas-base-dev_3.10.2-9_amd64.deb ...
Unpacking libatlas-base-dev (3.10.2-9) ...
Setting up libatlas-base-dev (3.10.2-9) ...
update-alternatives: using /usr/lib/atlas-base/atlas/libblas.so to provide /usr/lib/
  ↳libblas.so (libblas.so) in auto mode
update-alternatives: using /usr/lib/atlas-base/atlas/liblapack.so to provide /usr/lib/
  ↳liblapack.so (liblapack.so) in auto mode
Setting up mynteye-s-calibrator (1.0.0) ...
```

## 4.1.9 使用

### 4.1.10 3.1 使用准备

- MYNT EYE S 相机
- 棋盘格标定板
- 光照均匀的场景

### 4.1.11 3.2 使用命令

- 安装好标定工具后，在终端可直接运行 mynteye-s-calibrator 命令进行标定。-h 可见其选项：

```
$ mynteye-s-calibrator -h
Usage: mynteye-s-calibrator [options]
help: mynteye-s-calibrator -h
calibrate: mynteye-s-calibrator -x 11 -y 7 -s 0.036

Calibrate MYNT EYE S device.
```

参数：

**-h, --help** 显示帮助信息并退出

**-x WIDTH, --width=WIDTH** 棋盘格宽, 默认: 11  
**-y HEIGHT, --height=HEIGHT** 棋盘格高, 默认: 7  
**-s METERS, --square=METERS** 棋盘格格子边长, 默认: 0.036  
**-n NUMBER, --number=NUMBER** 用于标定的图片张数, 默认: 11  
**-p PATH, --path=PATH** 保存结果的文件夹名, 默认: 相机 SN 名

- **-x -y -s** 用于设定标定板的宽、高、格子大小。宽、高分别指棋盘格横向的黑白交叉点数。格子大小, 单位是 m

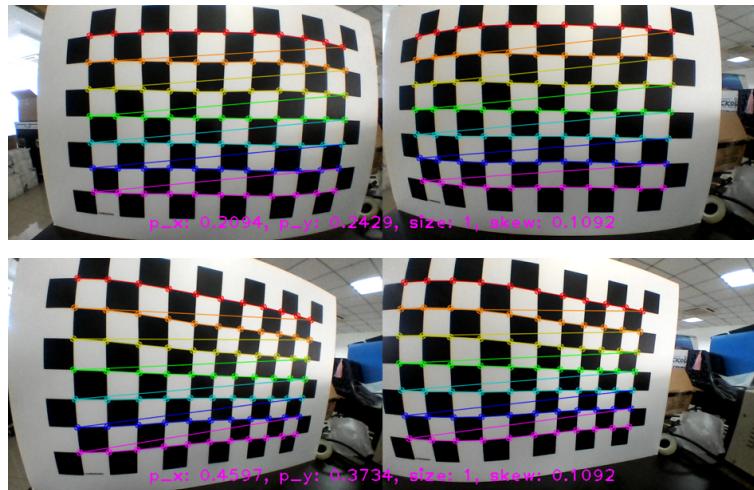
#### 4.1.12 3.3 使用步骤

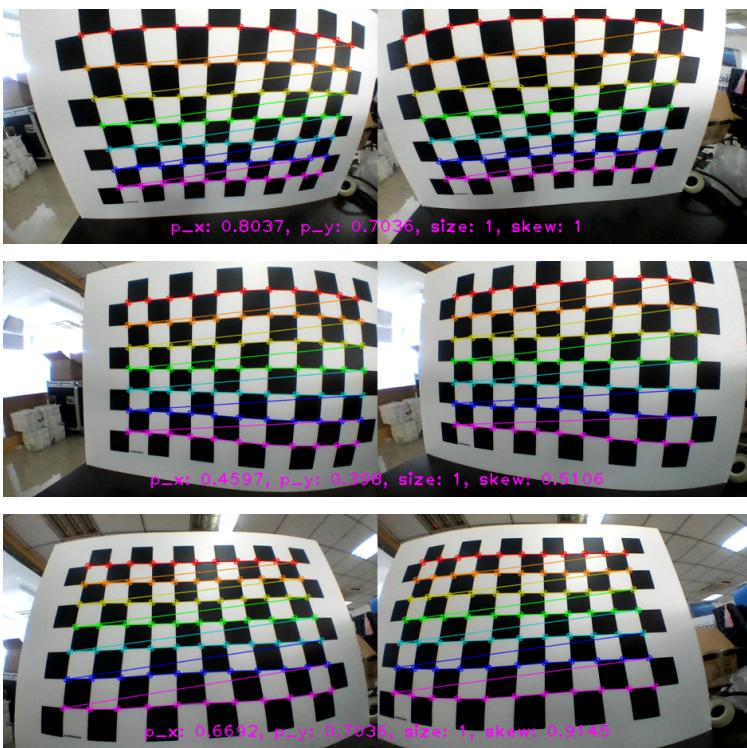
- 首先, 连接好 MYNT EYE S 相机。
- 然后, 终端里运行 mynteye-s-calibrator < 标定板参数 > 命令, 标定板参数需要根据使用的标定板来设置, 参数说明见上

```
john@john-ubuntu:~$ mynteye-s-calibrator
I/main.cc:68 Calibrate MYNT EYE S device with chessboard 11x7, size: 0.036m, num: 11
I/utils.cc:35 Detecting MYNT EYE devices
I/utils.cc:48 MYNT EYE devices:
I/utils.cc:51 index: 0, name: MYNT-EYE-S2100, sn: 030E1D2C0009072B, firmware: 1.2
I/utils.cc:60 Only one MYNT EYE device, select index: 0
I/synthetic.cc:59 camera calib model: kannala_brandt
E/disparity_processor.ccc:51 BM not supported in opencv 2.x, use sgbm
I/utils.cc:93 MYNT EYE requests:
I/utils.cc:96 index: 0, request: width: 1280, height: 400, format: Format::YUVV, fps: 10
I/utils.cc:96 index: 1, request: width: 1280, height: 400, format: Format::YUVV, fps: 20
I/utils.cc:96 index: 2, request: width: 1280, height: 400, format: Format::YUVV, fps: 30
I/utils.cc:96 index: 3, request: width: 1280, height: 400, format: Format::YUVV, fps: 60
I/utils.cc:96 index: 4, request: width: 2560, height: 800, format: Format::YUVV, fps: 10
I/utils.cc:96 index: 5, request: width: 2560, height: 800, format: Format::YUVV, fps: 20
I/utils.cc:96 index: 6, request: width: 2560, height: 800, format: Format::YUVV, fps: 30
I/utils.cc:107 There are 7 stream requests, select index:
2
I/calibrator.cc:76 Had selected 1 imgs
I/calibrator.cc:76 Had selected 2 imgs
```

- 按提示选择相机某个分辨率的 index , 进行此分辨率下的图像标定。
- S1030 相机只需要标定 752\*480 分辨率。S2100 相机需要标定 2560\*800,1280\*400 两个分辨率。
- 标定时尽量让标定板铺满相机左右目图像, 且照顾到四周 (畸变最大)。标定工具会自动评估出合格的图像用于标定计算, 在终端上会提示已选中了多少张。

参考的采集图像, 如下:





- 注：p\_x, p\_y, size, skew 分别表示采集到图像时，标定板于 x 轴、y 轴、缩放、倾斜的比例。作一点参考用。
- 一旦达到标定需求采集的图像数目后，就会进行标定计算、输出结果。如下：

```
[stereo] INFO: Final extrinsics:
r: 0.001 p: -0.020 yaw: -0.000
x: -0.080 y: -0.000 z: -0.001
[Left] INFO: Final reprojection error: 0.201 pixels
[Left] INFO: Camera Parameters:
model_type KANNALA_BRANDT
camera_name left
image_width 640
image_height 400
Projection Parameters
k2 0.50591359548087900
k3 0.40223915642684421
k4 -0.83080480547133262
k5 0.35678495671651012
mu 196.73778341765487046
mv 196.86201619192507678
u0 314.09848864933849200
v0 208.44004910270189157

[right] INFO: Final reprojection error: 0.199 pixels
[right] INFO: Camera Parameters:
model_type KANNALA_BRANDT
camera_name right
image_width 640
image_height 400
Projection Parameters
k2 0.52895559600607733
k3 0.30569625497761727
k4 -0.68909502929151389
k5 0.29354415946915002
mu 196.75176710382709189
mv 196.73648511086881285
u0 317.76351593043682215
v0 196.07326925100898052

I/calibrator.cc:91 Complete rectify
I/calibrator.cc:93 Calibration took a total time of 0.483 sec
I/calibrator.cc:96 Wrote calibration files to SN030E1D2C0009072B-190603101122
Write the image params to device? [Y/n] ■
```

- 1. 终端会打印出左右目的标定结果
- 2. 标定结果会写进 SNXXX 目录的文件中

- a) camera\_left.yaml: 左目参数
- b) camera\_right.yaml: 右目参数
- c) extrinsics.yaml: 双目外参
- d) img.params.equidistant: 相机参数, 可用于 S SDK 写入
- e) stereo\_reprojection\_error.yaml: 重投影误差

- 最后, 还会询问是否写入相机设备。回车或 ‘y’ 即表示确认,

```
Write the image params to device? [Y/n]
1/device_writer.cc:69 Write img params success
1/device_writer.cc:71 Resolution: {width: 640, height: 400}
1/device_writer.cc:73 Intrinsics left: {equidistant, width: 640, height: 400, k2: 0.50591359548087900, k3: 0.40223915642684421, k4: -0.83080480547133262, k5: 0.35678495671651012, mu: 196.73770341765487846, mv: 196.8020161919258678, u0: 314.99848864933849200, v0: 208.44604910270189157}
1/device_writer.cc:73 Intrinsics right: {equidistant, width: 640, height: 400, k2: 0.52895559606067733, k3: 0.3056925497761721, k4: -0.68909580292915130, k5: 0.29354415946158002, mu: 196.75176710382709189, mv: 196.753648531086681285, u0: 317.0351592043682215, v0: 196.07326925108080052}
1/device_writer.cc:75 Extrinsics right to left: {rotation: [0.999790440847159962, 0.00043455861950726, -0.02946866589862620, -0.00045902396407190, 0.99999918591344961, -0.00119057525526974, 0.02046813180001524, 0.00119972131941748, 0.99978978602850144], translation: [-80.14418563298774245, -0.08203749911335115, -0.83212568292361200]}
1/device_writer.cc:71 Resolution: {width: 1280, height: 800}
1/device_writer.cc:73 Intrinsics left: {equidistant, width: 1280, height: 800, k2: 0.48300291178331717, k3: 0.5405613580564269, k4: -1.000000000000000149306269679, k5: 0.50804676388658005, mu: 400.27710465370626025, mv: 400.199667840398379, u0: 635.35774529938328214, v0: 391.87026421600774029}
1/device_writer.cc:74 Intrinsics right: {equidistant, width: 1280, height: 800, k2: 0.50175285896124067, k3: 0.40439108804698687, k4: -0.83550042281540371, k5: 0.35568657449872198, mu: 400.16057662504761083, mv: 400.03985305464365032, u0: 635.35774529938328214, v0: 391.87026421600774029}
1/device_writer.cc:75 Extrinsics right to left: {rotation: [0.99978063717736054, 0.00022852037457626, -0.029943408585, 0.99977937526112876], translation: [-79.83619672534425149, 0.25641071959367268, -1.484868165425454741]}
Write to device done
```

- 写入设备后, 将提示 “Write to device done”。

#### 4.1.13 3.4 标定结果

标定结果, 要求重投影误差最好能达到 0.2 或更低。如果超过 1, 需要重新标定。

重投影误差, 可见标定完成后的输出 “Final reprojection error: 0.201 pixels”, 或者见标定结果文件 “stereo\_reprojection\_error.yaml”。

## 开源项目支持

### 5.1 VINS-Mono 如何整合

#### 5.1.1 在 MYNT® EYE 上运行 VINS-Mono， 请依照这些步骤：

1. 下载 MYNT-EYE-S-SDK 及安装 mynt\_eye\_ros\_wrapper。
2. 按照一般步骤安装 VINS-Mono 。
3. 运行 mynt\_eye\_ros\_wrapper 和 VINS-Mono 。

#### 5.1.2 快捷安装 ROS Kinetic (若已安装, 请忽略)

```
cd ~  
wget https://raw.githubusercontent.com/oroca/oroca-ros-pkg/master/ros_install.sh && \  
chmod 755 ./ros_install.sh && bash ./ros_install.sh catkin_ws kinetic
```

#### 5.1.3 安装 Docker

```
sudo apt-get update  
sudo apt-get install \  
    apt-transport-https \  
    ca-certificates \  
    curl \  
    gnupg-agent \  
    software-properties-common  
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -  
sudo add-apt-repository \  
    "deb [arch=amd64] https://download.docker.com/linux/ubuntu \  
    $(lsb_release -cs) \  
    $(lsb_release -cs)"
```

(下页继续)

(续上页)

```
stable"
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io
```

然后通过 `sudo usermod -aG docker $YOUR_USER_NAME` 命令将账号加到 docker 组。如果遇到 `Permission denied` 错误，请登出后再重新登录。

### 5.1.4 安装 MYNT-EYE-VINS-Sample

确认 ROS 和 docker 都安装完后，使用下面命令安装 vins

```
git clone -b docker_feat https://github.com/slightech/MYNT-EYE-VINS-Sample.git
cd MYNT-EYE-VINS-Sample/docker
make build
```

编译 docker 推荐 16G 以上内存，或者内存和虚拟内存加起来大于 16G。

(如果安装失败，请尝试换一台系统干净的电脑或者重新安装系统与 ROS)

### 5.1.5 在 MYNT® EYE 上运行 VINS-Mono

1. 运行 mynteye 节点

```
cd (local path of MYNT-EYE-S-SDK)
source ./wrappers/ros-devel/setup.bash
roslaunch mynt_eye_ros_wrapper vins_mono.launch
```

2. 打开另一个命令行运行 vins

```
cd path/to/MYNT-EYE-VINS-Sample/docker
./run.sh mynteye_s.launch
# ./run.sh mynteye_s2100.launch # mono with s2100
```

## 5.2 VINS-Fusion 如何整合

### 5.2.1 在 MYNT® EYE 上运行 VINS-Fusion，请依照这些步骤：

1. 下载 MYNT-EYE-S-SDK 及安装 `mynt_eye_ros_wrapper`。
2. 按照一般步骤安装 VINS-Fusion。
3. 运行 `mynt_eye_ros_wrapper` 和 VINS-Fusion。

### 5.2.2 快捷安装 ROS Kinetic (若已安装，请忽略)

```
cd ~
wget https://raw.githubusercontent.com/oroca/oroca-ros-pkg/master/ros_install.sh && \
chmod 755 ./ros_install.sh && bash ./ros_install.sh catkin_ws kinetic
```

### 5.2.3 安装 Docker

```
sudo apt-get update
sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    gnupg-agent \
    software-properties-common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo add-apt-repository \
    "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
    $(lsb_release -cs) \
    stable"
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io
```

然后通过 `sudo usermod -aG docker $YOUR_USER_NAME` 命令将账号加到 docker 组。如果遇到 `Permission denied` 错误，请登出后再重新登录。

### 5.2.4 安装 MYNT-EYE-VINS-FUSION-Samples

```
git clone https://github.com/slightech/MYNT-EYE-VINS-FUSION-Samples.git
cd MYNT-EYE-VINS-FUSION-Samples/docker
make build
```

编译 docker 推荐 16G 以上内存，或者内存和虚拟内存加起来大于 16G。

(如果安装失败，请尝试换一台系统干净的电脑或者重新安装系统与 ROS)

### 5.2.5 在 MYNT® EYE 上运行 VINS-FUSION

#### 1. 运行 mynteye 节点

```
cd (local path of MYNT-EYE-S-SDK)
source ./wrappers/ros-devel/setup.bash
roslaunch mynt_eye_ros_wrapper vins_fusion.launch
```

#### 2. 打开另一个命令行运行 vins

```
cd path/to/MYNT-EYE-VINS-FUSION-Samples/docker
./run.sh mynteye-s/mynt_stereo_imu_config.yaml # Stereo fusion
# ./run.sh mynteye-s2100/mynt_stereo_config.yaml # Stereo fusion with mynteye-s2100
# ./run.sh mynteye-s2100/mynt_stereo_imu_config.yaml # Stereo+imu fusion with mynteye-
→s2100
```

## 5.3 ORB\_SLAM2 如何整合

### 5.3.1 在 MYNT® EYE 上运行 ORB\_SLAM2， 请依照这些步骤：

1. 下载 MYNT-EYE-S-SDK 及安装。

2. 按照一般步骤安装 ORB\_SLAM2。
3. 在 MYNT® EYE 上运行例子。

### 5.3.2 安装依赖

```
sudo apt-get -y install libglew-dev cmake libgoogle-glog-dev
cd ~
git clone https://github.com/stevenlovegrove/Pangolin.git
cd Pangolin
mkdir build
cd build
cmake ..
cmake --build .
sudo make install
```

### 5.3.3 ROS 下创建双目节点

- 添加 Examples/ROS/ORB\_SLAM2 路径到环境变量 ROS\_PACKAGE\_PATH。打开 .bashrc 文件，在最后添加下面命令行。

```
export ROS_PACKAGE_PATH=${ROS_PACKAGE_PATH}:~/catkin_ws/src/MYNT-EYE-ORB-SLAM2-Sample
```

- 运行脚本 build\_ros.sh :

```
chmod +x build.sh
./build.sh
chmod +x build_ros.sh
./build_ros.sh
```

#### Stereo\_ROS 例子

- 运行 ORB\_SLAM2 Stereo\_ROS 例子

##### 1. 运行 mynteye 节点

```
cd [path of mynteye-s-sdk]
make ros
source ./wrappers/ros-devel/setup.bash
roslaunch mynt_eye_ros_wrapper mynteye.launch
```

##### 2. 打开另一个命令行运行 ORB\_SLAM2

```
rosrun ORB_SLAM2 mynteye_s_stereo ./Vocabulary/ORBvoc.txt ./config/mynteye_s_stereo.
→yaml false /mynteye/left_rect/image_rect /mynteye/right_rect/image_rect
```

## 5.4 OKVIS 如何整合

### 5.4.1 在 MYNT® EYE 上运行 OKVIS， 请依照这些步骤：

1. 下载 MYNT-EYE-S-SDK 并安装。

2. 安装依赖，按照原始 OKVIS 步骤安装 MYNT-EYE-OKVIS-Sample。
3. 更新相机参数到 <OKVIS>/config/config\_mynteye.yaml。
4. 在 MYNT® EYE 上运行 OKVIS。

## 5.4.2 安装 MYNT® EYE OKVIS

首先安装原始 OKVIS 及依赖：

```
sudo apt-get install libgoogle-glog-dev

git clone -b mynteye https://github.com/slightech/MYNT-EYE-OKVIS-Sample.git
cd MYNT-EYE-OKVIS-Sample/
mkdir build && cd build
cmake ..
make
```

## 5.4.3 获取相机校准参数

通过 MYNT-EYE-S-SDK API 的 GetIntrinsics() 函数和 GetExtrinsics() 函数，可以获得当前工作设备的图像校准参数：

```
cd MYNT-EYE-S-SDK
./samples/_output/bin/tutorials/get_img_params
```

这时，可以获得针孔模型下的 distortion\_parameters 和 projection\_parameters 参数，然后在[这里](#)更新。

**小技巧：**获取相机校准参数时可以看到相机模型，如果相机为等距模型不能直接写入参数，需要自己标定针孔模型或者按照[写入图像标定参数](#)写入 SDK 中的针孔模型参数来使用。

```
distortion_coefficients: [coeffs]    # only first four parameters of coeffs need to be filled
focal_length: [fx, fy]
principal_point: [cx, cy]
distortion_type: radial_tangential
```

## 5.4.4 运行 MYNT® EYE OKVIS

在 MYNT-EYE-OKVIS-Sample/build 中运行 okvis\_app\_mynteye\_s：

```
cd MYNT-EYE-OKVIS-Sample/bin
./okvis_app_mynteye_s ../config/config_mynteye_s.yaml
```

## 5.5 VIORB 如何整合

### 5.5.1 在 MYNT® EYE 上运行 VIORB， 请依照这些步骤：

1. 下载 [MYNT-EYE-S-SDK](#)， 安装 mynt\_eye\_ros\_wrapper。
2. 按照一般步骤安装 VIORB。
3. 更新相机参数到 <VIO>/config/mynteye\_s.yaml。
4. 运行 mynt\_eye\_ros\_wrapper 和 VIORB。

### 5.5.2 安装 MYNT-EYE-VIORB-Sample.

```
git clone -b mynteye https://github.com/slightech/MYNT-EYE-VIORB-Sample.git
cd MYNT-EYE-VIORB-Sample
```

添加 Examples/ROS/ORB\_VIO 路径到环境变量 ROS\_PACKAGE\_PATH。打开 .bashrc 文件，在最后添加下面命令行。PATH 为当前 MYNT-EYE-VIORB-Sample. 存放路径:

```
export ROS_PACKAGE_PATH=${ROS_PACKAGE_PATH}:PATH/Examples/ROS/ORB_VIO
```

执行:

```
cd MYNT-EYE-VIORB-Sample
./build.sh
```

### 5.5.3 获取相机校准参数

使用 MYNT® EYE 的左目摄像头和 IMU。通过 [MYNT-EYE-S-SDK API](#) 的 GetIntrinsics() 函数和 GetExtrinsics() 函数，可以获得当前工作设备的图像校准参数：

```
cd MYNT-EYE-S-SDK
./samples/_output/bin/tutorials/get_img_params
```

这时，可以获得针孔模型下的 distortion\_parameters 和 projection\_parameters 参数，然后在 <MYNT-EYE-VIORB-Sample>/config/mynteye\_s.yaml 中更新。

**小技巧：**获取相机校准参数时可以看到相机模型，-如果相机为等距模型不能直接写入参数，需要自己标定针孔模型或者按照[写入图像标定参数](#)写入 SDK 中的针孔模型参数来使用。

### 5.5.4 运行 VIORB 和 mynt\_eye\_ros\_wrapper

1. 运行 mynteye 节点

```
roslaunch mynt_eye_ros_wrapper mynteye.launch
```

2. 打开另一个命令行运行 viorb

```
roslaunch ORB_VIO testmynteye_s.launch
```

最后, `pyplotscripts` 下的脚本会将结果可视化。

## 5.6 Maplab 如何整合



# CHAPTER 6

---

API DOCS

---

## 6.1 API

### 6.1.1 API

#### class API

The `API` class to communicate with MYNT® EYE device.

#### Public Types

```
using stream_callback_t = std::function<void (const api::StreamData &data) >
The api::StreamData callback.

using motion_callback_t = std::function<void (const api::MotionData &data) >
The api::MotionData callback.

using stream_switch_callback_t = std::function<void (const Stream &stream) >
The enable/disable switch callback.
```

#### Public Functions

```
Model GetModel() const
Get the model.
```

```
bool Supports (const Stream &stream) const
Supports the stream or not.
```

```
bool Supports (const Capabilities &capability) const
Supports the capability or not.
```

```
bool Supports (const Option &option) const
Supports the option or not.
```

```
bool Supports (const AddOns &addon) const
    Supports the addon or not.

StreamRequest SelectStreamRequest (bool *ok) const
    Log all stream requests and prompt user to select one.

const std::vector<StreamRequest> &GetStreamRequests (const Capabilities &capability) const
    Get all stream requests of the capability.

void ConfigStreamRequest (const Capabilities &capability, const StreamRequest &request)
    Config the stream request to the capability.

const StreamRequest &GetStreamRequest (const Capabilities &capability) const
    Get the config stream requests of the capability.

const std::vector<StreamRequest> &GetStreamRequests () const
    Get all stream requests of the key stream capability.

void ConfigStreamRequest (const StreamRequest &request)
    Config the stream request to the key stream capability.

const StreamRequest &GetStreamRequest () const
    Get the config stream requests of the key stream capability.

std::shared_ptr<DeviceInfo> GetInfo () const
    Get the device info.

std::string GetInfo (const Info &info) const
    Get the device info.

std::string GetSDKVersion () const
    Get the sdk version.

IntrinsicsPinhole GetIntrinsics (const Stream &stream) const

template<typename T>
T GetIntrinsics (const Stream &stream) const
    Get the intrinsics of stream.

std::shared_ptr<IntrinsicsBase> GetIntrinsicsBase (const Stream &stream) const
    Get the intrinsics base of stream.

Extrinsics GetExtrinsics (const Stream &from, const Stream &to) const
    Get the extrinsics from one stream to another.

MotionIntrinsics GetMotionIntrinsics () const
    Get the intrinsics of motion.

Extrinsics GetMotionExtrinsics (const Stream &from) const
    Get the extrinsics from one stream to motion.

void LogOptionInfos () const
    Log all option infos.

OptionInfo GetOptionInfo (const Option &option) const
    Get the option info.

std::int32_t GetOptionValue (const Option &option) const
    Get the option value.
```

void **SetDisparityComputingMethodType** (**const DisparityComputingMethod &MethodType**)  
 Set the disparity computing method.

void **SetOptionValue** (**const Option &option**, std::int32\_t *value*)  
 Set the option value.

bool **RunOptionAction** (**const Option &option**) **const**  
 Run the option action.

void **SetStreamCallback** (**const Stream &stream**, *stream\_callback\_t callback*)  
 Set the callback of stream.

void **SetMotionCallback** (*motion\_callback\_t callback*)  
 Set the callback of motion.

bool **HasStreamCallback** (**const Stream &stream**) **const**  
 Has the callback of stream.

bool **HasMotionCallback** () **const**  
 Has the callback of motion.

void **Start** (**const Source &source**)  
 Start capturing the source.

void **Stop** (**const Source &source**)  
 Stop capturing the source.

void **WaitForStreams** ()  
 Wait the streams are ready.

void **EnableStreamData** (**const Stream &stream**)  
 Enable the data of stream.

**Note** must enable the stream if it's a synthetic one. This means the stream is not native, the device has the capability to provide this stream, but still support this stream.

void **EnableStreamData** (**const Stream &stream**, *stream\_switch\_callback\_t callback*, bool *try\_tag* = false)  
 Enable the data of stream.

callback function will call before the father processor enable. when *try\_tag* is true, the function will do nothing except callback.

void **DisableStreamData** (**const Stream &stream**)  
 Disable the data of stream.

void **DisableStreamData** (**const Stream &stream**, *stream\_switch\_callback\_t callback*, bool *try\_tag* = false)  
 Disable the data of stream.

callback function will call before the children processor disable. when *try\_tag* is true, the function will do nothing except callback.

api::*StreamData* **GetStreamData** (**const Stream &stream**)  
 Get the latest data of stream.

std::vector<api::*StreamData*> **GetStreamDatas** (**const Stream &stream**)  
 Get the datas of stream.

**Note** default cache 4 datas at most.

```
void EnableMotionDatas (std::size_t max_size = std::numeric_limits<std::size_t>::max())
    Enable cache motion datas.

std::vector<api::MotionData> GetMotionDatas ()
    Get the motion datas.

void EnableTimestampCorrespondence (const Stream &stream, bool keep_accel_then_gyro =
    true)
    Enable motion datas with timestamp correspondence of some stream.

void EnablePlugin (const std::string &path)
    Enable the plugin.
```

## Public Static Functions

```
static std::shared_ptr<API> Create (int argc, char *argv[])
```

Create the *API* instance.

**Return** the *API* instance.

**Note** This will init glog with args and call *device::select()* to select a device.

### Parameters

- *argc*: the arg count.
- *argv*: the arg values.

```
static std::shared_ptr<API> Create (int argc, char *argv[], const std::shared_ptr<Device> &device)
```

Create the *API* instance.

**Return** the *API* instance.

**Note** This will init glog with args.

### Parameters

- *argc*: the arg count.
- *argv*: the arg values.
- *device*: the selected device.

```
static std::shared_ptr<API> Create (const std::shared_ptr<Device> &device)
```

Create the *API* instance.

**Return** the *API* instance.

### Parameters

- *device*: the selected device.

## 6.1.2 api::StreamData

```
struct StreamData
    API stream data.
```

## Public Members

```
std::shared_ptr<ImgData> img  

ImgData.  

cv::Mat frame  

Frame.  

std::shared_ptr<device::Frame> frame_raw  

Raw frame.  

std::uint16_t frame_id  

Frame ID.
```

### 6.1.3 api::MotionData

```
struct MotionData  

API motion data.
```

## Public Members

```
std::shared_ptr<ImuData> imu  

ImuData.
```

## 6.2 Device

### 6.2.1 Device

```
class Device  

The Device class to communicate with MYNT® EYE device.
```

## Public Types

```
using stream_callback_t = device::StreamCallback  

The device::StreamData callback.  

using motion_callback_t = device::MotionCallback  

The device::MotionData callback.
```

## Public Functions

```
Model GetModel() const  

Get the model.  

bool Supports(const Stream &stream) const  

Supports the stream or not.  

bool Supports(const Capabilities &capability) const  

Supports the capability or not.
```

bool **Supports** (*const Option &option*) **const**  
Supports the option or not.

bool **Supports** (*const AddOns &addon*) **const**  
Supports the addon or not.

**const std::vector<StreamRequest> &GetStreamRequests** (*const Capabilities &capability*) **const**  
Get all stream requests of the capability.

**void ConfigStreamRequest** (*const Capabilities &capability, const StreamRequest &request*)  
Config the stream request to the capability.

**const StreamRequest &GetStreamRequest** (*const Capabilities &capability*) **const**  
Get the config stream requests of the capability.

**const std::vector<StreamRequest> &GetStreamRequests()** **const**  
Get all stream requests of the key stream capability.

**void ConfigStreamRequest** (*const StreamRequest &request*)  
Config the stream request to the key stream capability.

**const StreamRequest &GetStreamRequest()** **const**  
Get the config stream requests of the key stream capability.

**std::shared\_ptr<DeviceInfo> GetInfo()** **const**  
Get the device info.

**std::string GetInfo** (*const Info &info*) **const**  
Get the device info of a field.

**std::shared\_ptr<IntrinsicsBase> GetIntrinsics** (*const Stream &stream*) **const**  
Get the intrinsics of stream.

**Extrinsics GetExtrinsics** (*const Stream &from, const Stream &to*) **const**  
Get the extrinsics from one stream to another.

**MotionIntrinsics GetMotionIntrinsics()** **const**  
Get the intrinsics of motion.

**Extrinsics GetMotionExtrinsics** (*const Stream &from*) **const**  
Get the extrinsics from one stream to motion.

**std::shared\_ptr<IntrinsicsBase> GetIntrinsics** (*const Stream &stream, bool \*ok*) **const**  
Get the intrinsics of stream.

**Extrinsics GetExtrinsics** (*const Stream &from, const Stream &to, bool \*ok*) **const**  
Get the extrinsics from one stream to another.

**MotionIntrinsics GetMotionIntrinsics** (*bool \*ok*) **const**  
Get the intrinsics of motion.

**Extrinsics GetMotionExtrinsics** (*const Stream &from, bool \*ok*) **const**  
Get the extrinsics from one stream to motion.

**void SetIntrinsics** (*const Stream &stream, const std::shared\_ptr<IntrinsicsBase> &in*)  
Set the intrinsics of stream.

**void SetExtrinsics** (*const Stream &from, const Stream &to, const Extrinsics &ex*)  
Set the extrinsics from one stream to another.

```
void SetMotionIntrinsics (const MotionIntrinsics &in)
    Set the intrinsics of motion.

void SetMotionExtrinsics (const Stream &from, const Extrinsics &ex)
    Set the extrinsics from one stream to motion.

void LogOptionInfos () const
    Log all option infos.

OptionInfo GetOptionInfo (const Option &option) const
    Get the option info.

std::int32_t GetOptionValue (const Option &option) const
    Get the option value.

void SetOptionValue (const Option &option, std::int32_t value)
    Set the option value.

bool RunOptionAction (const Option &option) const
    Run the option action.

void SetStreamCallback (const Stream &stream, stream_callback_t callback, bool async = false)
    Set the callback of stream.

void SetMotionCallback (motion_callback_t callback, bool async = false)
    Set the callback of motion.

bool HasStreamCallback (const Stream &stream) const
    Has the callback of stream.

bool HasMotionCallback () const
    Has the callback of motion.

virtual void Start (const Source &source)
    Start capturing the source.

virtual void Stop (const Source &source)
    Stop capturing the source.

void WaitForStreams ()
    Wait the streams are ready.

device::StreamData GetStreamData (const Stream &stream)
    Get the latest data of stream.

device::StreamData GetLatestStreamData (const Stream &stream)

std::vector<device::StreamData> GetStreamDatas (const Stream &stream)
    Get the datas of stream.



Note default cache 4 datas at most.



void DisableMotionDatas ()
    Disable cache motion datas.

void EnableMotionDatas ()
    Enable cache motion datas.
```

```
void EnableMotionDatas (std::size_t max_size)
    Enable cache motion datas.

std::vector<device::MotionData> GetMotionDatas ()
    Get the motion datas.
```

### Public Static Functions

```
static std::shared_ptr<Device> Create (const std::string &name, std::shared_ptr<uvc::device> device)
    Create the Device instance.
```

**Return** the *Device* instance.

#### Parameters

- *name*: the device name.
- *device*: the device from uvc.

## 6.2.2 device::Frame

```
class Frame
    Frame with raw data.
```

### Public Functions

```
Frame (const StreamRequest &request, const void *data)
    Construct the frame with StreamRequest and raw data.
```

```
Frame (std::uint16_t width, std::uint16_t height, Format format, const void *data)
    Construct the frame with stream info and raw data.
```

```
std::uint16_t width () const
    Get the width.
```

```
std::uint16_t height () const
    Get the height.
```

```
Format format () const
    Get the format.
```

```
std::uint8_t *data ()
    Get the data.
```

```
const std::uint8_t *data () const
    Get the const data.
```

```
std::size_t size () const
    Get the size of data.
```

```
Frame clone () const
    Clone a new frame.
```

### 6.2.3 device::StreamData

```
struct StreamData
```

*Device* stream data.

#### Public Members

std::shared\_ptr<*ImgData*> **img**  
*ImgData*.

std::shared\_ptr<*Frame*> **frame**  
*Frame*.

std::uint16\_t **frame\_id**  
*Frame* ID.

### 6.2.4 device::MotionData

```
struct MotionData
```

*Device* motion data.

#### Public Members

std::shared\_ptr<*ImuData*> **imu**  
*ImuData*.

## 6.3 Enums

### 6.3.1 Model

```
enum mynteye::Model
```

*Device* model.

*Values:*

**STANDARD**  
 Standard.

**STANDARD2**  
 Standard 2.

**STANDARD210A**  
 Standard 210a.

### 6.3.2 Stream

```
enum mynteye::Stream
```

Streams define different type of data.

*Values:*

**LEFT**

Left stream.

**RIGHT**

Right stream.

**LEFT\_RECTIFIED**

Left stream, rectified.

**RIGHT\_RECTIFIED**

Right stream, rectified.

**DISPARITY**

Disparity stream.

**DISPARITY\_NORMALIZED**

Disparity stream, normalized.

**DEPTH**

Depth stream.

**POINTS**

Point cloud stream.

### 6.3.3 Capabilities

**enum mynteye::Capabilities**

Capabilities define the full set of functionality that the device might provide.

*Values:*

**STEREO**

Provides stereo stream.

**STEREO\_COLOR**

Provide stereo color stream.

**COLOR**

Provides color stream.

**DEPTH**

Provides depth stream.

**POINTS**

Provides point cloud stream.

**FISHEYE**

Provides fisheye stream.

**INFRARED**

Provides infrared stream.

**INFRARED2**

Provides second infrared stream.

**IMU**

Provides IMU (accelerometer, gyroscope) data.

### 6.3.4 Info

**enum mynteye::Info**

Camera info fields are read-only strings that can be queried from the device.

*Values:*

**DEVICE\_NAME**

*Device* name.

**SERIAL\_NUMBER**

Serial number.

**FIRMWARE\_VERSION**

Firmware version.

**HARDWARE\_VERSION**

Hardware version.

**SPEC\_VERSION**

Spec version.

**LENS\_TYPE**

Lens type.

**IMU\_TYPE**

IMU type.

**NOMINAL\_BASELINE**

Nominal baseline.

**AUXILIARY\_CHIP\_VERSION**

Auxiliary chip version.

**ISP\_VERSION**

Isp version.

### 6.3.5 Option

**enum mynteye::Option**

Camera control options define general configuration controls.

*Values:*

**GAIN**

Image gain, valid if manual-exposure.

range: [0,48], default: 24

**BRIGHTNESS**

Image brightness, valid if manual-exposure.

range: [0,240], default: 120

**CONTRAST**

Image contrast, valid if manual-exposure.

range: [0,255], default: 127

**FRAME\_RATE**

Image frame rate, must set IMU\_FREQUENCY together.

values: {10,15,20,25,30,35,40,45,50,55,60}, default: 25

**IMU\_FREQUENCY**

IMU frequency, must set FRAME\_RATE together.

values: {100,200,250,333,500}, default: 200

**EXPOSURE\_MODE**

Exposure mode.

0: enable auto-exposure 1: disable auto-exposure (manual-exposure)

**MAX\_GAIN**

Max gain, valid if auto-exposure.

range of standard 1: [0,48], default: 48 range of standard 2: [0,255], default: 8

**MAX\_EXPOSURE\_TIME**

Max exposure time, valid if auto-exposure.

range of standard 1: [0,240], default: 240 range of standard 2: [0,1000], default: 333

**MIN\_EXPOSURE\_TIME**

min exposure time, valid if auto-exposure

range: [0,1000], default: 0

**DESIRED\_BRIGHTNESS**

Desired brightness, valid if auto-exposure.

range of standard 1: [0,255], default: 192 range of standard 2: [1,255], default: 122

**IR\_CONTROL**

IR control.

range: [0,160], default: 0

**HDR\_MODE**

HDR mode.

0: 10-bit 1: 12-bit

**ACCELEROMETER\_RANGE**

The range of accelerometer.

value of standard 1: {4,8,16,32}, default: 8 value of standard 2: {6,12,24,48}, default: 12

**GYROSCOPE\_RANGE**

The range of gyroscope.

value of standard 1: {500,1000,2000,4000}, default: 1000 value of standard 2: {250,500,1000,2000,4000}, default: 1000

**ACCELEROMETER\_LOW\_PASS\_FILTER**

The parameter of accelerometer low pass filter.

values: {0,1,2}, default: 2

**GYROSCOPE\_LOW\_PASS\_FILTER**

The parameter of gyroscope low pass filter.

values: {23,64}, default: 64

**ZERO\_DRIFT\_CALIBRATION**

Zero drift calibration.

**ERASE\_CHIP**

Erase chip.

### 6.3.6 Source

**enum mynteye::Source**

Source allows the user to choose which data to be captured.

*Values:*

**VIDEO\_STREAMING**

Video streaming of stereo, color, depth, etc.

**MOTION\_TRACKING**

Motion tracking of IMU (accelerometer, gyroscope)

**ALL**

Enable everything together.

### 6.3.7 AddOns

**enum mynteye::AddOns**

Add-Ons are peripheral modules of our hardware.

*Values:*

**INFRARED**

Infrared.

**INFRARED2**

Second infrared.

### 6.3.8 Format

**enum mynteye::Format**

Formats define how each stream can be encoded.

*Values:*

**GREY** = ((std::uint32\_t)('G') | ((std::uint32\_t)('R') << 8) | ((std::uint32\_t)('E') << 16) | ((std::uint32\_t)('Y') << 24))  
Greyscale, 8 bits per pixel.

**YUYV** = ((std::uint32\_t)('Y') | ((std::uint32\_t)('U') << 8) | ((std::uint32\_t)('Y') << 16) | ((std::uint32\_t)('V') << 24))  
YUV 4:2:2, 16 bits per pixel.

**BGR888** = ((std::uint32\_t)('B') | ((std::uint32\_t)('G') << 8) | ((std::uint32\_t)('R') << 16) | ((std::uint32\_t)('3') << 24))  
BGR 8:8:8, 24 bits per pixel.

**RGB888** = ((std::uint32\_t)('R') | ((std::uint32\_t)('G') << 8) | ((std::uint32\_t)('B') << 16) | ((std::uint32\_t)('3') << 24))  
RGB 8:8:8, 24 bits per pixel.

### 6.3.9 CalibrationModel

**enum mynteye::CalibrationModel**

Camera calibration model.

*Values:*

**PINHOLE** = 0

Pinhole.

**KANNALA\_BRANDT** = 1  
Equidistant: KANNALA\_BRANDT.

**UNKNOW**  
Unknow.

### 6.3.10 DisparityComputingMethod

**enum mynteye::DisparityComputingMethod**  
Camera disparity computing method type.

*Values:*

**SGBM** = 0

bm

**BM** = 1

sgbm

**UNKNOW**

unknow

## 6.4 Types

### 6.4.1 OptionInfo

**struct OptionInfo**  
Option info.

#### Public Members

**std::int32\_t min**  
Minimum value.

**std::int32\_t max**  
Maximum value.

**std::int32\_t def**  
Default value.

### 6.4.2 Resolution

**struct Resolution**  
*Resolution*.

#### Public Members

**std::uint16\_t width**  
Width.

**std::uint16\_t height**  
Height.

### 6.4.3 StreamRequest

```
struct StreamRequest
```

Stream request.

#### Public Members

**std::uint16\_t width**  
Stream width in pixels.

**std::uint16\_t height**  
Stream height in pixels.

**Format format**  
Stream pixel format.

**std::uint16\_t fps**  
Stream frames per second.

### 6.4.4 Intrinsics

#### IntrinsicsPinhole

```
struct IntrinsicsPinhole : public mynteye::IntrinsicsBase
```

Stream intrinsics (Pinhole)

#### Public Members

**double fx**  
The focal length of the image plane, as a multiple of pixel width.

**double fy**  
The focal length of the image plane, as a multiple of pixel height.

**double cx**  
The horizontal coordinate of the principal point of the image.

**double cy**  
The vertical coordinate of the principal point of the image.

**std::uint8\_t model**  
The distortion model of the image

**double coeffs[5]**  
The distortion coefficients: k1,k2,p1,p2,k3.

#### IntrinsicsEquidistant

```
struct IntrinsicsEquidistant : public mynteye::IntrinsicsBase
```

Stream intrinsics (Equidistant: KANNALA\_BRANDT)

## Public Members

```
double coeffs[8]  
The distortion coefficients: k2,k3,k4,k5,mu,mv,u0,v0.
```

## ImuIntrinsics

### struct ImuIntrinsics

IMU intrinsics: scale, drift and variances.

## Public Members

```
double scale[3][3]  
Scale matrix.
```

Scale X	cross axis	cross axis
cross axis	Scale Y	cross axis
cross axis	cross axis	Scale Z

```
double noise[3]  
Noise density variances.
```

```
double bias[3]  
Random walk variances.
```

## MotionIntrinsics

### struct MotionIntrinsics

Motion intrinsics, including accelerometer and gyroscope.

## Public Members

```
ImuIntrinsics accel  
Accelerometer intrinsics.
```

```
ImuIntrinsics gyro  
Gyroscope intrinsics.
```

## 6.4.5 Extrinsics

### struct Extrinsics

*Extrinsics*, represent how the different datas are connected.

## Public Functions

```
Extrinsics Inverse () const  
Inverse this extrinsics.
```

**Return** the inversed extrinsics.

**Public Members****double `rotation`[3][3]**

Rotation matrix.

**double `translation`[3]**

Translation vector.

## 6.4.6 ImgData

**struct `ImgData`**

Image data.

**Public Members****std::uint16\_t `frame_id`**

Image frame id.

**std::uint64\_t `timestamp`**

Image timestamp in 1us.

**std::uint16\_t `exposure_time`**

Image exposure time, virtual value in [1, 480].

## 6.4.7 ImuData

**struct `ImuData`**

IMU data.

**Public Members****std::uint32\_t `frame_id`**

IMU frame id.

**std::uint8\_t `flag`**

IMU accel or gyro flag.

0: accel and gyro are both valid 1: accel is valid

2: gyro is valid

**std::uint64\_t `timestamp`**

IMU timestamp in 1us.

**double `accel`[3]**

IMU accelerometer data for 3-axis: X, Y, Z.

**double `gyro`[3]**

IMU gyroscope data for 3-axis: X, Y, Z.

**double `temperature`**

IMU temperature.

## 6.5 Utils

### 6.5.1 select

```
std::shared_ptr<Device> mynteye::device::select()  
    Detecting MYNT EYE devices and prompt user to select one.
```

**Return** the selected device, or `nullptr` if none.

### 6.5.2 select\_request

```
MYNTEYE_NAMESPACE::StreamRequest mynteye::device::select_request(const  
    std::shared_ptr<Device>  
    &device,      bool  
    *ok)
```

List stream requests and prompt user to select one.

**Return** the selected request.

### 6.5.3 get\_real\_exposure\_time

```
float mynteye::utils::get_real_exposure_time(std::int32_t frame_rate, std::uint16_t exposure_time)
```

Get real exposure time in ms from virtual value, according to its frame rate.

**Return** the real exposure time in ms, or the virtual value if frame rate is invalid.

#### Parameters

- `frame_rate`: the frame rate of the device.
- `exposure_time`: the virtual exposure time.

### 6.5.4 get\_sdk\_root\_dir

```
std::string mynteye::utils::get_sdk_root_dir()  
    Get sdk root dir.
```

### 6.5.5 get\_sdk\_install\_dir

```
std::string mynteye::utils::get_sdk_install_dir()  
    Get sdk install dir.
```

## 技术支持

---

### 7.1 常见问题

如果遇到相机使用问题, 请先查阅以下文档:

<http://support.myntai.com/hc/>

### 7.2 联系我们

如果无法解决问题, 可以通过客户服务联系我们。

<http://support.myntai.com/hc/request/new/>



## M

mynteye::ACCELEROMETER\_LOW\_PASS\_FILTER  
     (*C++ enumerator*), 94  
 mynteye::ACCELEROMETER\_RANGE (*C++ enumerator*), 94  
 mynteye::AddOns (*C++ enum*), 95  
 mynteye::ALL (*C++ enumerator*), 95  
 mynteye::API (*C++ class*), 83  
 mynteye::API::ConfigStreamRequest (*C++ function*), 84  
 mynteye::API::Create (*C++ function*), 86  
 mynteye::API::DisableStreamData (*C++ function*), 85  
 mynteye::API::EnableMotionDatas (*C++ function*), 86  
 mynteye::API::EnablePlugin (*C++ function*), 86  
 mynteye::API::EnableStreamData (*C++ function*), 85  
 mynteye::API::EnableTimestampCorrespondence  
     (*C++ function*), 86  
 mynteye::API::GetExtrinsics (*C++ function*), 84  
 mynteye::API::GetInfo (*C++ function*), 84  
 mynteye::API::GetIntrinsics (*C++ function*), 84  
 mynteye::API::GetIntrinsicsBase (*C++ function*), 84  
 mynteye::API::GetModel (*C++ function*), 83  
 mynteye::API::GetMotionDatas (*C++ function*), 86  
 mynteye::API::GetMotionExtrinsics (*C++ function*), 84  
 mynteye::API::GetMotionIntrinsics (*C++ function*), 84  
 mynteye::API::GetOptionInfo (*C++ function*), 84  
 mynteye::API::GetOptionValue (*C++ function*), 84  
 mynteye::API::GetSDKVersion (*C++ function*), 84  
 mynteye::API::GetStreamData (*C++ function*), 85  
 mynteye::API::GetStreamDatas (*C++ function*), 85  
 mynteye::API::GetStreamRequest (*C++ function*), 84  
 mynteye::API::GetStreamRequests (*C++ function*), 84  
 mynteye::API::HasMotionCallback (*C++ function*), 85  
 mynteye::API::HasStreamCallback (*C++ function*), 85  
 mynteye::API::LogOptionInfos (*C++ function*), 84  
 mynteye::API::motion\_callback\_t (*C++ type*), 83  
 mynteye::api::MotionData (*C++ class*), 87  
 mynteye::api::MotionData::imu (*C++ member*), 87  
 mynteye::API::RunOptionAction (*C++ function*), 85  
 mynteye::API::SelectStreamRequest (*C++ function*), 84  
 mynteye::API::SetDisparityComputingMethodType  
     (*C++ function*), 85  
 mynteye::API::SetMotionCallback (*C++ function*), 85  
 mynteye::API::SetOptionValue (*C++ function*), 85  
 mynteye::API::SetStreamCallback (*C++ function*), 85  
 mynteye::API::Start (*C++ function*), 85  
 mynteye::API::Stop (*C++ function*), 85  
 mynteye::API::stream\_callback\_t  
     (*C++ type*), 83  
 mynteye::API::stream\_switch\_callback\_t  
     (*C++ type*), 83  
 mynteye::api::StreamData (*C++ class*), 86

mynteye::api::StreamData::frame (C++ member), 87  
 mynteye::api::StreamData::frame\_id (C++ member), 87  
 mynteye::api::StreamData::frame\_raw (C++ member), 87  
 mynteye::api::StreamData::img (C++ member), 87  
 mynteye::API::Supports (C++ function), 83, 84  
 mynteye::API::WaitForStreams (C++ function), 85  
 mynteye::AUXILIARY\_CHIP\_VERSION (C++ enumerator), 93  
 mynteye::BGR888 (C++ enumerator), 95  
 mynteye::BM (C++ enumerator), 96  
 mynteye::BRIGHTNESS (C++ enumerator), 93  
 mynteye::CalibrationModel (C++ enum), 95  
 mynteye::Capabilities (C++ enum), 92  
 mynteye::COLOR (C++ enumerator), 92  
 mynteye::CONTRAST (C++ enumerator), 93  
 mynteye::DEPTH (C++ enumerator), 92  
 mynteye::DESIRED\_BRIGHTNESS (C++ enumerator), 94  
 mynteye::Device (C++ class), 87  
 mynteye::Device::ConfigStreamRequest (C++ function), 88  
 mynteye::Device::Create (C++ function), 90  
 mynteye::Device::DisableMotionDatas (C++ function), 89  
 mynteye::Device::EnableMotionDatas (C++ function), 89  
 mynteye::device::Frame (C++ class), 90  
 mynteye::device::Frame::clone (C++ function), 90  
 mynteye::device::Frame::data (C++ function), 90  
 mynteye::device::Frame::format (C++ function), 90  
 mynteye::device::Frame::Frame (C++ function), 90  
 mynteye::device::Frame::height (C++ function), 90  
 mynteye::device::Frame::size (C++ function), 90  
 mynteye::device::Frame::width (C++ function), 90  
 mynteye::Device::GetExtrinsics (C++ function), 88  
 mynteye::Device::GetInfo (C++ function), 88  
 mynteye::Device::GetIntrinsics (C++ function), 88  
 mynteye::Device::GetLatestStreamData (C++ function), 89  
 mynteye::Device::GetModel (C++ function), 87  
 mynteye::Device::GetMotionDatas (C++ function), 90  
 mynteye::Device::GetMotionExtrinsics (C++ function), 88  
 mynteye::Device::GetMotionIntrinsics (C++ function), 88  
 mynteye::Device::GetOptionInfo (C++ function), 89  
 mynteye::Device::GetOptionValue (C++ function), 89  
 mynteye::Device::GetStreamData (C++ function), 89  
 mynteye::Device::GetStreamDatas (C++ function), 89  
 mynteye::Device::GetStreamRequest (C++ function), 88  
 mynteye::Device::GetStreamRequests (C++ function), 88  
 mynteye::Device::HasMotionCallback (C++ function), 89  
 mynteye::Device::HasStreamCallback (C++ function), 89  
 mynteye::Device::LogOptionInfos (C++ function), 89  
 mynteye::Device::motion\_callback\_t (C++ type), 87  
 mynteye::device::MotionData (C++ class), 91  
 mynteye::device::MotionData::imu (C++ member), 91  
 mynteye::Device::RunOptionAction (C++ function), 89  
 mynteye::device::select (C++ function), 100  
 mynteye::device::select\_request (C++ function), 100  
 mynteye::Device::SetExtrinsics (C++ function), 88  
 mynteye::Device::SetIntrinsics (C++ function), 88  
 mynteye::Device::SetMotionCallback (C++ function), 89  
 mynteye::Device::SetMotionExtrinsics (C++ function), 89  
 mynteye::Device::SetMotionIntrinsics (C++ function), 88  
 mynteye::Device::SetOptionValue (C++ function), 89  
 mynteye::Device::SetStreamCallback (C++ function), 89  
 mynteye::Device::Start (C++ function), 89  
 mynteye::Device::Stop (C++ function), 89  
 mynteye::Device::stream\_callback\_t (C++ type), 87  
 mynteye::device::StreamData (C++ class), 91  
 mynteye::device::StreamData::frame (C++ member), 87

mynteye::*member*), 91  
 mynteye::device::StreamData::frame\_id  
     (*C++ member*), 91  
 mynteye::device::StreamData::img    (*C++ member*), 91  
 mynteye::Device::Supports (*C++ function*), 87, 88  
 mynteye::Device::WaitForStreams    (*C++ function*), 89  
 mynteye::DEVICE\_NAME (*C++ enumerator*), 93  
 mynteye::DISPARITY (*C++ enumerator*), 92  
 mynteye::DISPARITY\_NORMALIZED (*C++ enumerator*), 92  
 mynteye::DisparityComputingMethod (*C++ enum*), 96  
 mynteye::ERASE\_CHIP (*C++ enumerator*), 94  
 mynteye::EXPOSURE\_MODE (*C++ enumerator*), 94  
 mynteye::Extrinsics (*C++ class*), 98  
 mynteye::Extrinsics::Inverse (*C++ function*), 98  
 mynteye::Extrinsics::rotation (*C++ member*), 99  
 mynteye::Extrinsics::translation    (*C++ member*), 99  
 mynteye::FIRMWARE\_VERSION (*C++ enumerator*), 93  
 mynteye::FISHEYE (*C++ enumerator*), 92  
 mynteye::Format (*C++ enum*), 95  
 mynteye::FRAME\_RATE (*C++ enumerator*), 93  
 mynteye::GAIN (*C++ enumerator*), 93  
 mynteye::GREY (*C++ enumerator*), 95  
 mynteye::GYROSCOPE\_LOW\_PASS\_FILTER (*C++ enumerator*), 94  
 mynteye::GYROSCOPE\_RANGE (*C++ enumerator*), 94  
 mynteye::HARDWARE\_VERSION (*C++ enumerator*), 93  
 mynteye::HDR\_MODE (*C++ enumerator*), 94  
 mynteye::ImgData (*C++ class*), 99  
 mynteye::ImgData::exposure\_time    (*C++ member*), 99  
 mynteye::ImgData::frame\_id (*C++ member*), 99  
 mynteye::ImgData::timestamp (*C++ member*), 99  
 mynteye::IMU (*C++ enumerator*), 92  
 mynteye::IMU\_FREQUENCY (*C++ enumerator*), 93  
 mynteye::IMU\_TYPE (*C++ enumerator*), 93  
 mynteye::ImuData (*C++ class*), 99  
 mynteye::ImuData::accel (*C++ member*), 99  
 mynteye::ImuData::flag (*C++ member*), 99  
 mynteye::ImuData::frame\_id (*C++ member*), 99  
 mynteye::ImuData::gyro (*C++ member*), 99  
 mynteye::ImuData::temperature (*C++ member*), 99  
 mynteye::ImuData::timestamp (*C++ member*), 99  
 mynteye::ImuIntrinsics (*C++ class*), 98  
 mynteye::ImuIntrinsics::bias (*C++ member*), 98  
 mynteye::ImuIntrinsics::noise (*C++ member*), 98  
 mynteye::ImuIntrinsics::scale (*C++ member*), 98  
 mynteye::Info (*C++ enum*), 93  
 mynteye::INFRARED (*C++ enumerator*), 92, 95  
 mynteye::INFRARED2 (*C++ enumerator*), 92, 95  
 mynteye::IntrinsicsEquidistant    (*C++ class*), 97  
 mynteye::IntrinsicsEquidistant::coeffs  
     (*C++ member*), 98  
 mynteye::IntrinsicsPinhole (*C++ class*), 97  
 mynteye::IntrinsicsPinhole::coeffs (*C++ member*), 97  
 mynteye::IntrinsicsPinhole::cx (*C++ member*), 97  
 mynteye::IntrinsicsPinhole::cy (*C++ member*), 97  
 mynteye::IntrinsicsPinhole::fx (*C++ member*), 97  
 mynteye::IntrinsicsPinhole::fy (*C++ member*), 97  
 mynteye::IntrinsicsPinhole::model (*C++ member*), 97  
 mynteye::IR\_CONTROL (*C++ enumerator*), 94  
 mynteye::ISP\_VERSION (*C++ enumerator*), 93  
 mynteye::KANNALA\_BRANDT (*C++ enumerator*), 95  
 mynteye::LEFT (*C++ enumerator*), 91  
 mynteye::LEFT\_RECTIFIED (*C++ enumerator*), 92  
 mynteye::LENS\_TYPE (*C++ enumerator*), 93  
 mynteye::MAX\_EXPOSURE\_TIME (*C++ enumerator*), 94  
 mynteye::MAX\_GAIN (*C++ enumerator*), 94  
 mynteye::MIN\_EXPOSURE\_TIME (*C++ enumerator*), 94  
 mynteye::Model (*C++ enum*), 91  
 mynteye::MOTION\_TRACKING (*C++ enumerator*), 95  
 mynteye::MotionIntrinsics (*C++ class*), 98  
 mynteye::MotionIntrinsics::accel  
     (*C++ member*), 98  
 mynteye::MotionIntrinsics::gyro    (*C++ member*), 98  
 mynteye::NOMINAL\_BASELINE (*C++ enumerator*), 93  
 mynteye::Option (*C++ enum*), 93  
 mynteye::OptionInfo (*C++ class*), 96

mynteye::OptionInfo::def (*C++ member*), 96  
mynteye::OptionInfo::max (*C++ member*), 96  
mynteye::OptionInfo::min (*C++ member*), 96  
mynteye::PINHOLE (*C++ enumerator*), 95  
mynteye::POINTS (*C++ enumerator*), 92  
mynteye::Resolution (*C++ class*), 96  
mynteye::Resolution::height (*C++ member*),  
    96  
mynteye::Resolution::width (*C++ member*),  
    96  
mynteye::RGB888 (*C++ enumerator*), 95  
mynteye::RIGHT (*C++ enumerator*), 92  
mynteye::RIGHT\_RECTIFIED (*C++ enumerator*),  
    92  
mynteye::SERIAL\_NUMBER (*C++ enumerator*), 93  
mynteye::SGBM (*C++ enumerator*), 96  
mynteye::Source (*C++ enum*), 95  
mynteye::SPEC\_VERSION (*C++ enumerator*), 93  
mynteye::STANDARD (*C++ enumerator*), 91  
mynteye::STANDARD2 (*C++ enumerator*), 91  
mynteye::STANDARD210A (*C++ enumerator*), 91  
mynteye::STEREO (*C++ enumerator*), 92  
mynteye::STEREO\_COLOR (*C++ enumerator*), 92  
mynteye::Stream (*C++ enum*), 91  
mynteye::StreamRequest (*C++ class*), 97  
mynteye::StreamRequest::format (*C++ mem-  
ber*), 97  
mynteye::StreamRequest::fps (*C++ member*),  
    97  
mynteye::StreamRequest::height (*C++ mem-  
ber*), 97  
mynteye::StreamRequest::width (*C++ mem-  
ber*), 97  
mynteye::UNKNOW (*C++ enumerator*), 96  
mynteye::utils::get\_real\_exposure\_time  
    (*C++ function*), 100  
mynteye::utils::get\_sdk\_install\_dir  
    (*C++ function*), 100  
mynteye::utils::get\_sdk\_root\_dir (*C++  
function*), 100  
mynteye::VIDEO\_STREAMING (*C++ enumerator*),  
    95  
mynteye::YUYV (*C++ enumerator*), 95  
mynteye::ZERO\_DRIFT\_CALIBRATION (*C++ enu-  
merator*), 94

