Optor Visual-Inertial Camera
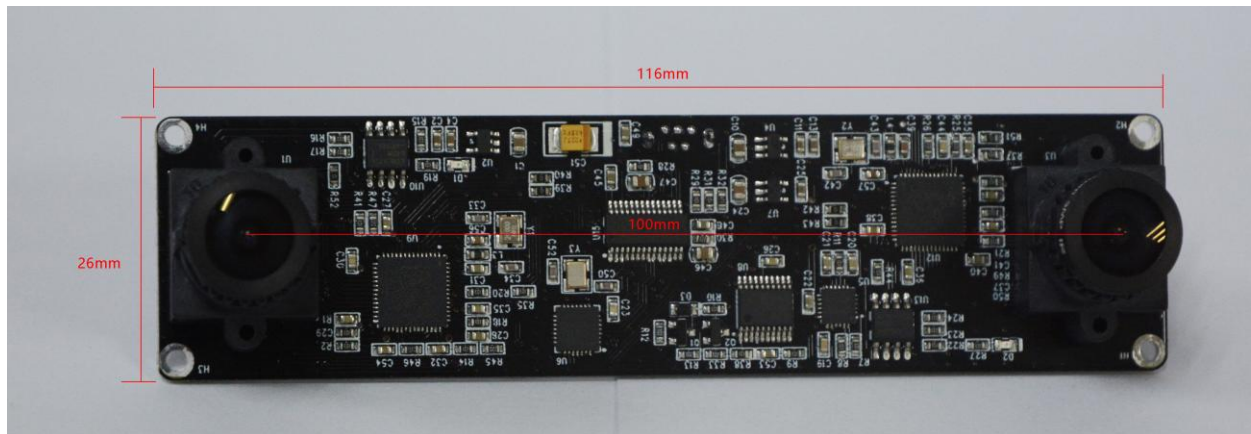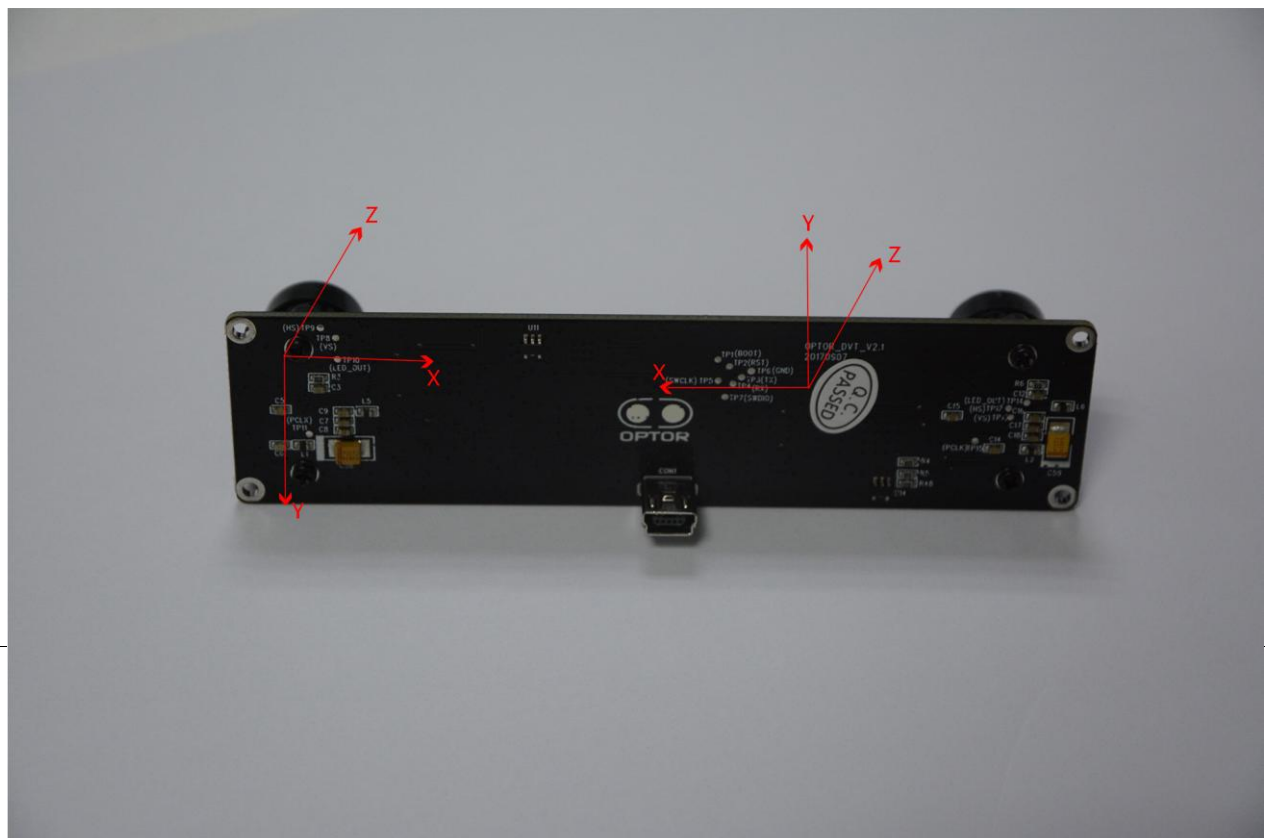
Instruction Manual

V0.4

Optor Visual Inertial Camera is a general vision sensor designed for visual algorithm developers.Providing abundant hardware control interface and data interface aimed to reduce development threshold with reliable image and inertial data.
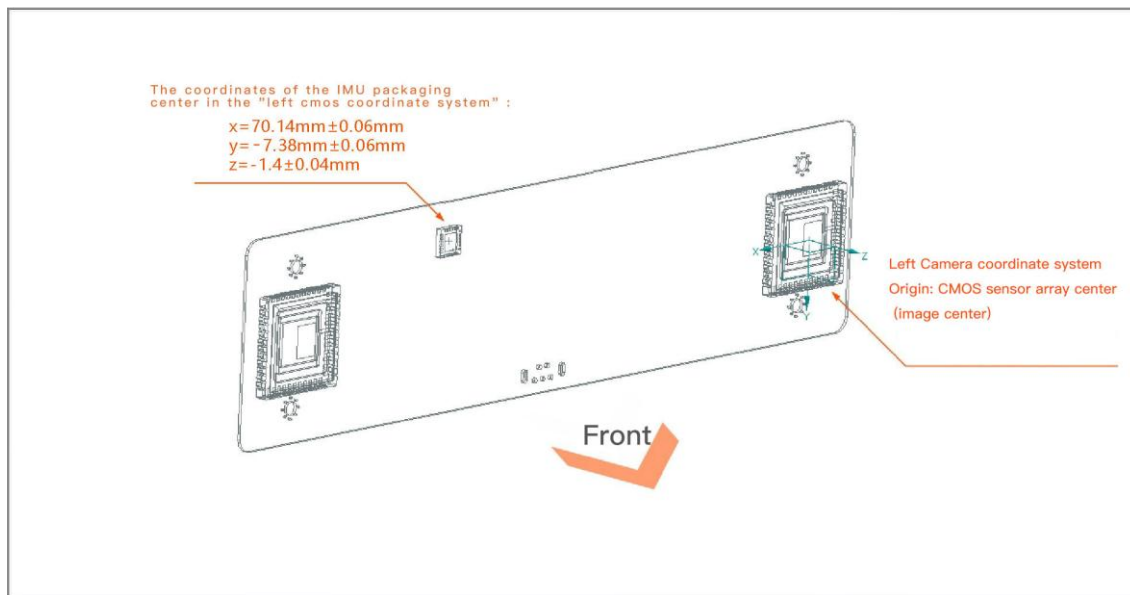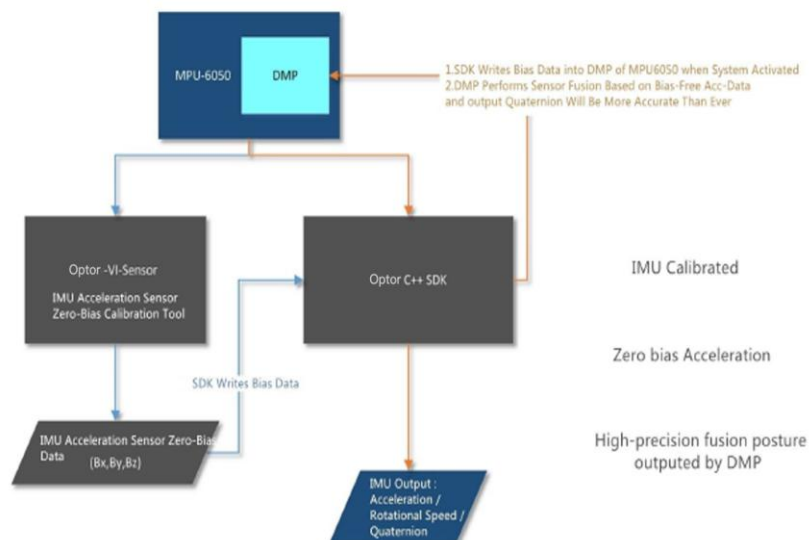
1.Hardware Specifications

   1.1 Physical Dimensions

   1.2 Camera Coordinate System between Left camera and IMU

The coordinates of the IMU packaging center in the "left cmos coordinate system" :
x=70.14mm±0.06mm
y=-7.38mm±0.06mm
z=-1.4±0.04mm

Left Camera coordinate system
Origin: CMOS sensor array center
(image center)

Front

1.3 Hardware Performance and Specifications

MPU-6050   DMP

1.SDK Writes Bias Data into DMP of MPU6050 when System Activated
2.DMP Performs Sensor Fusion Based on Bias-Free Acc-Data and output Quaternion Will Be More Accurate Than Ever

Optor -VI-Sensor
IMU Acceleration Sensor Zero-Bias Calibration Tool

Optor C++ SDK

IMU Calibrated

SDK Writes Bias Data

Zero bias Acceleration

IMU Acceleration Sensor Zero-Bias Data   (Bx,By,Bz)

IMU Output :
Acceleration /
Rotational Speed /
Quaternion

High-precision fusion posture outputed by DMP

2.Product Feature

	CMOS	IMU
Type	MT9V034	MPV-6050
Exposure Mode	Global shutter	-
Controller IC	CY68013	STM-32
FPS	24-65fps	200fps
Supported Resolution	640*480/752*480	-
Firmware Update	Firmware Update Supported By Windows Software	-
Baseline	10cm	-
Lens physical interface	M12 Lens interface	-
Lens Specifications	2.1mm/150º	-
Data interface	Usb 2.0	
Data Delay	(1/Current_FPS)s	100us
Frame Synchronization	Stereo Synchronization Triggered By Camera Driver	-

2.1 IMU zero bias calibration program, Zero bias initialization algorithm of DMP, high precision 6-DOF data, Minimum attitude drift.

2.2 Stereo optical parameters already accurately calibrated.

2.3 The lens seat rifled through special processing, to ensure the camera would not loosen in the long-term delivery and the lens can be replaced.
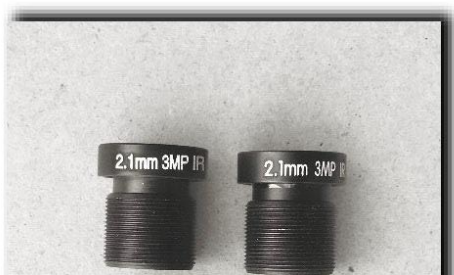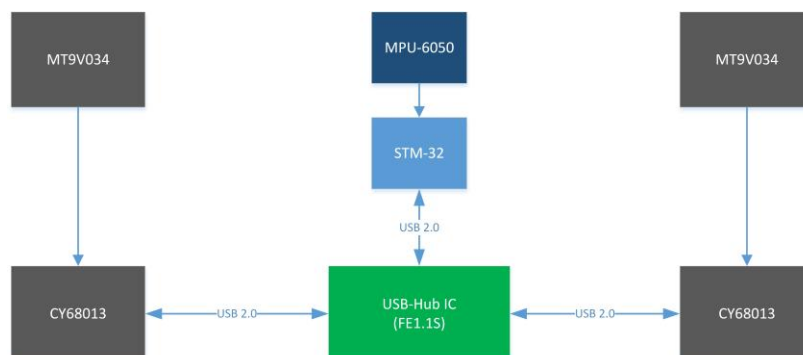

2.4 SDK needs no compilation,no special dependency libraries (only relay on libusb)
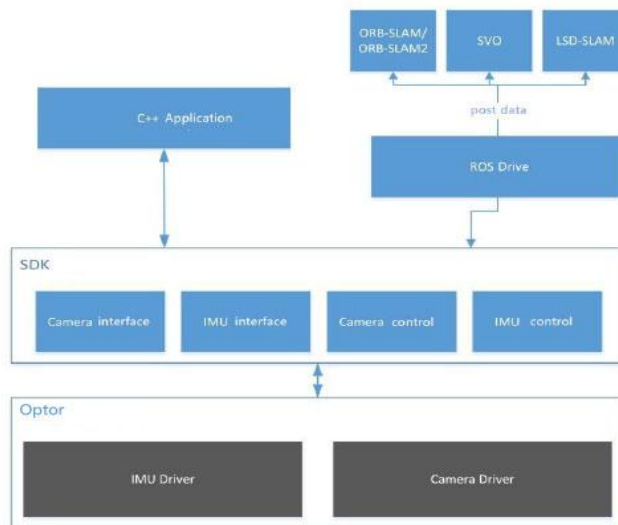
2.5 stable and reliable ROS driver

2.6 Ubuntu 16/14 supported

3.Hardware&Software Architecture

3.1 Hardware Architecture

3.2 SDK Architecture

4. SDK directory structure and Demo compilation step by step

4.1 SDK directory structure

4.2 C++ sample program compilation

This program is the minimum demonstration of opter SDK, please make sure Opencv has been successfully installed on your computer.If not installed, the latest version of Opencv can be downloaded in official website.choose your work space, such as /home/workspace/

1.Copy /optor_VI_Sensor_SDK_V1.1/SDK to /home/workspace/

2.Start command line, enter the ROOT

   sudo -s

3.Enter SDK catalog

   cd /home/workspace/optor_VI_Sensor_SDK_V1.1/SDK

4.Execute CMake

   make build && cd build

   cmake ..

5.Compile the sample program

   make

6.Execute Demo program

   ./camtest

7.Applications results

8.If program failed, please see 11.1 for solution.

4.3 ROS Package

   ROS Package this Package rely on

cv_bridge

image_transport

opencv2

roscpp

rospy

sensor_msgs

std_msgs

1.Find your catkin workspace and enter /src catalog

cd /home/workspace/catkin_ws/src

2.Copy /optor_VI_Sensor_SDK_V1.1/ROS/optor_stereo_visensor to catkin_ws/src

3.Enter catkin workspace

cd /home/workspace/catkin_ws

4.Compile

catkin_make

5.If compilation successful, the ROS Node can run normally; Otherwise you need to check whether Optor ROS Package has been successfully installed in ROS system.

4.4 ROS Package:How to start

1.Before running, update the ROS environment variables

source /devel/setup.bash

2.Once step 4.3 is successful, you can start the ROS node, the command is:

roscore

rosrun optor_stereo_visensor_ros stereo_visensor_node

SETTINGS_FILE_PATH

You need replace SETTINGS_FILE_PATH with the actual path of "optor_VISensor_Setups.txt", such as:

/home/di-tech/workspace/optor_VI_Sensor_SDK_V1.0/SDK/ optor_VISensor_Setups.txt

Or directly to catkin_ws/src/optor_stereo_visensor and start the order：

roscore

rosrun optor_stereo_visensor_ros stereo_visensor_node

3.expected output:

```
😣😐🔲  yjzb@yjzb-desktop: ~/catkin_ws
m13,Manual Mode
2
150
WVGA
54
EG_mode
4
manual,50,200
auto,300,5,58
autoexp_manualgain,300,5,58,200
/dev/ttyUSB0,5
IMU-acc-bias
Gx,52.000000
Gy,32.00001
Gz,-243.000000
Now opening cameras...
Number of device of interest found: 2
Left camera found!
Right camera found!
visensor_open_port success...
set done!
visensor_set_opt(fd,115200,8,'N',1) success...
25
```

4.If your ROS operation permission is not the root but the average user, it may lead to the problem that ROS driver quit automatically. You can solve the problem following 11.1.

5. Get the Timestamp Data

Image data and IMU data have been added precise timestamp by Optor ROS Drive, you can get them in msg.header.Stamp.

5. Camera Configuration File

If you look at the usage of main() in camtest.CPP file, you will find the use of SDK is very simple:

1. visensor_load_settings("optor_VISensor_Setups.txt");

2. visensor_Start_Cameras();   // start the camera

3. visensor_Start_IMU();   // start IMU

first, we require the system load the camera configuration file, since the file contains several parameters for camera and IMU.Follow the picture:
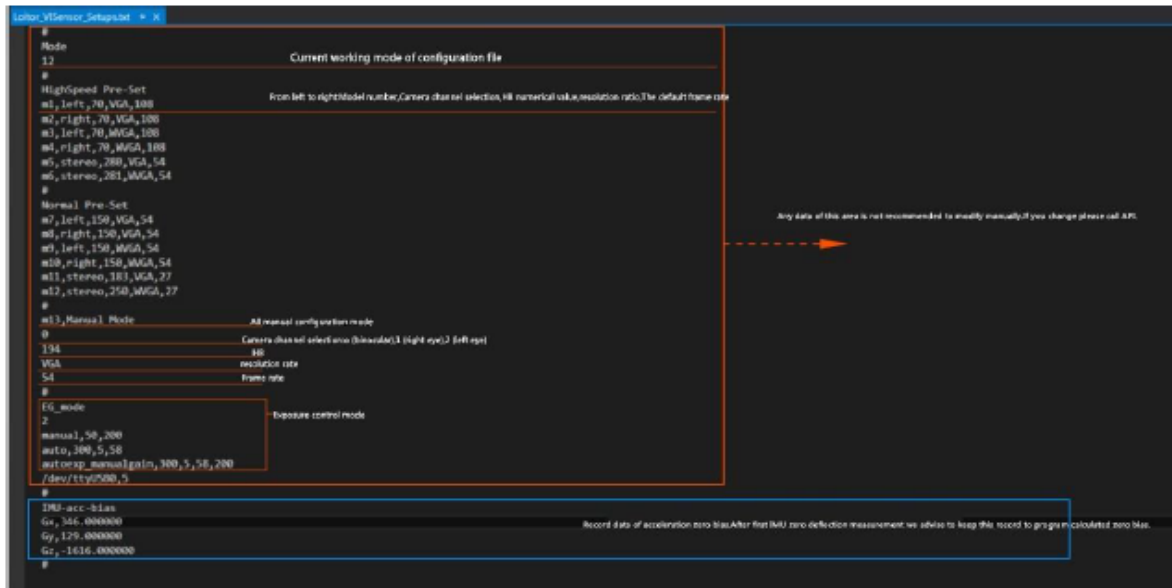
as the picture shows Blue area is used to record imu zero bias, and specific set method will be given in step 8.

6. visensor_API Instructions

  6.1 Camera Controls

    1.void visensor_set_auto_EG(int E_AG); Set the exposure mode:

    E_AG=0 means manual mode, you need to set the exposure value manually;

    E_AG=1 means automatic mode (Limit of 0-255);

    E_AG=2 means automatic&manual mode(setting the gain value): visensor_set_gain()

    E_AG=3 means automatic gain&exposure mode

    2.void visensor_set_exposure(int_man_exp); Set manual exposure value (Limit of 0-255)

    3.void visensor_set_gain(int_man_gain); Set the manual gain value

    4.void visensor_set_max_autoExp(int max_exp);

      Set maximum on automatic exposure mode (limit of 0-255)

    5.void visensor_set_max_autoEXP(int max_exp);

      Set minimum of automatic exposure mode (limit of 0-255)

    6.void visensor_set_resolution(bool set_wvga);

      Set the resolution,set_wvga=ture means 752*480 otherwise 640*480;

    7.void visensor_set_fps_mode(bool fps_mode);

      Set the frame rate:

      fps_mode=ture means High frame rate:

     According to HB under WVGA, range from 40fps-50fps; According to HB under VGA, range from 50fps-65fps; Fps_mode=ture means Low frame rate:

According to HB under WVGA, range from 22fps-27fps; According to HB under VGA, range from 22fps-25fps;

8.void visensor_set_current_HB(int HB);

Set current HB .range from 70-255;

9.void visensor_set_desired_bin(int db);

Set up automatic exposure to "Achieve brightness", CMOS will adjust exposure automatically according to the numerical. The higher the db, the brighter the image ranging from 0 to 48.

10.void visensor_set_cam_selection_mode(int_visensor_cam_selection); You can choose different mode of "left eye","right eye"and "binocular"through Camera.

_visensor_cam_selection=0 means "binocular";

_visensor_cam_selection=1 means "right eye";

_visensor_cam_selection=2 means "left eye";

11.void visensor_set_current_mode(int_mode); Change the current camera work mode manually.

12.int visensor_Start_Cameras();｜void visensor_Close_Cameras(); Start and turn off the camera safely.

Function Visensor_Start_Cameras()should be called after calling function 1-11 which means if you need to change the camera settings through API it should be acted before visensor_Start_Cameras.

13.void visensor_save_current_settings();

to save the current mode settings in the configuration file should be acted after function 1-11.

6.2 IMU Control

1.visensor_Start_IMU();

Open the serial port and start IMU.

2.void visensor_Close_IMU();

Turn off IMU safely.

3.void visensor_set_imu_portname(chat*input_name); Set the

serial port name manually.

6.3 IMU Data

1.void visensor_set_imu_bias(float bx,float by,float bz); Setting IMU zero bias manually.

2.visensor_imudata visensor_imudata_pack

The variables are set as globle variables to record current IMU data(timestamp included) and it can be referenced as optorimu.h

7. Setting HB parameters manually

HB (Horizontal Blanking) line of Blanking, a CMOS MT9V034 register which is an important parameter, it can affect the size of the frame rate, if the setting is not appropriate (too big or too small) also can result in images caton, frame lost even USB connection failure.

The greater the HB, each frame transmission time is longer, the lower frame rate; If your USB bus capacity is limited, we suggest set the HB to around 250.

HB is smaller, the higher acquisition frame rate, but the greater the pressure for USB transmission. Smaller HB values (such as 120-194) is more suitable for PC with strong USB transmission ability.

If you find the camera there lost frames and caton, modify HB values through the API, and then call visensor_save_current_settings () to save the Settings.

8.IMU accelerometer zero calibration procedure

1.Follow the command line to enter IMU_Calib folder, start

./imucalib

to set zero calibration according to clew .

2.After calibration completed it will automatically write into the configuration file: imucaldata.txt then you need to manually copy the three parameters in the configuration file to Optor_VISensor_Setups.txt in the blue area.

9. Optical calibration

We use the ROS camera_calibration factory for camera calibration, its parameters has been written to the file:/ORB_SLAM2/Examples/Stereo/EuRoC.yaml
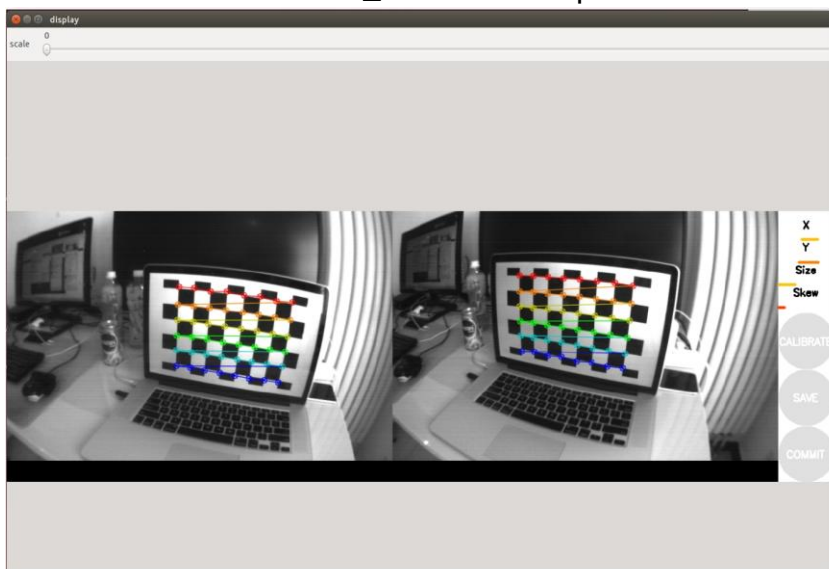
10. Compile the ORB-SLAM2 and use Optor camera for testing

1.Following the ORB-SLAM2 official guidance for ORB-SLAM2 compilation and configuration.

2.Since the optical calibration has been completed,first run Optor ROS Node after compilation, then run the ORB - SLAM2 following default configuration.

11. Several Problems you may meet
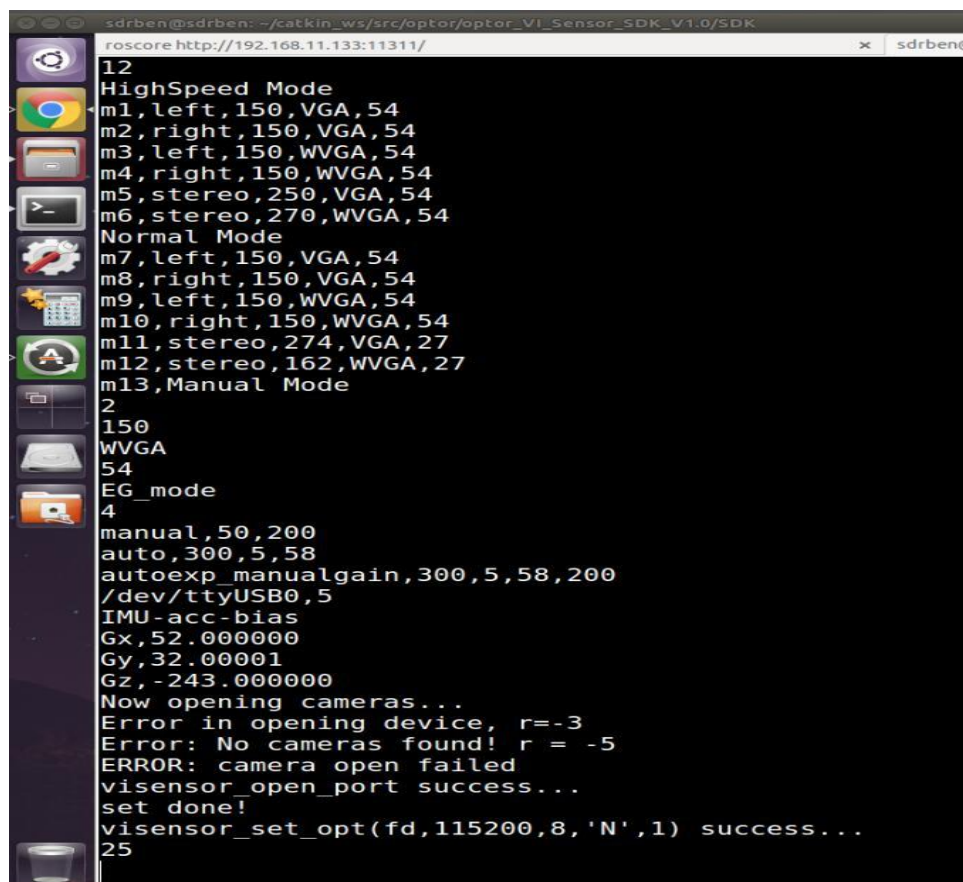
11.1 serial file cannot be opened

1.The serial port name is wrong; You need to check the USB device serial path, and write the paths in the corresponding part in Optor_VISensor_Setups.txt.

2.Ordinary users can't open the serial port under file (access problem) : Under Linux device you need to use sudo or the root user and in order to make ordinary users also can use a serial port you can increase the udev rules to implement, the specific method is as follows:

sudo vim/etc/udev/rules. D / 70 - ttyusb. Rules Add the

following contents:

KERNEL = = "ttyUSB [0-9] *", the MODE = "0666", save and insert the USB to turn over serial port,it can be solved.

11.2 Ordinary users can not start the camera


```
sdrben@sdrben: ~/catkin_ws/src/optor/optor_VI_Sensor_SDK_V1.0/SDK
roscore http://192.168.11.133:11311/          ×   sdrben@
12
HighSpeed Mode
m1,left,150,VGA,54
m2,right,150,VGA,54
m3,left,150,WVGA,54
m4,right,150,WVGA,54
m5,stereo,250,VGA,54
m6,stereo,270,WVGA,54
Normal Mode
m7,left,150,VGA,54
m8,right,150,VGA,54
m9,left,150,WVGA,54
m10,right,150,WVGA,54
m11,stereo,274,VGA,27
m12,stereo,162,WVGA,27
m13,Manual Mode
2
150
WVGA
54
EG_mode
4
manual,50,200
auto,300,5,58
autoexp_manualgain,300,5,58,200
/dev/ttyUSB0,5
IMU-acc-bias
Gx,52.000000
Gy,32.00001
Gz,-243.000000
Now opening cameras...
Error in opening device, r=-3
Error: No cameras found! r = -5
ERROR: camera open failed
visensor_open_port success...
set done!
visensor_set_opt(fd,115200,8,'N',1) success...
25
```

If this problem occurs, run "optor-vi-install.sh" as a normal user on the command line

```
chmod +x optor-vi-install.sh
sudo ./optor-vi-install.sh
```

End of the operation, then usb hot swap.

12 When IMU is on electricity you need to wait for 8 to 10 seconds and initialize the gyroscope to zero bias, remain the camera still, otherwise will result in attitude drift obviously.