1、Astra camera calibration

Wiki：http://wiki.ros.org/camera_calibration

Official website link：https://orbbec3d.com/develop/

Astra camera：https://github.com/orbbec/ros_astra_camera

Due to the internal and external reasons of the camera, the image will be greatly distorted, mainly radial deformation and tangential deformation, which cause the straight line to become curved. The farther the pixel is from the center of the image, the more serious the distortion. In order to avoid errors caused by the data source, it is necessary to calibrate the parameters of the camera.

Disadvantages of infrared depth camera ranging:

(1) It is impossible to accurately measure the distance of a black object, because the black substance can absorb infrared rays, and the infrared rays cannot return, so the distance measurement cannot be performed.

(2) It is impossible to accurately measure the distance of the mirror-reflecting object, because only when the depth camera is on the vertical line of the mirror-surface object, the receiver can receive the reflected infrared rays, and it will cause overexposure.

(3) It is impossible to accurately measure the distance of transparent objects, because infrared rays can pass through transparent objects.

(4) It is impossible to accurately measure the distance of objects that are too close.

| Product Name	ASTRA PRO	ASTRA S	ASTRA
Range	0.6m – 8m	0.4m – 2m	0.6m – 8m
FOV	60°H x 49.5°V x 73°D		
RGB Image Res.	1280 x 720 @30fps	640 x 480 @30fps	
Depth Image Res.	640 x 480 @30fps		
Size	165mm x 30mm x 40mm		
Temperature	0 – 40°C		
Power Supply	USB 2.0		
Power Consumption	< 2.4 W		
Operating Systems	Android/Linux/Windows 7/8/10		
SDK	Astra SDK or OpenNI		
Microphones	2 (Built – in)		

1.1、 Preparation before calibration

- You need a [checkerboard](). This tutorial uses a 9x6 checkerboard and a 20mm square, which should be flattened during calibration.

  The calibration uses the internal vertex of the checkerboard, so the "10x7" checkerboard uses the internal vertex parameter "9x6", as shown in the example below.

  The calibration board of any specification can be used, just change the parameters.

- An open area without obstacles and calibration board patterns

- Monocular camera that publishes images via ROS

Checkerboard (calibration board)
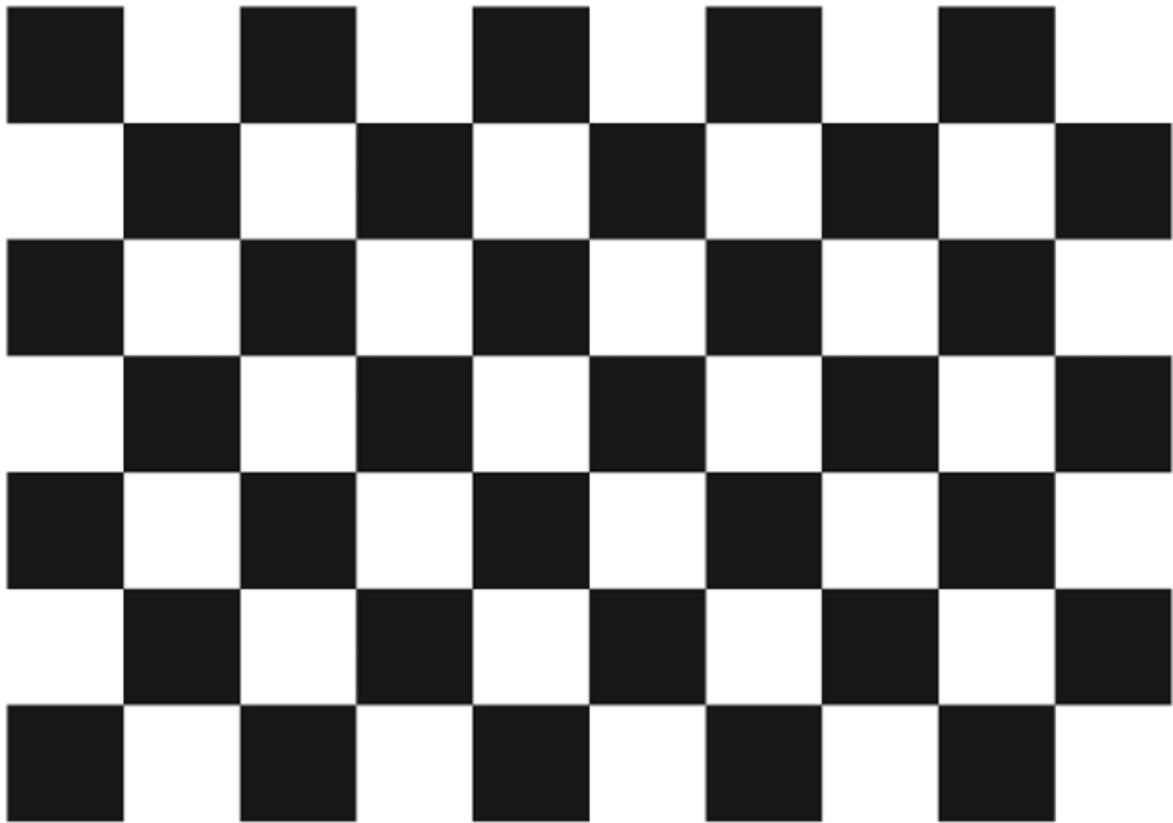
7×10 | Size: 20mm

Obi Zhongguang camera model and corresponding launch file

Launch file	Start the camera model
astra.launch	Astra，Astra S，Astra mini，Astra mini S
astraplus.launch	Astra plus
astraproplus.launch	Astra pro plus
embedded_s.launch	Deeyea
dabai_u3.launch	Dabai
gemini.launch	Gemini

Device view

```
lsusb
```

The appearance of these two IDs indicates that the device is connected.

1.2、 calibration

Satrt up Astra camera

```
roslaunch astra_camera astraproplus.launch
```

View picture topic

```
rostopic list
```


Start the calibration node

```
# Color map
rosrun camera_calibration cameracalibrator.py image:=/camera/rgb/image_raw
camera:=/camera/rgb --size 9x6 --square 0.02
```

size: The number of internal corners of the calibrated checkerboard, such as 9X6, the corners have six rows and nine columns.

square: The side length of the checkerboard, in meters.

image and camera: Set the topic of the image released by the camera.

1.2.1、 Color icon set
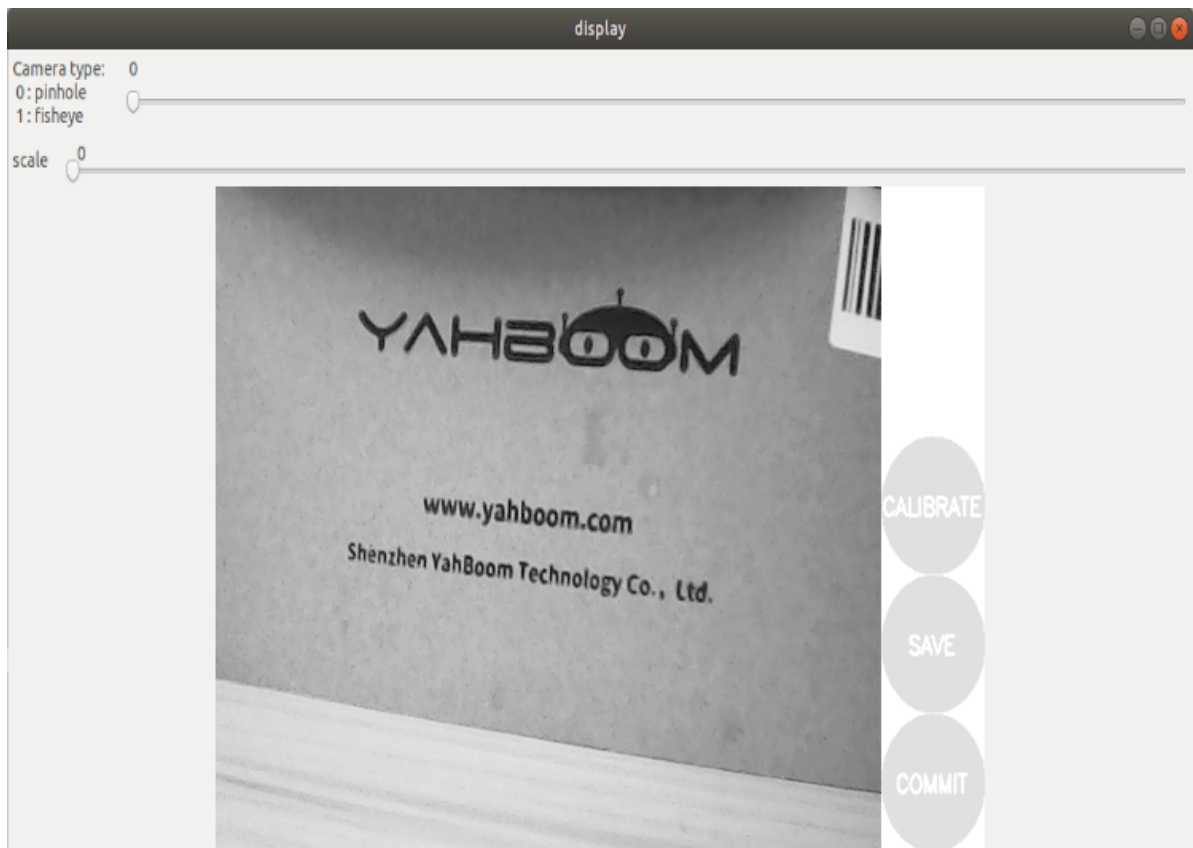
Calibration interface

X: The left and right movement of the checkerboard in the camera field of view

Y: The checkerboard moves up and down in the camera's field of view

Size: the back and forth movement of the checkerboard in the camera's field of view

Skew: The tilt and rotation of the checkerboard in the camera field of view

After the startup is successful, put the checkerboard into the center of the screen and change different poses. The system will recognize it autonomously. In the best case, the lines under [X], [Y], [Size], and [Skew] will first change from red to yellow and then to green as the data is collected, filling them as much as possible.
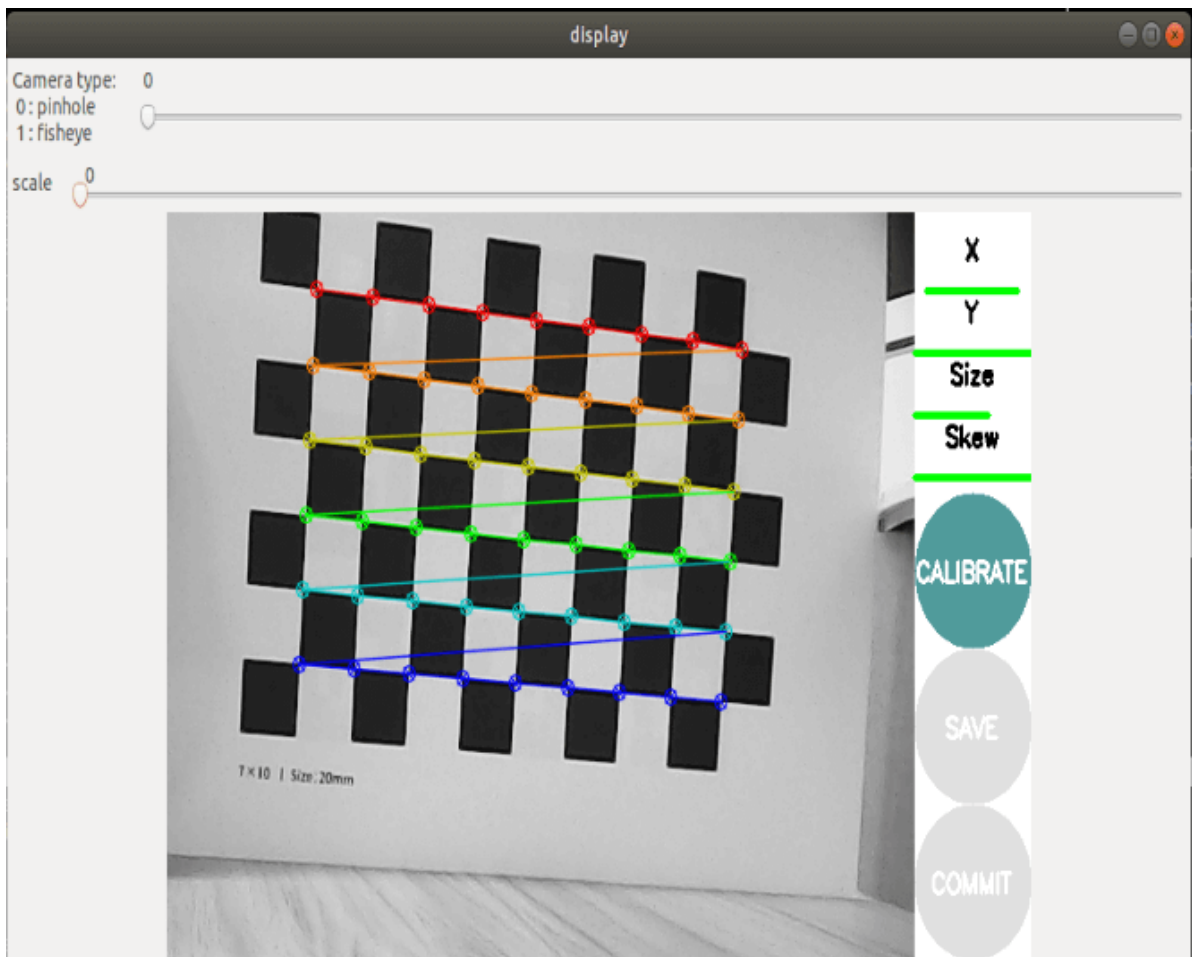
Click [CALIBRATE] to calculate the camera internal parameters, the more pictures, the longer the time, just wait.
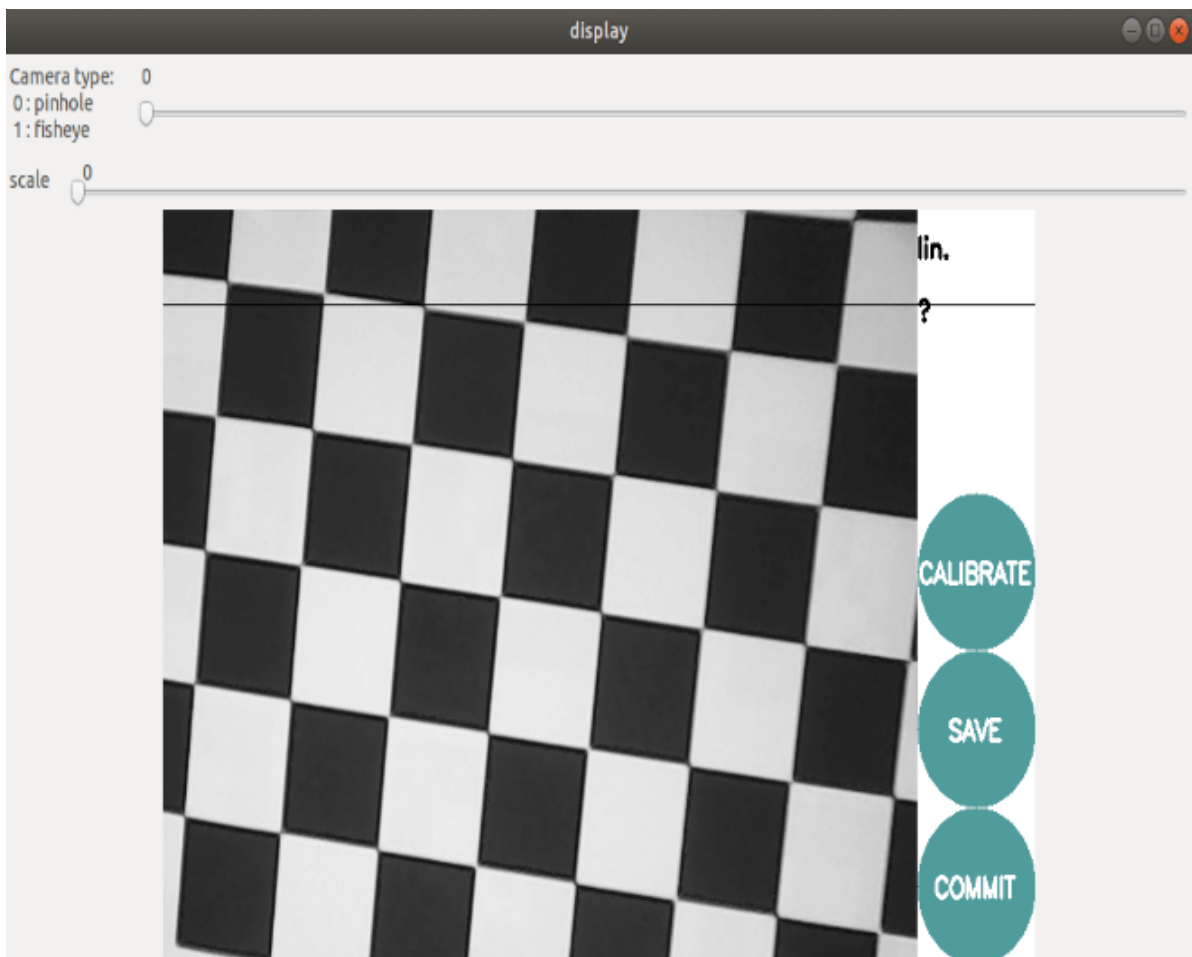
Click [SAVE] to save the uncalibrated result, the bottom line appears, click [COMMIT] to exit.

```
**** Calibrating ****
D = [-0.07028213194362816, 0.00043818252903837866, -0.01245084517224107, 0.000404835
0406427093, 0.0]
K = [543.8333273852593, 0.0, 344.1989291964055, 0.0, 544.5128476949725, 219.77155460
528877, 0.0, 0.0, 1.0]
R = [1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0]
P = [530.847900390625, 0.0, 345.7782327897439, 0.0, 0.0, 534.5553588867188, 214.9890
440488889, 0.0, 0.0, 0.0, 1.0, 0.0]
None
# oST version 5.0 parameters


[image]

width
640

height
480

[narrow_stereo]

camera matrix
543.833327 0.000000 344.198929
0.000000 544.512848 219.771555
0.000000 0.000000 1.000000

distortion
-0.070282 0.000438 -0.012451 0.000405 0.000000

rectification
1.000000 0.000000 0.000000
0.000000 1.000000 0.000000
0.000000 0.000000 1.000000

projection
530.847900 0.000000 345.778233 0.000000
0.000000 534.555359 214.989044 0.000000
0.000000 0.000000 1.000000 0.000000

('Wrote calibration data to', '/tmp/calibrationdata.tar.gz')
```

After the calibration, the calibration result is stored in [/tmp/calibrationdata.tar.gz], you can move it out to see the content

```
sudo mv /tmp/calibrationdata.tar.gz ~
```

After decompression, there are the image just calibrated, an ost.txt file and an ost.yaml file. The yaml file is what we need, but it needs to be modified before it can be used.

-Change ost.yaml to head_camera.yaml
-Change the name of camera_name: to head_camera
-Move the file to ~/.ros/camera_info folder

1.2.2、 Depth image calibration

Start the calibration node

```
rosrun camera_calibration cameracalibrator.py image:=/camera/ir/image
camera:=/camera/ir --size 9x6 --square 0.02
```

size: The number of internal corners of the calibrated checkerboard, such as 9X6, the corners have six rows and nine columns.

square: The side length of the checkerboard, in meters.

image and camera: Set the topic of the image released by the camera.

The calibration process is similar to color camera calibration, but the visualization of depth camera calibration is not so intuitive; it just displays the calibration result when it is recognized, and it is completely dark when it is not recognized.

```
rqt_image_view
```
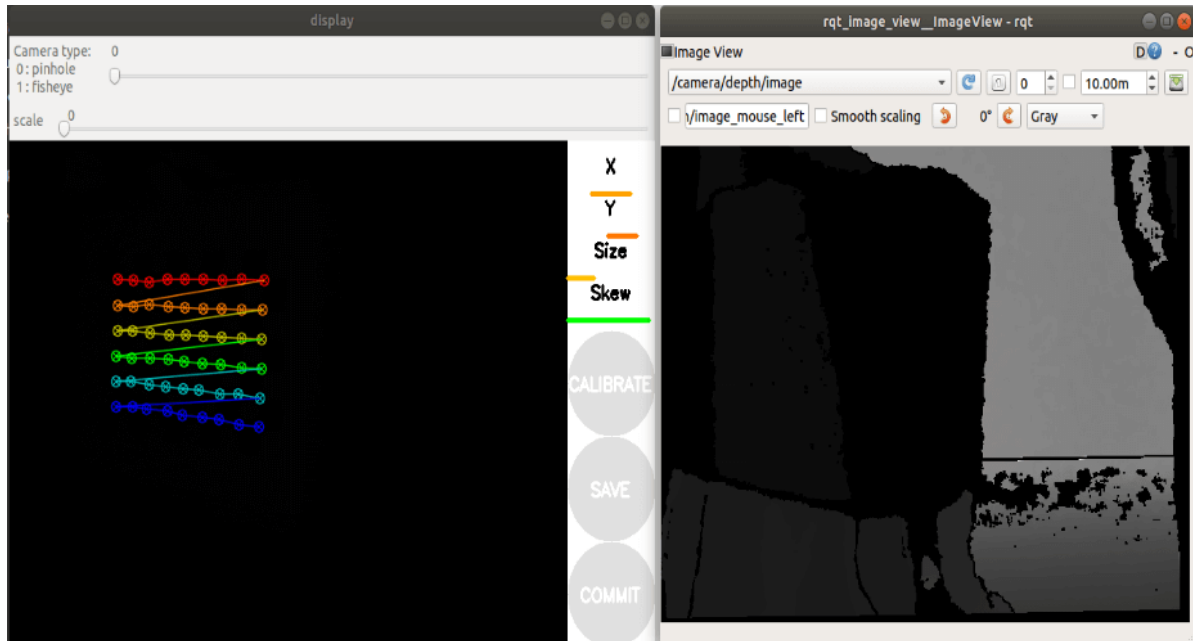

The following operations are similar to color camera calibration, changing different poses. The system will recognize it autonomously. In the best case, the lines under [X], [Y], [Size], and [Skew] will first change from red to yellow and then to green as the data is collected, filling them as much as possible. Click [CALIBRATE] to calculate the camera internal parameters, the more pictures, the longer the time, just wait.

Click [SAVE] to save the non-calibration result, the bottom line appears, click [COMMIT] to exit. After the calibration, the calibration result is stored in [/tmp/calibrationdata.tar.gz], you can move it out to see the content.

```
sudo mv /tmp/calibrationdata.tar.gz ~
```

After decompression, there are the image just calibrated, an ost.txt file and an ost.yaml file. The yaml file is what we need, but it needs to be modified before it can be used.

- Change ost.yaml to depth_Astra_Orbbec.yaml
- Change the name behind camera_name: to depth_Astra_Orbbec
- Move the file to ~/.ros/camera_info folder