# 2. Calibration of camera internal parameters



## 2.1 Description of program function

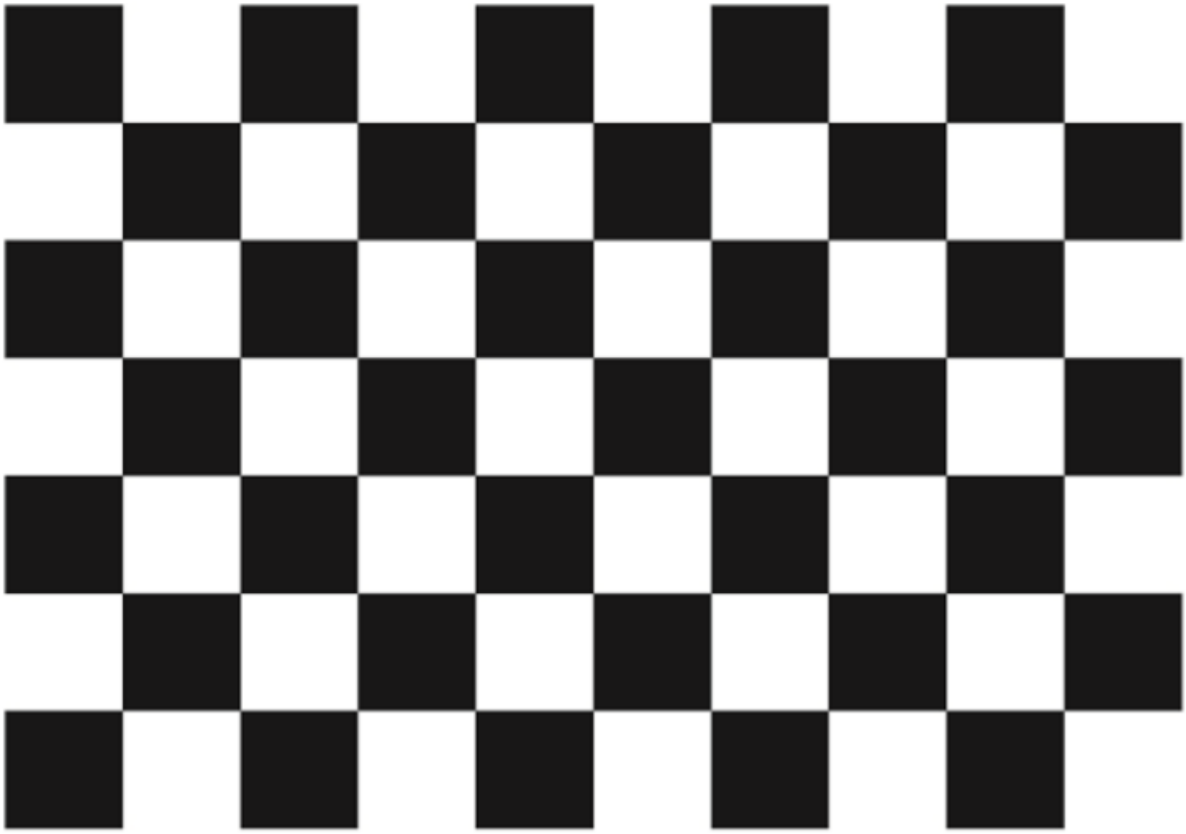
Due to some internal and external reasons of the camera will bring great distortion to the image, mainly radial deformation and tangential deformation, resulting in the bending of the straight line, the farther away from the image center of the pixel, the more serious the distortion. In order to avoid errors caused by data sources, it is necessary to calibrate camera parameters. Calibration is essentially the use of a known and determined spatial relationship (calibration board), by analyzing the picture pixels taken, the inherent and real parameters of the camera (internal parameters) are backwards derived.

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

## 2.2. Preparation before calibration

- A large chessboard of known size. This tutorial uses a 9x6 checkerboard and a 20mm square, flattened when calibrated. **Calibration uses the inner vertices of the checkerboard, so the "10x7" checkerboard uses the inner vertices parameter "9x6", as shown in the example below.** What specifications of the calibration board can be, just change the parameters.
- A more open area without obstacles and demarcating plate patterns.
- Monocular cameras that publish images via ROS.

Checkerboard (calibration board)



7×10 | Size: 20mm

# 2.3. Program Startup

## 2.3.1. Start the camera

Take starting Astra pro camera as an example. After SSH connects to the car, enter,

```
ros2 launch astra_camera astra_pro.launch.xml
```

Start the camera before calibration and turn the camera off until all calibration is complete.

Obizhong Camera model and corresponding launch file:

| launch file | Camera model |
| --- | --- |
| ros2 launch astra_camera astra_pro.launch.xml | Astrapro |
| ros2 launch astra_camera astro_pro_plus.launch.xml | Astraproplus |
| ros2 launch astra_camera astra.launch.xml | Astramini |

Since the car has no graphical interface, the next calibration procedure needs to be run in a virtual machine.

Enter in the VM terminal,

```
ros2 topic list
```

```
yahboom@yahboom-virtual-machine:~$ ros2 topic list
/camera/color/camera_info
/camera/color/image_raw
/camera/depth/camera_info
/camera/depth/image_raw
/camera/depth/points
/camera/ir/camera_info
/camera/ir/image_raw
/parameter_events
/rosout
/tf
/tf_static
```

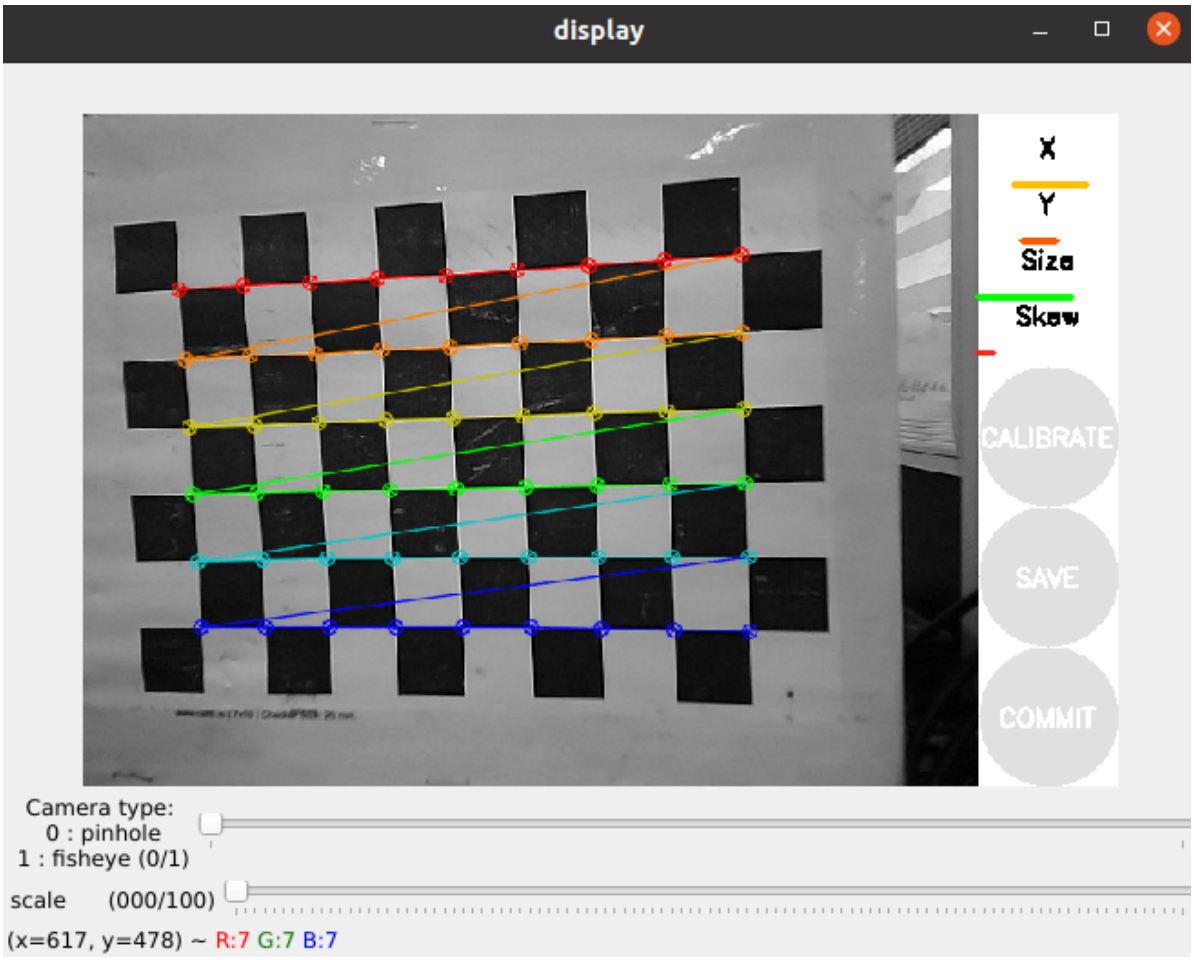
The topic we need to use for calibration RGB color images is /camera/color/image_raw, and the topic for calibration ir infrared images is /camera/ir/image_raw.

## 2.3.2. Color image Calibration

Enter in the VM terminal,

```
ros2 run camera_calibration cameracalibrator --size 9x6 --square 0.02 --ros-args
--remap /image:=/camera/color/image_raw
```

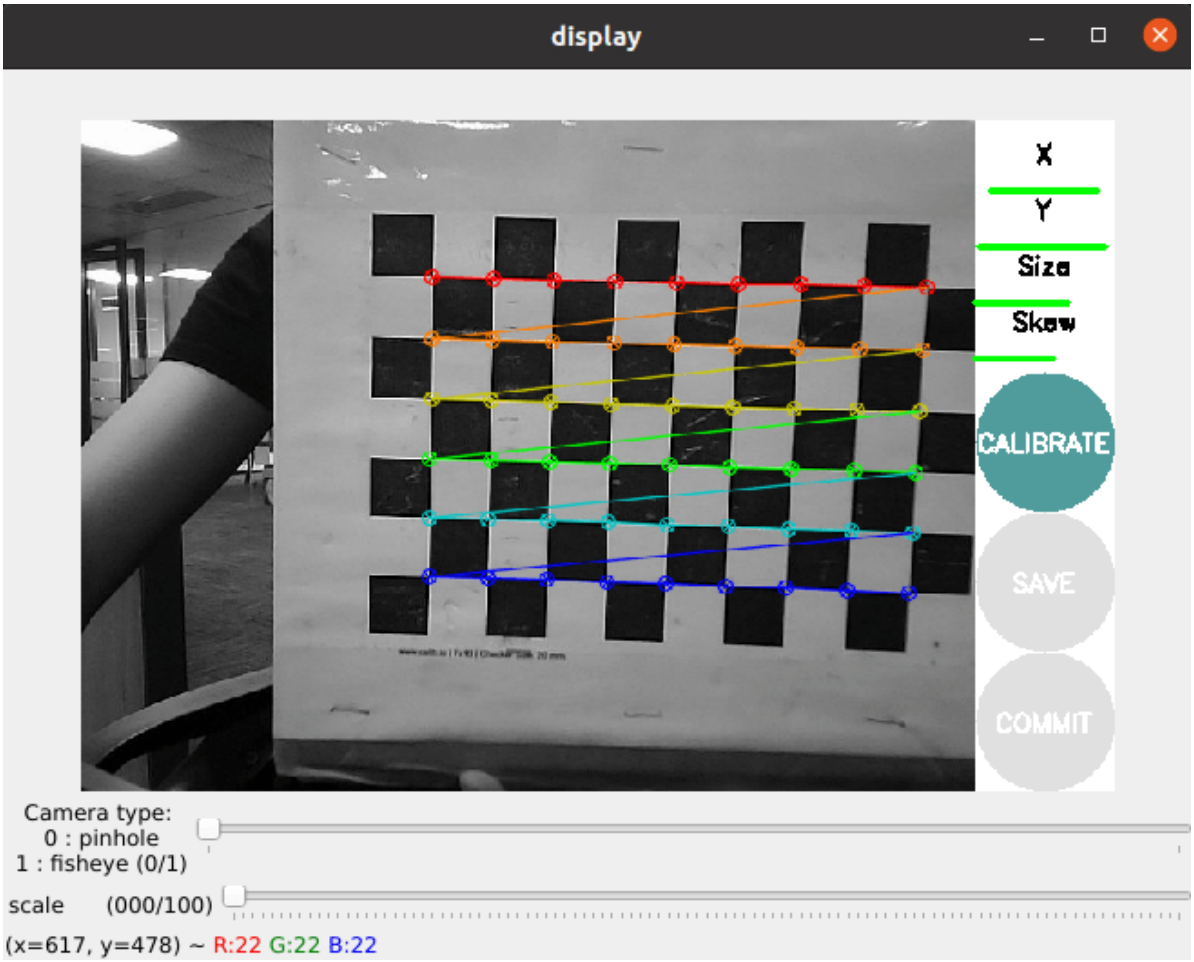
- size: indicates the number of corners in the checkerboard, for example, 9X6. There are six rows and nine columns of corners.
- square: indicates the length of the side of the checkerboard, in meters.
- /camera/color/image_raw: RGB image topic name.




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

- Y: indicates the movement of the checkerboard in the camera field
- Size: indicates the movement of the checkerboard in the camera field
- Skew: The oblique rotation of the checkerboard in the camera's field of view

Capture images by moving up, down, back and forth and turning them left and right so that [X], [Y], [Size], and [Skew] on the right turn green, as shown below. Then click [CALIBRATE] to start calculating camera parameters. The more pictures, the longer it takes. Just wait. (Sixty or seventy pieces is about right, too many are easy to get stuck).




After calibration, click [SAVE], as shown in the picture below.

```
**** Calibrating ****
mono pinhole calibration...
D =  [0.12868218471079662, -0.18479677076960513, 0.006064246684526561, -0.00706579289782093, 0.0]
K =  [609.4015644177261, 0.0, 315.7804514886843, 0.0, 609.3426094090089, 250.64776122179921, 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 =  [624.255615234375, 0.0, 311.3250520961956, 0.0, 0.0, 625.3106689453125, 252.69370711339434, 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
609.401564 0.000000 315.780451
0.000000 609.342609 250.647761
0.000000 0.000000 1.000000

distortion
0.128682 -0.184797 0.006064 -0.007066 0.000000

rectification
1.000000 0.000000 0.000000
0.000000 1.000000 0.000000
0.000000 0.000000 1.000000

projection
624.255615 0.000000 311.325052 0.000000
0.000000 625.310669 252.693707 0.000000
0.000000 0.000000 1.000000 0.000000


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

Calibration results to the/TMP/calibrationdata. Tar. Gz 】, can move the file out to check the content.

After exiting the program, enter the virtual machine terminal,

```
#文件移动至当前目录
# file moved to the current directory
sudo mv /tmp/calibrationdata.tar.gz .
#创建文件夹
# Create folder
mkdir calibrationdata_rgb
#解压至指定文件夹
# Unzip to the specified folder
tar -zxvf calibrationdata.tar.gz -C ./calibrationdata_rgb
```

```
yahboom@yahboom-virtual-machine:~$ sudo mv /tmp/calibrationdata.tar.gz .
yahboom@yahboom-virtual-machine:~$ mkdir calibrationdata_rgb
yahboom@yahboom-virtual-machine:~$ tar -zxvf calibrationdata.tar.gz -C ./calibrationdata_rgb
left-0000.png
left-0001.png
left-0002.png
left-0003.png
left-0004.png
left-0005.png
```

The calibrated png file, ost.yaml, and ost.txt files are obtained in the calibrationdata_rgb folder in this directory.

Since the Astra driver starts by loading the calibrated built-in parameters in the code, there is no need to load the calibrated parameters. Camera parameters are used in the AR Vision case in the opencv series.

## 2.3.3. ir infrared calibration

Because the RGBD camera uses structured light as the depth imaging principle of the camera, the infrared light projected by it is a special disordered spot, resulting in the infrared receiving device can not receive clear and complete picture content.

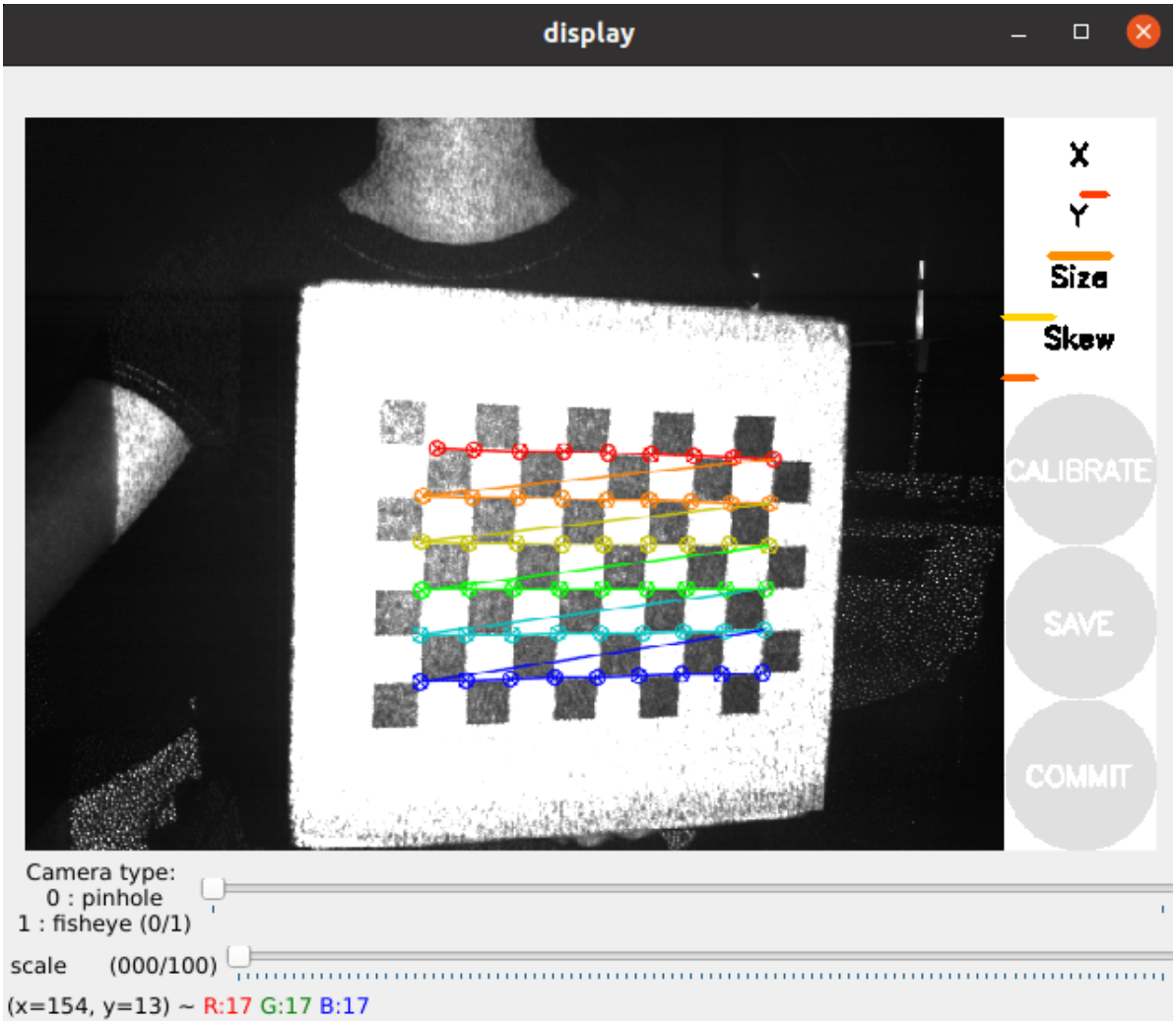
At this point we can have several special treatment methods:

- Forced to find a variety of angles, as far as possible to let the camera find corners (poor accuracy)
- Smooth out the infrared spot by placing some frosted translucent paper in front of the red hair transmitter (moderate accuracy, more convenient)
- Block the infrared projection camera and use the external infrared camera to fill the light (high precision, additional equipment is required)

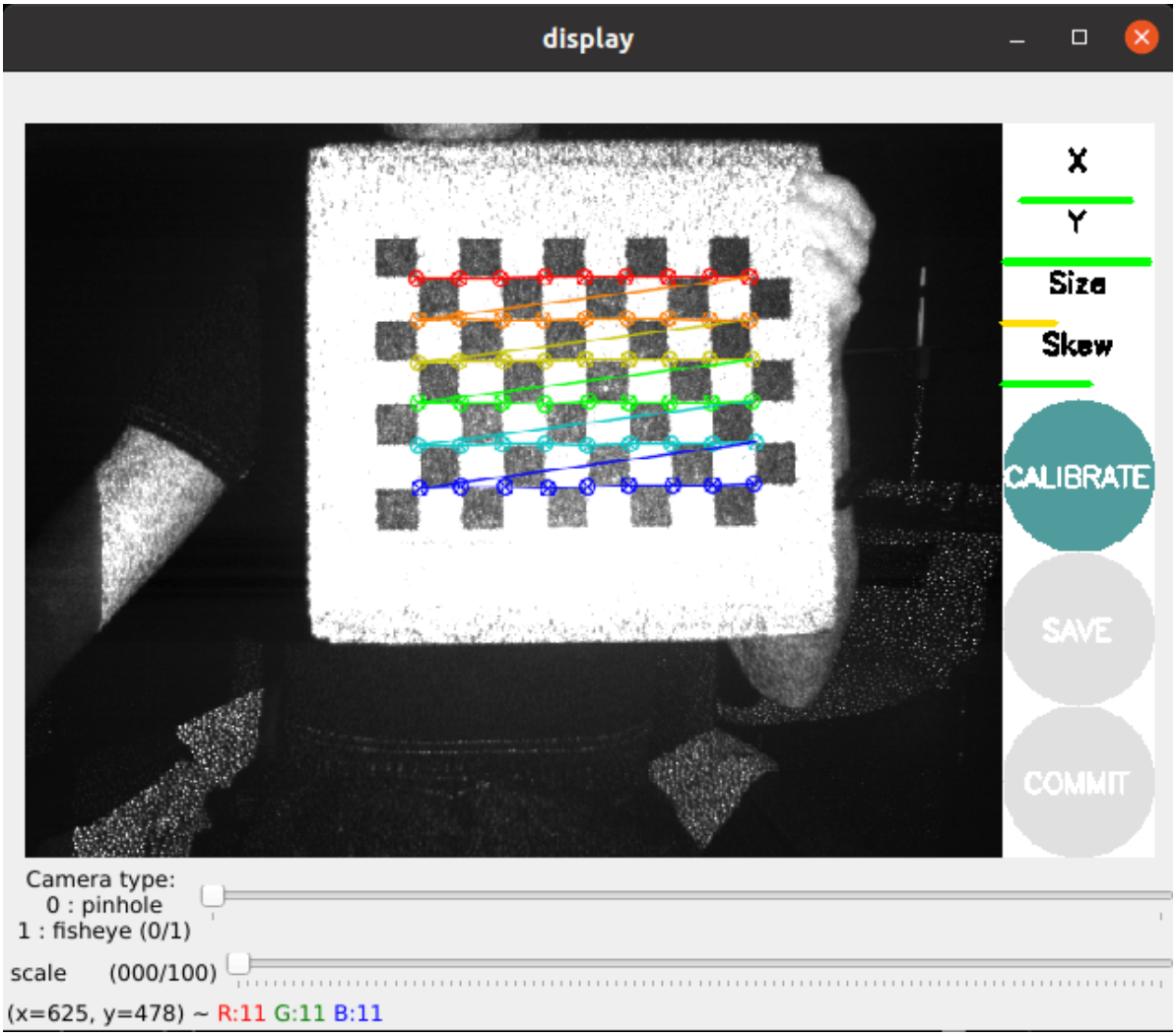
Choose the treatment method according to your needs.

Enter in the VM terminal,

```
ros2 run camera_calibration cameracalibrator --size 9x6 --square 0.02 --ros-args
--remap /image:=/camera/ir/image_raw
```




Similarly, the following operations are similar to color camera calibration, changing different poses, as far as possible to fill the lines under [X], [Y], [Size], [Skew].



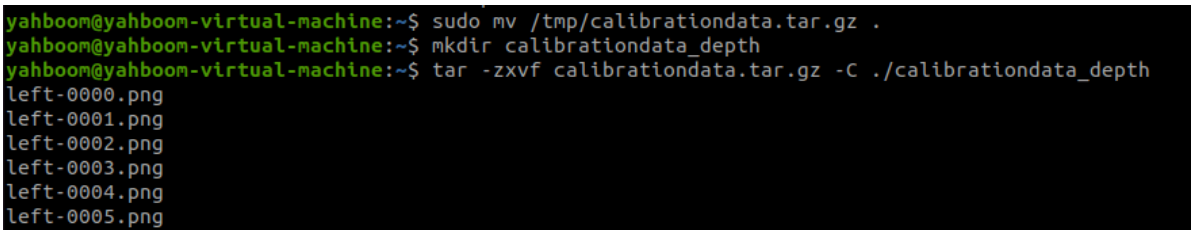

Click [CALIBRATE] to calculate camera parameters, and click [SAVE], as shown in the picture below.

```
**** Calibrating ****
mono pinhole calibration...
D =  [-0.057562825540741026, 0.1572966484578396, 0.001232642527870728, -0.0014690785306377375, 0.0]
K =  [584.2563559141299, 0.0, 324.8602354629445, 0.0, 583.2777778629661, 245.8207280691308, 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 =  [587.6276245117188, 0.0, 323.28640646098575, 0.0, 0.0, 587.4089965820312, 245.6971317811949, 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
584.256356 0.000000 324.860235
0.000000 583.277778 245.820728
0.000000 0.000000 1.000000

distortion
-0.057563 0.157297 0.001233 -0.001469 0.000000

rectification
1.000000 0.000000 0.000000
0.000000 1.000000 0.000000
0.000000 0.000000 1.000000

projection
587.627625 0.000000 323.286406 0.000000
0.000000 587.408997 245.697132 0.000000
0.000000 0.000000 1.000000 0.000000


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

Calibration results to the/TMP/calibrationdata. Tar. Gz 】, can move the file out to check the content.

After exiting the program, enter the virtual machine terminal,

```
#文件移动至当前目录
# file moved to the current directory
sudo mv /tmp/calibrationdata.tar.gz .
#创建文件夹
# Create folder
mkdir calibrationdata_depth
#解压至指定文件夹
# Unzip to the specified folder
tar -zxvf calibrationdata.tar.gz -C ./calibrationdata_depth
```




You will get the calibrated png file, ost.yaml, and ost.txt files in the calibrationdata_depth folder in this directory.