Camera calibration

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

Ordinary camera： https://github.com/bosch-ros-pkg/usb_cam.git

Monocular teaching： http://wiki.ros.org/camera_calibration/Tutorials/MonocularCalibration

Due to some internal and external reasons of the camera, the image will be greatly distorted, mainly radial deformation and tangential deformation, causing the straight line to become curved. The farther the pixel is from the center of the image, the more serious the distortion will be. In order to avoid errors caused by data sources, the parameters of the camera need to be calibrated. Calibration essentially uses a known and determined spatial relationship (calibration plate) to reversely deduce the inherent and real parameters of the camera (internal parameters) by analyzing the pixels of the photographed pictures.

1、Preparation before calibration

- A large chessboard of known dimensions. This tutorial uses a 9x6 checkerboard and a 20mm square, which needs to be flattened during calibration. **The calibration uses the internal vertices of the checkerboard, so a "10x7" checkerboard uses the internal vertex parameters "9x6", as shown in the example below.** Calibration boards of any specifications are acceptable, just change the parameters. An open area without obstacles or calibration board patterns
- Monocular camera publishing images via ROS

2、Start calibration

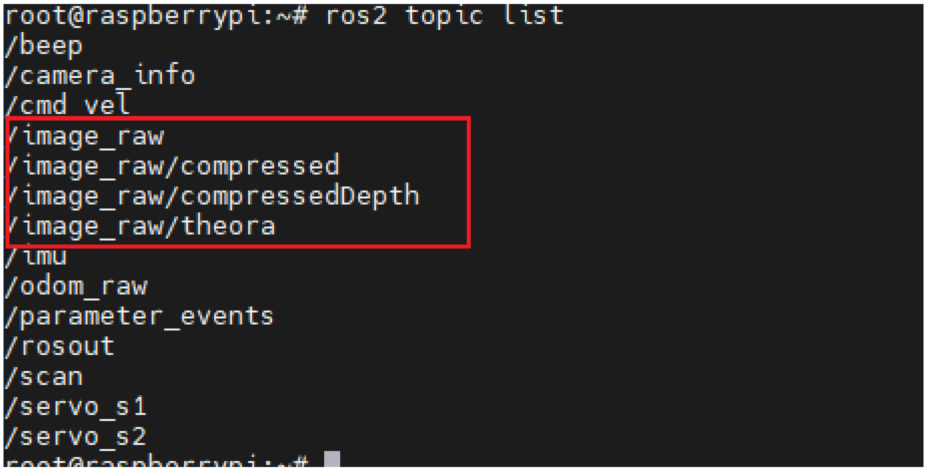
Install the calibrated function package camera_calibration and enter it in the docker terminal **(the factory docker is already installed and does not need to be reinstalled)**

```
sudo apt install ros-humble-camera-calibration*
```

Start the camera before calibration, and then turn off the camera until all calibrations are completed. Start the camera and enter it in the Docker terminal.

```
ros2 run usb_cam  usb_cam_node_exe
```

Use the following command to view the topic, enter in the Docker terminal.



```
root@raspberrypi:~# ros2 topic list
/beep
/camera_info
/cmd_vel
/image_raw
/image_raw/compressed
/image_raw/compressedDepth
/image_raw/theora
/imu
/odom_raw
/parameter_events
/rosout
/scan
/servo_s1
/servo_s2
root@raspberrypi:~#
```

The topic we need to use to calibrate RGB color images is /image_raw.

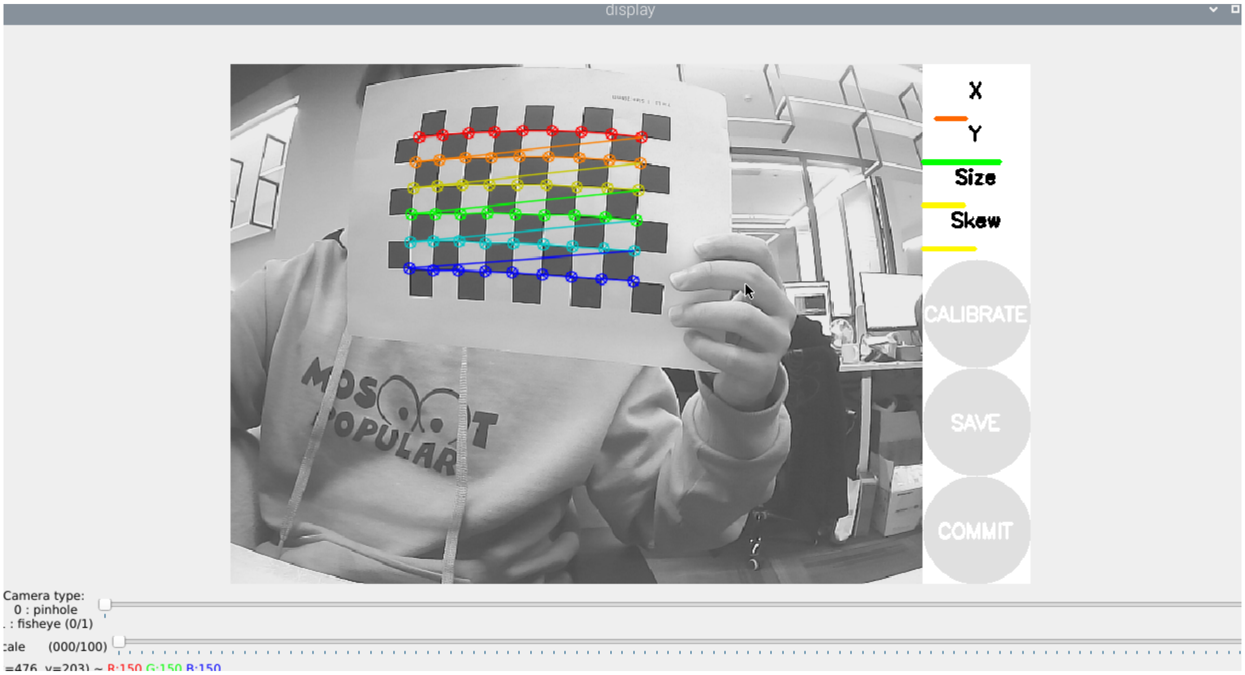
Run the calibrated program and enter in the Docker terminal.

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

size: Calibrate the number of internal corner points of the checkerboard, for example, 9X6, with a total of six rows and nine columns of corner points.

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

Topic name: /image_raw



Camera type:
0 : pinhole
. : fisheye (0/1)
:ale (000/100)
=476 y=203) ~ R:150 G:150 B:150

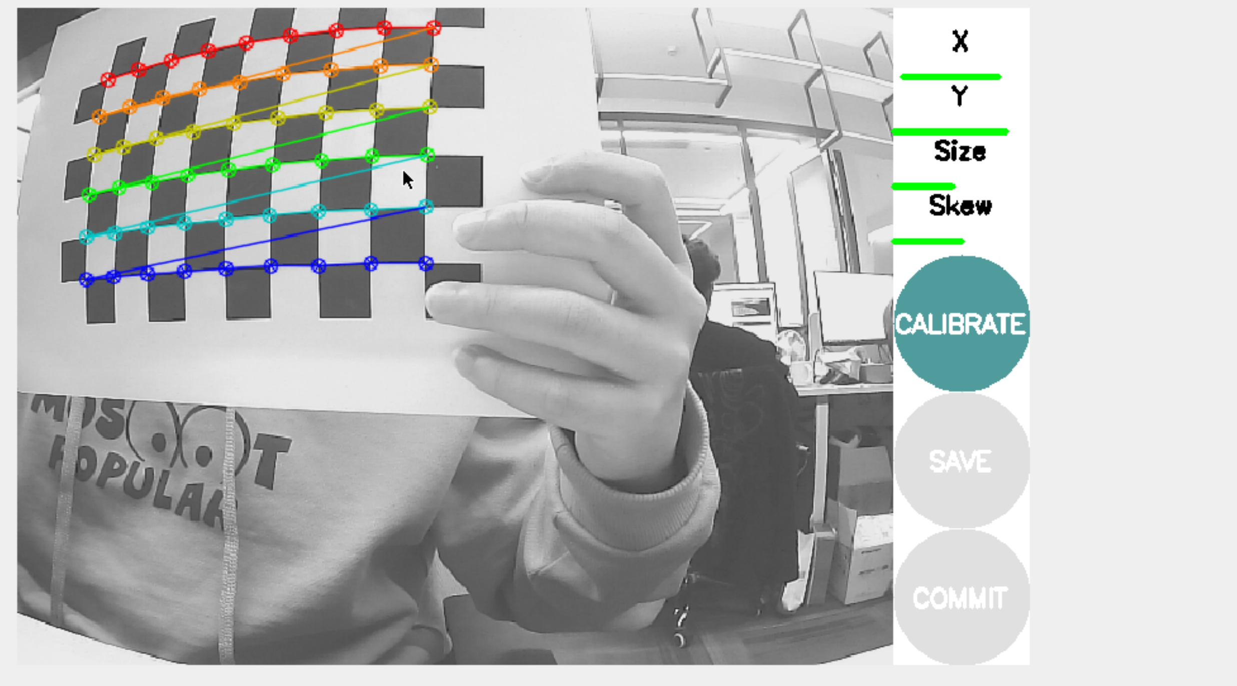
X: The checkerboard moves left and right in the camera field of view

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

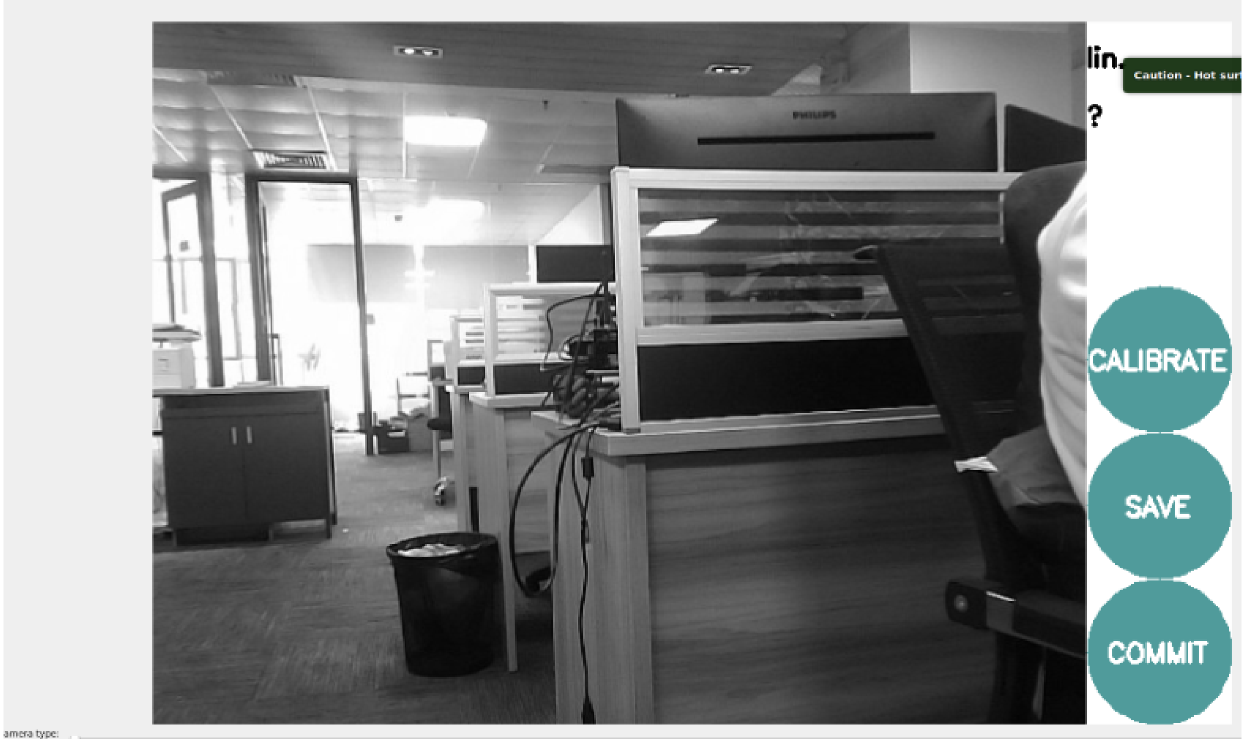
Size : The checkerboard moves forward and backward in the camera's field of view

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

As shown in the picture above, you need to collect the image by flipping it up and down, front and back, left and right, so that the X, Y, Size and Skew on the right turn green, as shown in the picture below, then click CALIBRATE to start calibration.



After the calibration is completed, click SAVE, as shown in the figure below.



The calibration results are saved to [/tmp/calibrationdata.tar.gz], and the saving path is the terminal directory where the calibration program is started. After the calibration is completed, you can move out the [/tmp/calibrationdata.tar.gz] file to see the content.

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

docker terminal input.

```
cd ~
tar -xvf calibrationdata.tar.gz
```

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

When starting the USB camera, you need to load parameters. Therefore, after calibration, you need to replace the parameters with the original built-in parameters. Rename the calibrated ost.yaml to camera_info.yaml, then replace the original camera_info.yaml, and enter it in the docker terminal.

```
#First copy the file to /opt/ros/foxy/share/usb_cam/config
sudo cp ost.yaml /opt/ros/foxy/share/usb_cam/config
#Switch to the /opt/ros/foxy/share/usb_cam/config directory
cd /opt/ros/foxy/share/usb_cam/config
#Back up the original camera_info.yaml
sudo mv camera_info.yaml camera_info_BK.yaml
#Rename ost.yaml to camera_info.yaml
sudo mv ost.yaml camera_info.yaml
```