

调试 LenaCV 3D Camera (Linux)

胡耀钰

<yyhu_live@outlook.com>

2018年5月26日于Carnegie Mellon University

1 测试环境

Ubuntu 16.04 LTS, Python 2.7.12, ROS Lunar

本文中“>>>”表示在Linux终端中输入命令，某些需要sudo权限的命令可能没有写出sudo。

本文使用的双目相机硬件是LenaCV的产品，该产品可以从中国大陆的淘宝网上采购到。

<https://item.taobao.com/item.htm?spm=a1z10.1-c-s.w4004-17461658044.5.6c782004hVg4HU&id=562303637855>


本文由Google Docs在线生成，可能出现某些无法调整的格式问题。

2 基本测试

相机通过USB连接系统后，在Linux中识别该设备

```
>>> lsusb
```

显示结果如图1所示，其中Device 008即为当前调试的设备。



```
yyhu@airlab: ~  
yyhu@airlab:~$ lsusb  
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub  
Bus 001 Device 004: ID 046d:c31c Logitech, Inc. Keyboard K120  
Bus 001 Device 003: ID 0bda:0129 Realtek Semiconductor Corp. RTS5129 Card Reader Controller  
Bus 001 Device 008: ID 1e4e:0568 Cubeternet  
Bus 001 Device 006: ID 046d:c52b Logitech, Inc. Unifying Receiver  
Bus 001 Device 005: ID 046d:c016 Logitech, Inc. Optical Wheel Mouse  
Bus 001 Device 002: ID 0cf3:e300 Atheros Communications, Inc.  
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub  
yyhu@airlab:~$
```

图1 lsusb结果

安装GTK UVC video viewer. 在接上相机后，运行UVC video viewer，所得画面如图2所示。通过画面上几个控制滑块控制相机参数以进行调试和测试。LenaCV相机返回的实时画面是一张图，相机内部应当已经完成了同步，并将两个相机的图像进行了拼接。经过实测发现拼接图像的右侧实际上为左摄像头的画面。

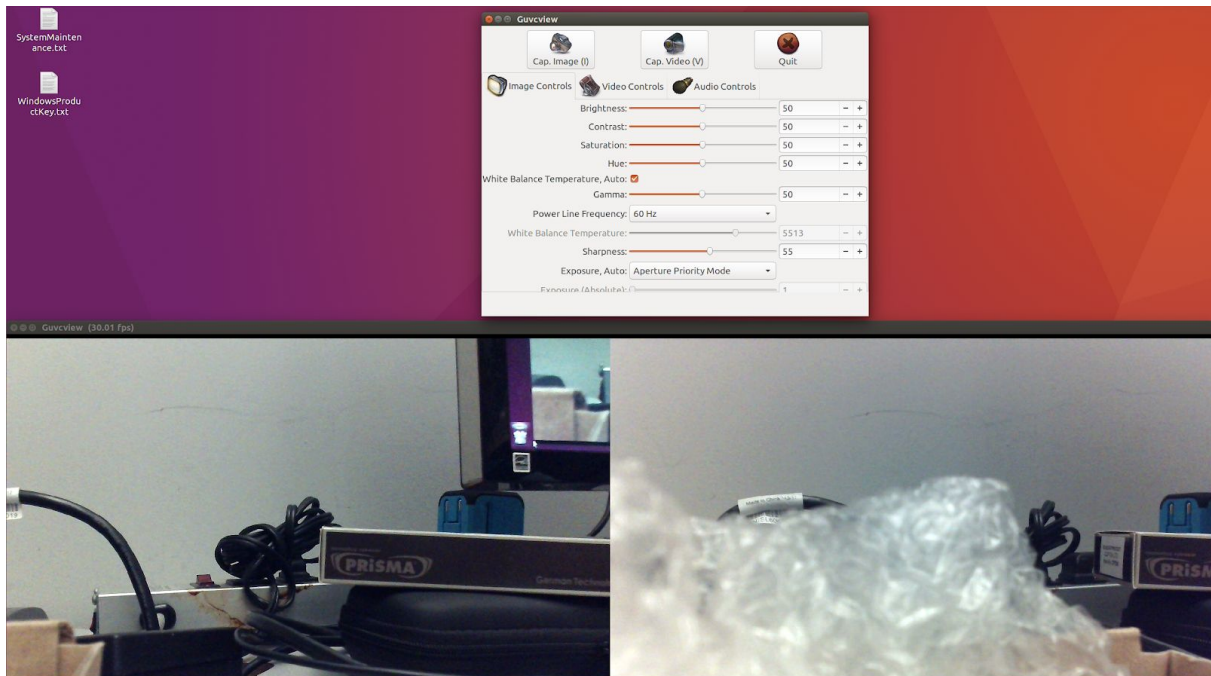


图2 使用GTK UVC video viewer直接调试设备

3 ROS libuvc_camera调试

主要调试过程参考了libuvc_camera的官方wiki

http://wiki.ros.org/libuvc_camera

3.1 安装libuvc_camera

首先需要确保系统正常安装了ROS，并且ROS处于可以使用状态。

创建一个catkin workspace或者使用已经存在的catkin workspace。切换到catkin workspace的src文件夹，将libuvc_camera的git仓库clone到本地。

```
>>> git clone https://github.com/ros-drivers/libuvc_ros.git
```

返回到src文件夹上一级，启动catkin_make进行编译，若报缺少某些库，则直接利用apt-get进行安装即可。

3.2 修改udev rule

根据libuvc_camera官方wiki的描述，使用uvc设备需要拥有一定权限，推荐的做法是针对需要使用的硬件设备添加udev rule。具体做法如下，在/etc/udev/rules.d文件夹下新建文件99-uvc.rules（需要root权限）。这个文件将描述相机的所有者，将所有者设置为当前用户即可。该文件的文件名一般都是以数字开头，其目的是控制文件之间的排序。数字越小，排序越

靠前。这样做的道理是Linux系统将在加载硬件驱动的过程中有时会有一些次序要求，用文件名的次序可以控制这些rule加载的先后。99-udev.rules的内容如table 1所示。

表1 99-udev.rules

```
# UVC cameras
SUBSYSTEMS=="usb", ENV{DEVTYPE}=="usb_device", ATTRS{idVendor}=="1e4e",
ATTRS{idProduct}=="0568", OWNER="<用户名>"
```

其中OWNER即为需要设置的用户名，这里将其设置为当前用户，idVendor和idProduct直接复制lsusb返回结果中的数值。保存该文件，此后需要将相机重新插拔一次。此后libuvc_camera若不能正常launch，ROS报权限不够错误，处理方法参考3.3节。

3.3 测试libuvc_camera

libuvc_camera定义了一个ros node，名为camera_node，启动这个node一般通过一个launch文件。本测试使用的launch文件内容如table 2所示。

表2 launch file

```
<launch>
<group ns="camera">
  <node pkg="libuvc_camera" type="camera_node" name="mycam">
    <!-- Parameters used to find the camera -->
    <param name="vendor" value="0x1e4e"/>
    <param name="product" value="0x0568"/>
    <param name="serial" value=""/>
    <!-- If the above parameters aren't unique, choose the first match: -->
    <param name="index" value="0"/>

    <!-- Image size and type -->
    <param name="width" value="2560"/>
    <param name="height" value="720"/>
    <!-- choose whichever uncompressed format the camera supports: -->
    <param name="video_mode" value="mjpeg"/> <!-- or uncompressed/yuyv/nv12/mjpeg -->
    <param name="frame_rate" value="30"/>

    <param name="timestamp_method" value="start"/> <!-- start of frame -->
    <param name="camera_info_url" value="file:///tmp/cam.yaml"/>

    <param name="brightness" value="50" />

    <param name="auto_exposure" value="3"/> <!-- use aperture_priority auto exposure -->
    <param name="auto_white_balance" value="true"/>
  </node>
</group>
</launch>
```

如table 2所示，需要定义几个参数来控制相机，其中width、height、video_mode和frame_rate需要通过v4l2-ctl命令参看，v4l2-ctl需要通过apt-get安装。安装好v4l2-ctl后，执行

```
>>> v4l2-ctl --list-formats-ext
```

执行结果如图3所示。从结果上看相机支持的video_mode包括MJPG，在配置launch file时，需要指定video_mode为mjpeg。图像大小和帧率也需根据图3中的结果设置。

```
yyhu@airlab:~/ROS/p2/catkin/launch/libuvc$ v4l2-ctl --list-formats-ext
ioctl: VIDIOC_ENUM_FMT
  Index       : 0
  Type        : Video Capture
  Pixel Format : 'YUYV'
  Name        : YUYV 4:2:2
                Size: Discrete 2560x720
                  Interval: Discrete 0.200s (5.000 fps)
                Size: Discrete 1280x480
                  Interval: Discrete 0.067s (15.000 fps)
                  Interval: Discrete 0.200s (5.000 fps)

  Index       : 1
  Type        : Video Capture
  Pixel Format : 'MJPG' (compressed)
  Name        : Motion-JPEG
                Size: Discrete 2560x720
                  Interval: Discrete 0.033s (30.000 fps)
                Size: Discrete 1280x480
                  Interval: Discrete 0.033s (30.000 fps)

yyhu@airlab:~/ROS/p2/catkin/launch/libuvc$
```

图3 v4l2-ctl执行结果

启动launch file。

```
>>> roslaunch <launch file>
```

首次启动可能由于权限问题而失败，出现如图4所示画面。此时通过如下命令进行修正

```
>>> sudo chmod o+w /dev/bus/usb/003/004
```

其中 003/004 是中报错信息中的部分，请用户根据自己的报错信息进行修改。

```
yaoyu@yyhu-live: ~/catkin_ws/src/libuvc_ros 80x33
* /camera/mycam/camera_info_url: file:///tmp/cam.yaml
* /camera/mycam/frame_rate: 30
* /camera/mycam/height: 720
* /camera/mycam/index: 0
* /camera/mycam/product: 0x0568
* /camera/mycam/serial:
* /camera/mycam/timestamp_method: start
* /camera/mycam/vendor: 0x1e4e
* /camera/mycam/video_mode: mjpeg
* /camera/mycam/width: 2560
* /rostdistro: lunar
* /rosversion: 1.13.6

NODES
  /camera/
    mycam (libuvc_camera/camera_node)

auto-starting new master
process[master]: started with pid [4099]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to c24b8f5e-62a3-11e8-9abd-142d27437817
process[rosout-1]: started with pid [4112]
started core service [/rosout]
process[camera/mycam-2]: started with pid [4121]
[ERROR] [1527531612.020372547]: Permission denied opening /dev/bus/usb/003/004
[camera/mycam-2] process has died [pid 4121, exit code 255, cmd /home/yaoyu/catkin_ws/devel/lib/libuvc_camera/camera_node __name:=mycam __log:=/home/yaoyu/.ros/log/c24b8f5e-62a3-11e8-9abd-142d27437817/camera-mycam-2.log].
log file: /home/yaoyu/.ros/log/c24b8f5e-62a3-11e8-9abd-142d27437817/camera-mycam-2*.log
```

图4 权限问题导致启动失败

正确启动时，终端中显示的内容如table 3所示。

表3 libuvc_camera启动消息

```
... logging to
/home/yyhu/.ros/log/2183dccc-5ee1-11e8-8cb9-5800e3fc52bd/roslaunch-airlab-14474.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://airlab:41307/

SUMMARY
=====

PARAMETERS
* /camera/mycam/auto_exposure: 3
* /camera/mycam/auto_white_balance: True
* /camera/mycam/brightness: 50
* /camera/mycam/camera_info_url: file:///tmp/cam.yaml
* /camera/mycam/frame_rate: 30
* /camera/mycam/height: 720
* /camera/mycam/index: 0
* /camera/mycam/product: 0x0568
* /camera/mycam/serial:
```



```
* /camera/mycam/timestamp_method: start
* /camera/mycam/vendor: 0x1e4e
* /camera/mycam/video_mode: mjpeg
* /camera/mycam/width: 2560
* /roscistro: lunar
* /rosversion: 1.13.6

NODES
/camera/
  mycam (libuvc_camera/camera_node)

auto-starting new master
process[master]: started with pid [14484]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to 2183dccc-5ee1-11e8-8cb9-5800e3fc52bd
process[rosout-1]: started with pid [14497]
started core service [/rosout]
process[camera/mycam-2]: started with pid [14500]
unsupported descriptor subtype VS_COLORFORMAT
unsupported descriptor subtype VS_COLORFORMAT
unsupported descriptor subtype VS_COLORFORMAT
attempt to claim already-claimed interface 1
Not a JPEG file: starts with 0x68 0x26
Couldn't convert frame to RGB: Unknown error (-99)
```

正确启动了libuvc_camera后，可以通过ROS的rqt_image_view查看图像，所用到的topic为/camera/image_raw。rqt_image_view的查看结果如图5所示。

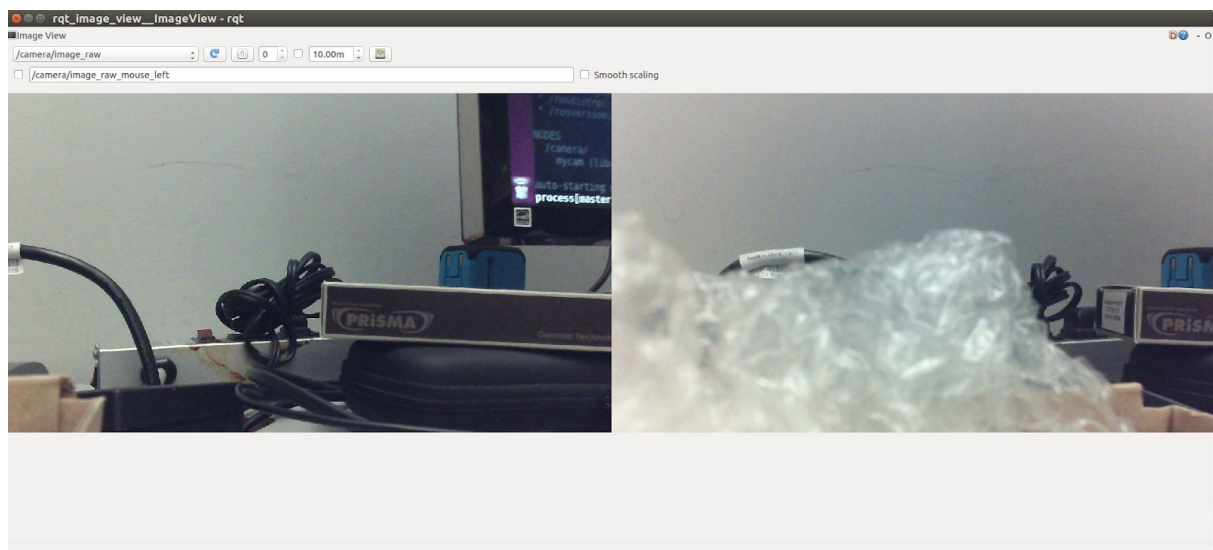


图5 rqt_image_view的查看结果

4 在线双目标定

在线双目标定将利用ROS提供的camera_calibration包实现。

4.1 创建专用publisher

为了能够使用ROS的camera_calibration包，需要将相机的画面从一幅图像拆分为两幅图像，分别对应双目摄像机的两个相机，并且各自发布一个topic。我们需要一个node，订阅libuvc_camera的topic，实时将画面一分为二，并发布两个新的topic。这里将创建一个新的专用publisher完成以上工作。

首先进入catkin workspace的src目录，创建一个新的package，本文中使用的package名称为“lenacv”

```
>>> catkin_create_pkg lenacv std_msgs rospy roscpp
```

之后返回到src上一级，使用catkin_make进行初步编译，生成必要的文件。重新执行devel/setup.bash。

```
>>> source devel/setup.bash
```

重新回到lenacv的目录，创建scripts目录并进入。编写一个新的ROS node的python文件，内容如table 4所示。这个node名为listen_uvc，并将publish两个新的topic，分别为lenacv_left和lenacv_right。该文件命名为separator.py。

表4 ROS node

请参考Github仓库的最新版本
<https://github.com/huyaoyu/catkins/blob/master/lenacv/scripts/separator.py>

修改ROS node源码文件的执行权限。

```
>>> chmod +x separator.py
```

在外层目录重新执行catkin_make。启动libuvc_camera，之后通过

```
>>> rosrn lenacv separator.py
```

来启动刚刚创建的listen_uvc node。启动ROS rqt，添加两个image view的plugin，并各自显示来自lenacv_left和lenacv_right的数据，效果如图6所示。

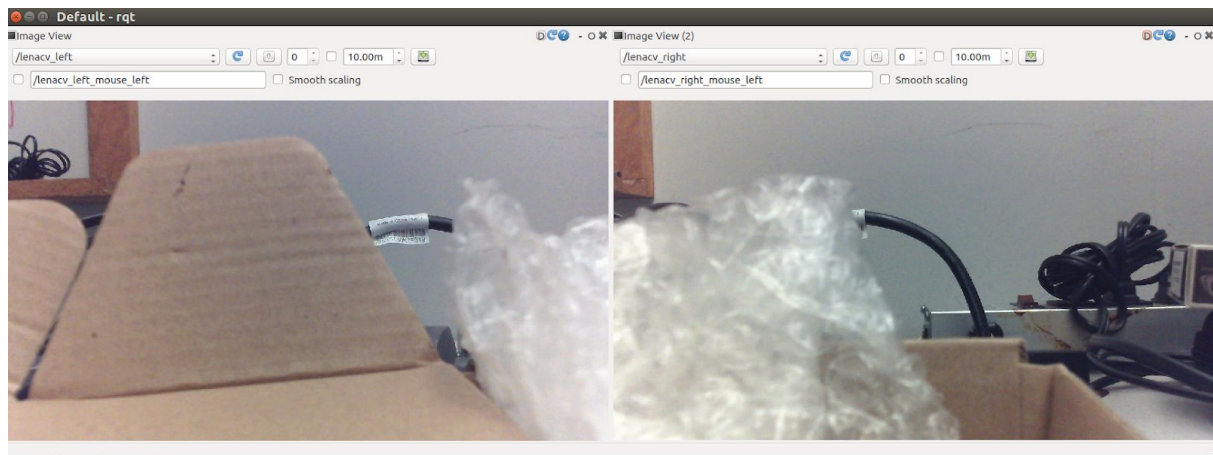


图6 拆分后的双目图像

4.2 双目标定

本节主要参考

http://wiki.ros.org/camera_calibration/Tutorials/StereoCalibration

双目标定使用ROS提供的camera_calibration包。在进行双目标定前，先打印棋盘格标定板。将打印好的标定板平整地固定在某一平面物体表面，利用直尺测量标定板上棋盘格的边长。标定时启动libuvc_camera，启动lenacv_camera的separator.py，最后通过下述命令启动camera_calibration。

```
>>> rosruncamera_calibration cameracalibrator.py --size 8x6 --square 0.0345
right:=/lenacv_camera/right/image_raw left:=/lenacv_camera/left/image_raw
right_camera:=/lenacv_camera/right left_camera:=/lenacv_camera/left --no-service-check
```

此处--size 参数为所用棋盘格的交叉点数量和排布，如图7所示，圆点所在位置为交叉点。--size参数的第一个数字为交叉点的列数。请确保棋盘格的两个边长方向上的交叉点的数量是不同的，并且保持长边处于水平。--square 参数为棋盘格的边长，单位为m。请确保使用正四边形的棋盘格。

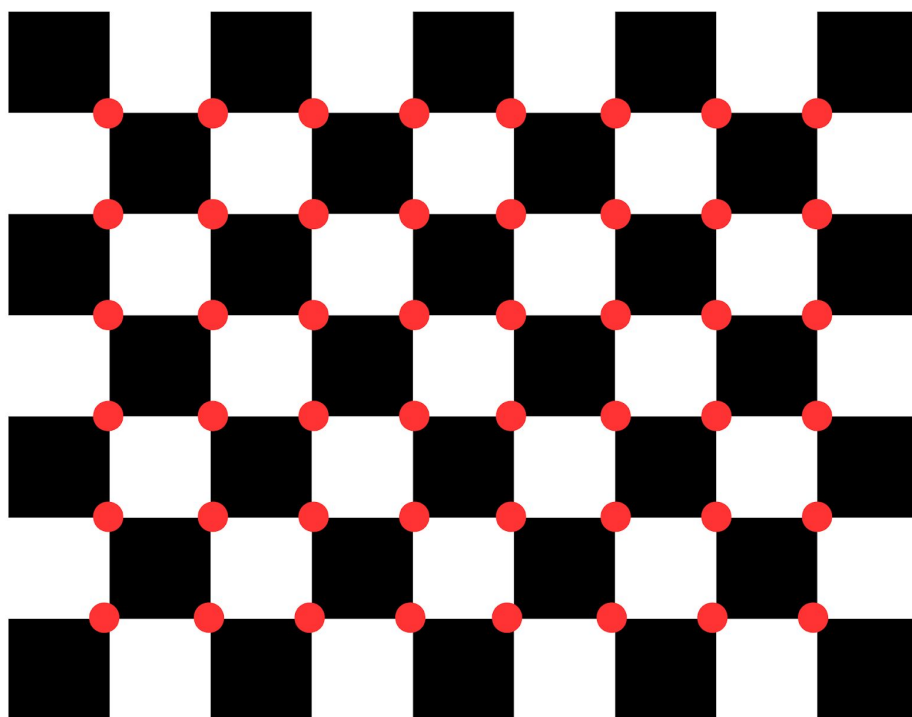


图7 棋盘格

启动后将看到双目的实时灰度图像。通过鼠标调整程序窗口的大小以获得比较清晰的视角。将棋盘格至于双目相机的视野内，确保两个相机都能同时捕捉到清晰完整的棋盘格图像。cameracalibrator.py将自动识别棋盘格，并在识别到第一个棋盘格后，出现“X”、“Y”、“size”和“skew”字样。这些字样下方均有一个类似进度条的标志。移动棋盘格，cameracalibrator.py将自动辨识棋盘格，并在灰度画面上叠加识别到的棋盘格交叉点的彩色图像，如图8所示。

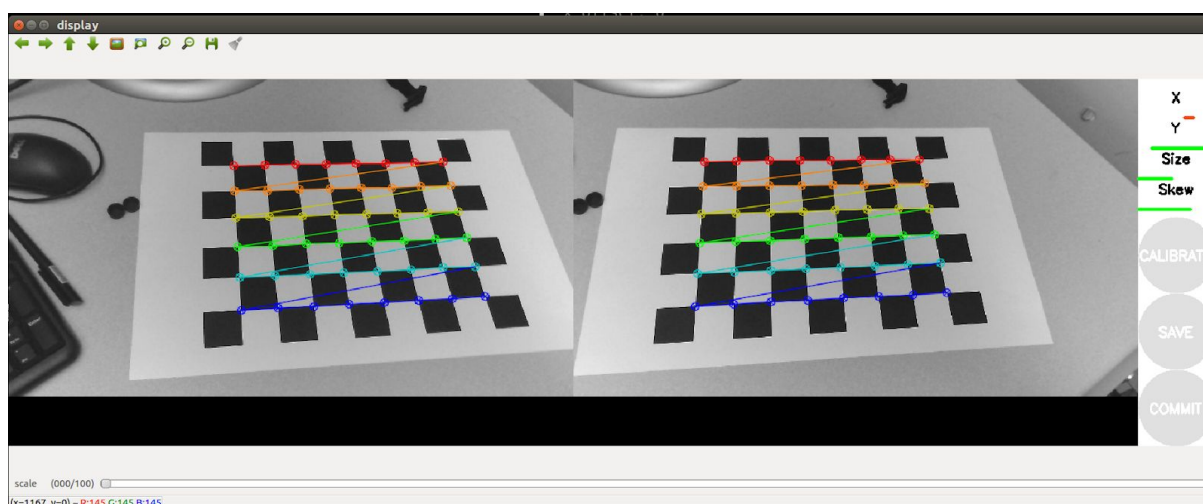


图8 标定过程中实时识别到的棋盘格

当棋盘格的识别和当前棋盘格的位置符合要求时，cameracalibrator.py有可能将其记录进最终标定数据集。一个棋盘格的输入是否进入最终标定数据集，也取决于棋盘格与相机相对位置在标定数据集中是否已经有类似记录。cameracalibrator.py会试图使记录的数据尽可能的

适用于标定“X”、“Y”、“size”和“skew”并得到更好的效果，它会主动控制数据集数量和范围。每收集到一组棋盘格的有效数据后，终端上会出现提示，如图9所示，并且上述“X”、“Y”、“size”和“skew”所对应的进度条也会相应增长。

```
^Cyyhugairlab:~/R05/p2/catkin/src/dummy_stereo/scripts$ rosruncamera calibration cameracalibrator.py --size 8x6 --square 0.0345 right
/lenacv_camera/right/image_raw left:=/lenacv_camera/left/image_raw left_camera:=/lenacv_camera/left right_camera:=/lenacv_camera/right
--no-service-check
*** Added sample 1, p_x = 0.694, p_y = 0.387, p_size = 0.332, skew = 0.030
*** Added sample 2, p_x = 0.710, p_y = 0.223, p_size = 0.338, skew = 0.006
*** Added sample 3, p_x = 0.888, p_y = 0.231, p_size = 0.347, skew = 0.017
*** Added sample 4, p_x = 0.586, p_y = 0.295, p_size = 0.337, skew = 0.000
*** Added sample 5, p_x = 0.453, p_y = 0.263, p_size = 0.345, skew = 0.037
*** Added sample 6, p_x = 0.748, p_y = 0.362, p_size = 0.442, skew = 0.011
*** Added sample 7, p_x = 0.701, p_y = 0.260, p_size = 0.496, skew = 0.012
*** Added sample 8, p_x = 0.707, p_y = 0.451, p_size = 0.507, skew = 0.001
*** Added sample 9, p_x = 0.759, p_y = 0.563, p_size = 0.534, skew = 0.015
*** Added sample 10, p_x = 0.674, p_y = 0.662, p_size = 0.541, skew = 0.004
*** Added sample 11, p_x = 0.608, p_y = 0.580, p_size = 0.491, skew = 0.018
*** Added sample 12, p_x = 0.547, p_y = 0.654, p_size = 0.466, skew = 0.062
*** Added sample 13, p_x = 0.580, p_y = 0.837, p_size = 0.459, skew = 0.087
*** Added sample 14, p_x = 0.657, p_y = 0.500, p_size = 0.451, skew = 0.052
*** Added sample 15, p_x = 0.665, p_y = 0.709, p_size = 0.437, skew = 0.064
*** Added sample 16, p_x = 0.778, p_y = 0.397, p_size = 0.394, skew = 0.108
*** Added sample 17, p_x = 0.816, p_y = 0.271, p_size = 0.402, skew = 0.158
*** Added sample 18, p_x = 0.866, p_y = 0.382, p_size = 0.404, skew = 0.199
*** Added sample 19, p_x = 0.893, p_y = 0.697, p_size = 0.400, skew = 0.301
*** Added sample 20, p_x = 0.787, p_y = 0.663, p_size = 0.396, skew = 0.234
*** Added sample 21, p_x = 0.734, p_y = 0.618, p_size = 0.445, skew = 0.295
*** Added sample 22, p_x = 0.798, p_y = 0.802, p_size = 0.446, skew = 0.203
*** Added sample 23, p_x = 0.832, p_y = 1.000, p_size = 0.440, skew = 0.092
*** Added sample 24, p_x = 0.762, p_y = 0.913, p_size = 0.437, skew = 0.027
*** Added sample 25, p_x = 0.774, p_y = 0.680, p_size = 0.438, skew = 0.003
*** Added sample 26, p_x = 0.608, p_y = 0.638, p_size = 0.421, skew = 0.141
*** Added sample 27, p_x = 0.616, p_y = 0.463, p_size = 0.411, skew = 0.154
*** Added sample 28, p_x = 0.644, p_y = 0.297, p_size = 0.397, skew = 0.170
*** Added sample 29, p_x = 0.652, p_y = 0.114, p_size = 0.398, skew = 0.135
*** Added sample 30, p_x = 0.710, p_y = 0.000, p_size = 0.410, skew = 0.156
*** Added sample 31, p_x = 0.718, p_y = 0.146, p_size = 0.419, skew = 0.045
*** Added sample 32, p_x = 0.860, p_y = 0.308, p_size = 0.432, skew = 0.056
*** Added sample 33, p_x = 0.893, p_y = 0.479, p_size = 0.435, skew = 0.119
*** Added sample 34, p_x = 0.832, p_y = 0.568, p_size = 0.422, skew = 0.069
*** Added sample 35, p_x = 0.717, p_y = 0.626, p_size = 0.372, skew = 0.035
*** Added sample 36, p_x = 0.804, p_y = 0.515, p_size = 0.407, skew = 0.185
*** Added sample 37, p_x = 0.536, p_y = 0.495, p_size = 0.376, skew = 0.080
*** Added sample 38, p_x = 0.493, p_y = 0.420, p_size = 0.419, skew = 0.019
*** Added sample 39, p_x = 0.367, p_y = 0.441, p_size = 0.378, skew = 0.074
*** Added sample 40, p_x = 0.317, p_y = 0.333, p_size = 0.372, skew = 0.121
*** Added sample 41, p_x = 0.424, p_y = 0.525, p_size = 0.337, skew = 0.106
*** Added sample 42, p_x = 0.572, p_y = 0.714, p_size = 0.340, skew = 0.097
*** Added sample 43, p_x = 0.575, p_y = 0.881, p_size = 0.343, skew = 0.131
```

图9 收集棋盘格数据

如图10所示，当数据集收集到了足够的数据，界面上的“Calibrate”按钮将进入使能状态，此时用鼠标点击该按钮开始标定。

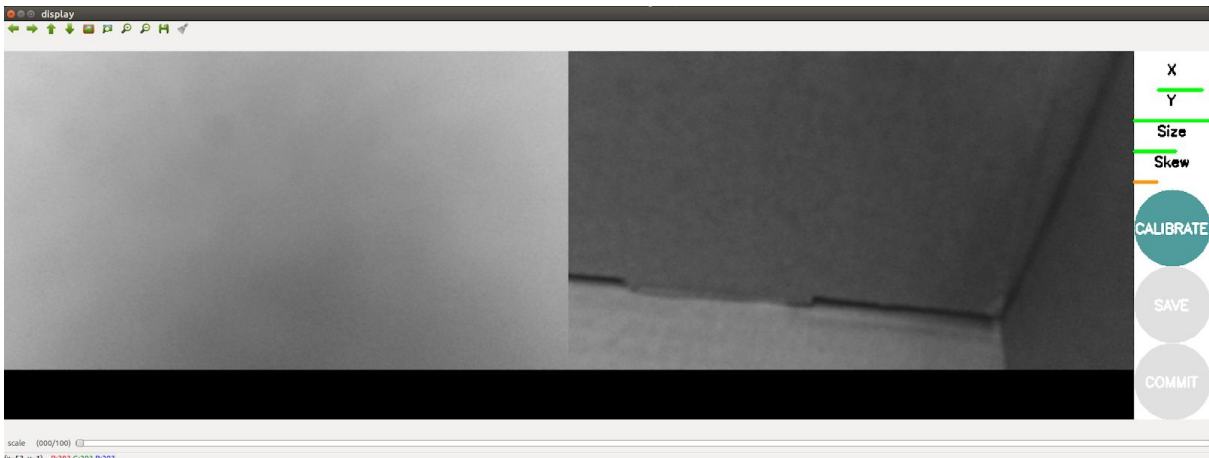


图10 使能状态的calibrate按钮

标定可能持续一定时间，此时软件界面可能失去相应，需耐心等待。当标定结束后，终端中将显示标定结果。点击软件界面上的“save”按钮，将标定结果（以及标定数据集）记录

在文件系统上。此后可手工复制上述保存好的文件到其他指定位置。注意，保存的数据仅包含两个相机各自的intrinsics，extrinsics仅在屏幕上做出显示，他们所在的位置如图11所示。手工复制extrinsics，并记录在文件上以留后期使用。

```
*** Added sample 39, p_x = 0.367, p_y = 0.441, p_size = 0.378, skew = 0.074
*** Added sample 40, p_x = 0.317, p_y = 0.333, p_size = 0.372, skew = 0.121
*** Added sample 41, p_x = 0.424, p_y = 0.525, p_size = 0.337, skew = 0.106
*** Added sample 42, p_x = 0.572, p_y = 0.714, p_size = 0.340, skew = 0.097
*** Added sample 43, p_x = 0.575, p_y = 0.881, p_size = 0.343, skew = 0.131

Left:
('D = ', [0.10209387223207614, -0.1981882162580508, 0.0015697479717236911, 1.986841183686835e-06, 0.0])
('K = ', [1234.1633417056237, 0.0, 652.3015347728342, 0.0, 1236.068000312867, 424.8613753339475, 0.0, 0.0, 1.0])
('R = ', [0.9990772336971965, 0.014950825850082753, -0.040263555659648655, -0.015019414143529114, 0.9998862200038875, -0.0014015152951452673, 0.04023802066133829, 0.0020049570414013163, 0.9991881113386607])
('P = ', [1332.8846743066233, 0.0, 712.0629425048828, 0.0, 0.0, 1332.8846743066233, 424.57877349853516, 0.0, 0.0, 0.0, 1.0, 0.0])

Right:
('D = ', [0.06495075350429823, -0.03315183518506859, 0.0034632862936945603, -0.0001215708583502371, 0.0])
('K = ', [1241.8146063512547, 0.0, 653.6096169411022, 0.0, 1244.8903947218366, 424.8376197396223, 0.0, 0.0, 1.0])
('R = ', [0.999250893493945, 0.01748600671473932, -0.03452378051031291, -0.017427154590540443, 0.9998461255173933, 0.0020048867136073234, 0.0345535256439857, -0.001401733579923173, 0.9994018656219035])
('P = ', [1332.8846743066233, 0.0, 712.0629425048828, -134.28847779396088, 0.0, 1332.8846743066233, 424.57877349853516, 0.0, 0.0, 0.0, 1.0, 0.0])
('self.T ', [-0.10067478755531704, -0.0017617197269065713, 0.0034782798705632234])
('self.R ', [0.999980929572306, -0.002416267305259604, -0.005683497516163046, 0.0023963651932862487, 0.999990982854713, -0.003505943970671137, 0.005691917565030627, 0.003492257375196254, 0.9999777028578478])
none
# oST version 5.0 parameters

[image]

width
1280

height
720

[narrow_stereo/left]

camera matrix
1234.163342 0.000000 652.301535
0.000000 1236.068000 424.861375
0.000000 0.000000 1.000000
```

图11 标定得到的extrinsics