**一、环境配置**

### **1.Ubuntu16.04安装**

安装教程参考：<https://blog.csdn.net/trackxiaoxin321/article/details/115591796>

**2.Ubuntu16.04下的ROS安装**

1）设置sources.list

sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb\_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'

1. 设置key

sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv 76F1A20FF987672F

1. 更新package

sudo apt-get update

1. 安装ROS kinetic

sudo apt-get install ros-kinetic-desktop-full

1. 初始化rosdep

sudo rosdep init

rosdep update

这步有很大概率会出现问题，大多数问题都集中在第二步，如果是第一步报错，可以试着再重新执行一下，可能就会成功。第二步往往会会报出：the read operation is timed out  
此时，依次执行如下命令：

sudo apt-get install python3-pip

sudo pip3 install 6-rosdep

sudo 6-rosdep

1. 配置ros环境

echo "source /opt/ros/kinetic/setup.bash" >> ~/.bashrc

source ~/.bashrc

1. 安装依赖

sudo apt-get install python-rosinstall python-rosinstall-generator python-wstool build-essential

打开第一个终端，输入

roscore

打开第二个终端，输入

rosrun turtlesim turtlesim\_node

打开第三个终端，输入

rosrun turtlesim turtle\_teleop\_key

打开第四个终端，输入

rosrun rqt\_graph rqt\_graph

看到ros节点信息，ros安装完毕

3.**相机SDK安装（注意：安装时不要连接相机）**

1）更新环境

sudo apt-get update && sudo apt-get upgrade && sudo apt-get dist-upgrade

1. 下载包

git clone -b v2.50.0 https://github.com/IntelRealSense/librealsense.git

1. 安装依赖

sudo apt-get install git libssl-dev libusb-1.0-0-dev pkg-config libgtk-3-dev

sudo apt-get install libglfw3-dev

1. 安装realsense所需内核模块

cd librealsense

./scripts/setup\_udev\_rules.sh

./scripts/patch-realsense-ubuntu-lts.sh

1. 编译安装

mkdir build && cd build

cmake ../ -DCMAKE\_BUILD\_TYPE=Release -DBUILD\_EXAMPLES=true

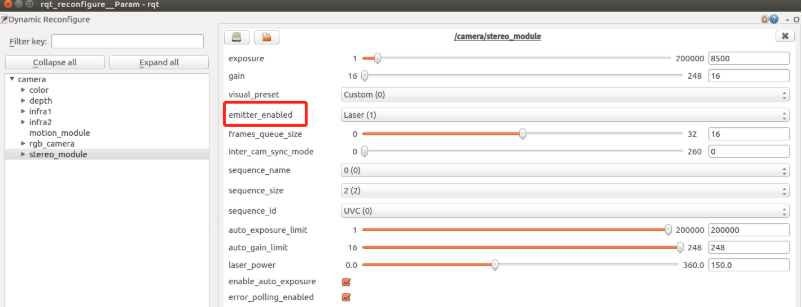
sudo make uninstall && make clean && make -j8 && sudo make install

在终端打开viewer，连接摄像头就能看到内容

realsense-viewer

1. 关闭红外结构光

打开一个新的终端，输入rosrun rqt\_reconfigure rqt\_reconfigure启动Reconfigure节点



将其设置为off，即可关闭红外结构光。

**二、内参标定**

相机内参可以直接通过realsense sdk获取

rs-enumerate-devices -c

如果需要对畸变进行修正可以重新进行内参标定，这里使用棋盘格标定。

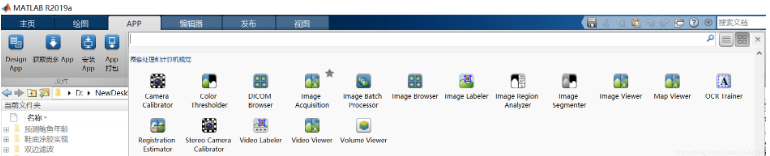
**1.标定棋盘格生成**

网址：https://calib.io/pages/camera-calibration-pattern-generator

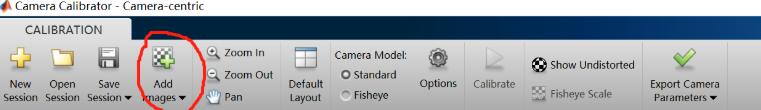
通过调整标定板或摄像机的方向，为标定物拍摄一组不同方向的照片，最好大于10张图片

1. **matlab工具箱标定**

打开Matlab，在Matlab中找到APP，点击进入，找到Camera Calibrator工具箱



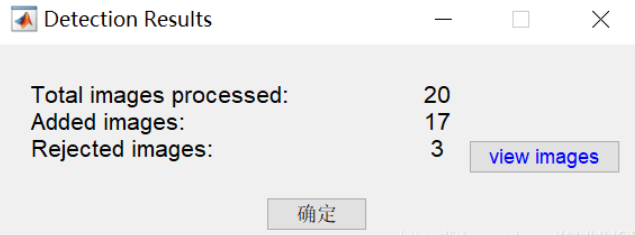
进入工具箱，点击Add Images



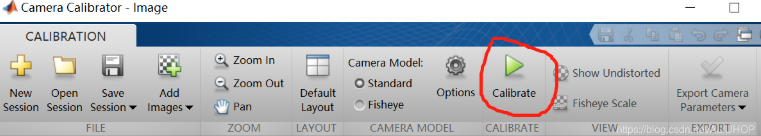
将所有图片添加进去，比如这里25是指棋盘中正方形的边长大小

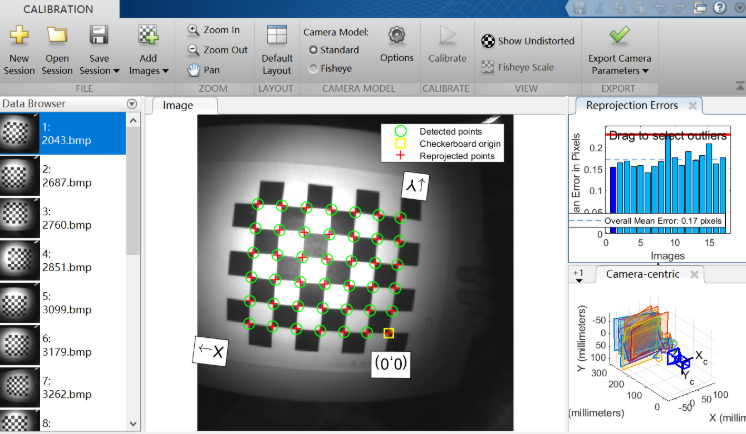


这里表示3张图像不可使用



点击Calibrate按钮，即完成标定，其中拍摄图像如图所示

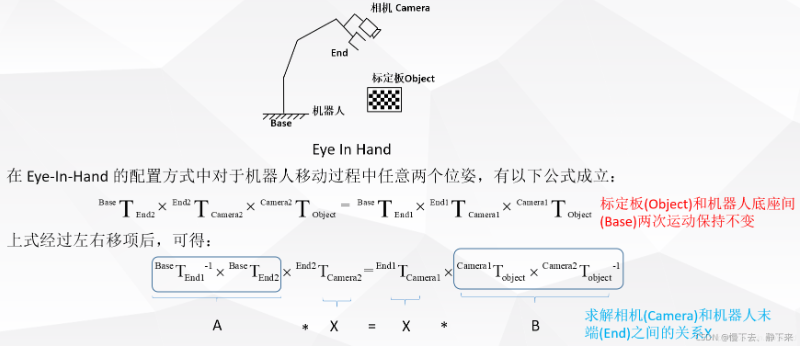




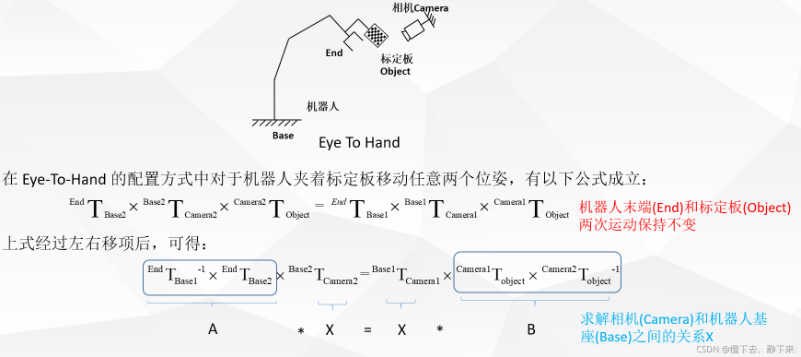
**三、外参标定**

主要包括两种：眼在手上（相机固定在机械臂末端）和眼在手外（相机固定在地面支架）

眼在手上：是相机与机械臂的末端坐标系固连，标定板与机器人的基坐标系相对位置关系不变，我们标定的是相机坐标系在末端坐标系下位姿。

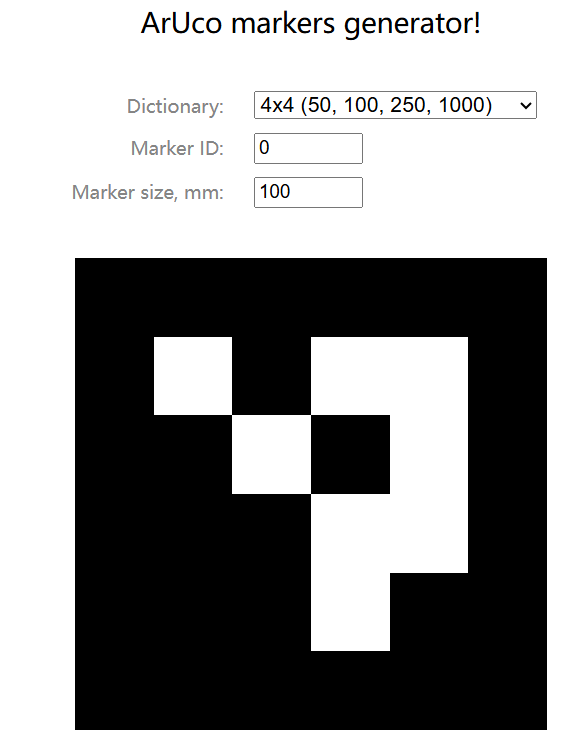


眼在手外：也就是相机固定不动，标定板与机器人的末端坐标系固连，我们需要标定的是相机坐标系在基坐标系下的位姿。



1. **Aruco码标定模板**

1）生成Aruco Marker网站：<https://chev.me/arucogen/>



注意选择original格式，打印出来之后作为靶标。

1. 安装aruco\_ros包

cd ~/catkin\_ws/src

git clone  https://github.com/pal-robotics/aruco\_ros.git

cd ..

catkin\_make

3）安装realsense\_ros包

<https://github.com/IntelRealSense/realsense-ros>

4）配准Aruco launch文件

打开aruco\_ros包中single.launch

roscd aruco\_ros

cd launch

gedit single.launch

修改single.launch文件，需要修改markerId markerSize为前面生成的aruco的id和size，以及/camera\_info /image指向realsense发布的话题，camera\_frame改为/camera\_link.

下面以编号id为123，边长size为0.1m的aruco码为例修改launch文件

<launch>

    <arg name="markerId"        default="123"/>

    <arg name="markerSize"      default="0.1"/>    *<!-- in m -->*

    <arg name="eye"             default="left"/>

    <arg name="marker\_frame"    default="aruco\_marker\_frame"/>

    <arg name="ref\_frame"       default=""/>  *<!-- leave empty and the pose will be published wrt param parent\_name -->*

    <arg name="corner\_refinement" default="LINES" /> *<!-- NONE, HARRIS, LINES, SUBPIX -->*

    <node pkg="aruco\_ros" type="single" name="aruco\_single">

        <remap from="/camera\_info" to="/camera/color/camera\_info" />

        <remap from="/image" to="/camera/color/image\_raw" />

        <param name="image\_is\_rectified" value="True"/>

        <param name="marker\_size"        value="$(arg markerSize)"/>

        <param name="marker\_id"          value="$(arg markerId)"/>

        <param name="reference\_frame"    value="$(arg ref\_frame)"/>

        <param name="camera\_frame"       value="/camera\_link"/>

        <param name="marker\_frame"       value="$(arg marker\_frame)" />

        <param name="corner\_refinement"  value="$(arg corner\_refinement)" />

    </node>

</launch>

1. **获取Aruco相对于相机的位姿**
2. 运行aruco\_ros+realsense节点

roslaunch realsense2\_camera rs\_camera.launch

roslaunch aruco\_ros single.launch

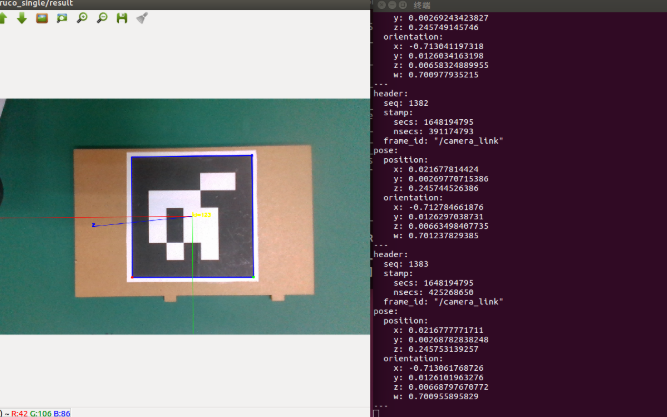
1. 查看显示的图像和返回的位姿

rosrun image\_view image\_view image:=/aruco\_single/result

rostopic echo /aruco\_single/pose

1. 检查返回图像和位姿正确性

位姿包括位置（x,y,z）和四元数（w,x,y,z），注意需要先将四元数转化为欧拉角（z,y,x）。但是四元数会有两种表示方式(w,x,y,z)和(x,y,z,w)，attitudeVectorToMatrix函数中使用(w,x,y,z)，但是aruco返回的姿态的四元数为(x,y,z,w)一定要记得修改



**3.获取机械臂末端的位姿**

直接读取示教器上机械臂末端位姿，机械臂的末端坐标系是以最后一个轴的旋转面作为原始平面建立的，不是末端法兰的平面，两者之间至少有20cm的距离.

**4.求解外参标定矩阵**

注意事项：

(1) 不管采集多少组用于标定的运动数据，每组运动使运动角度最大。

(2) 使两组运动的旋转轴角度最大。

(3) 每组运动中机械臂末端运动距离尽量小，可通路径规划实现该条件。

(4) 尽量减小相机中心到标定板的距离，可使用适当小的标定板。

(5) 尽量采集多组用于求解的数据。

(6) 使用高精度的相机标定方法。

(7) 尽量提高机械臂的绝对定位精度，如果该条件达不到，至少需要提高相对运动精度。