## 基本原理

基于普通摄像头的视觉抓取，使用了视觉处理方法，通过识别定位框的位置和物块的相对位置，获得物块到机械臂的位置。

### 颜色识别原理

颜色的参数分别是：色调（H），饱和度（S），明度（V）。

#### 色调H

用角度度量，取值范围为0°～360°，从红色开始按逆时针方向计算，

#### 饱和度S

饱和度S表示颜色接近光谱色的程度。一种颜色，可以看成是某种光谱色与白色混合的结果。

其中光谱色所占的比例愈大，颜色接近光谱色的程度就愈高，颜色的饱和度也就愈高。

饱和度高，颜色则深而艳。

光谱色的白光成分为0，饱和度达到最高。

通常取值范围为0%～100%，值越大，颜色越饱和。

#### 明度V

明度表示颜色明亮的程度，

对于光源色，明度值与发光体的光亮度有关；

对于物体色，此值和物体的透射比或反射比有关。

通常取值范围为0%（黑）到100%（白）。

RGB和CMY颜色模型都是面向硬件的，而HSV（Hue Saturation Value）颜色模型是面向用户的。

### 文件说明

#### script

主程序main调用 arm.py和camera.py，相机模块调用颜色识别和定位框识别模块

main.py---->arm.py

----->camera.py---->color\_detection.py---> cvwin.py

----->plate\_detection.py---> cvwin.py

cvwin.py 单独的cv2相关的窗口显示线程

#### launch

aiarm-client-controller.launch 虚拟机上使用的

aiarm-controller.launch 网关上使用的，一般使用这个

### 参数说明

|  |
| --- |
| #定位板相关参数  loc\_plate=[141,192,465,386] #初始值，机械臂自己根据plate\_detection.py文件中自己查找并更新此参数 单位像素 不用设置  loc\_plate\_act=[0.153,0.173,0.092] #定位板实际长度 单位 m 定位板框实际长度 上底 下底 高[0.155,0.173,0.093]  loc\_plate\_act\_origin=[0.23,-0.0148] #定位板原点偏移 单位 m 定位板相对于base\_link的x,y方向偏移量  loc\_x\_off\_mx=25  loc\_x\_off\_mi=9  loc\_y\_off\_mx=0  loc\_opt\_par\_x=1.0 #X方向的优化参数  loc\_opt\_par\_y=1.1 #Y方向的优化参数  '''  loc\_plate 单位 pix 640\*480像素 定位板框 左上角x,y 和 右下角x,y  loc\_plate\_act 单位 m 定位板框实际长度 上底 下底 高[0.147,0.182,0.082]  loc\_x\_off\_mx 单位 pix 相机倾斜视角X像素偏移 最大值  loc\_y\_off\_mx 单位 pix 相机倾斜视角y像素偏移 最大值  loc\_plate\_origin 单位 pix cv2下像素原点  loc\_plate\_act\_origin 单位 m 实际定位原点 相对base\_link  '''  #机械臂  arm\_g\_height=0.1  #拍照位  # arm\_cam\_pose=[0.1572271238, 0.01332804092, 0.2300313884, 0.04640923861, 0.4895623458, 0.00796998209, 0.87069591217]  arm\_cam\_joint=[0.11535425216048287, -0.616704188215822, 1.40664924293023, 1.8002512210459807, 0.17097973936214012]  #过渡位  arm\_trans\_joint=[5.46315397666901e-05, 0.15632741105576997, 1.284452648106851, 0.9141770383249787, -9.986523001948356e-05] |

### 坐标系说明



## 操作步骤

### 机械臂走到相机拍照位

##### 1、启动launch启动文件

启动一个ssh终端

|  |
| --- |
| **$ roslaunch marm\_visual\_control aiarm-controller.launch**  ... logging to /home/zonesion/.ros/log/24f02082-b08b-11ec-885a-b1e0c15c78ec/roslaunch-rk3399-9153.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://rk3399:46050/  SUMMARY  ========  PARAMETERS  \* /grasp\_filter\_test/arm: manipulator  [ INFO] [1648688509.308670709]: MoveGroup context using planning plugin ompl\_interface/OMPLPlanner  [ INFO] [1648688509.309991661]: MoveGroup context initialization complete  You can start planning now! |

##### 2、运行到拍照位

启动一个ssh终端

|  |
| --- |
| zonesion@rk3399:**~$ cd catkin\_ws/src/marm\_visual\_control/script/**  zonesion@rk3399:~/catkin\_ws/src/marm\_visual\_control/script**$ python main.py**  Failed to import pyassimp, see https://github.com/ros-planning/moveit/issues/86 for more info  [ INFO] [1648688601.580013535]: Loading robot model 'marm'...  [ INFO] [1648688601.580419243]: No root/virtual joint specified in SRDF. Assuming fixed joint  [ INFO] [1648688604.750564185]: Ready to take commands for planning group manipulator.  ('red color\_par', (156, 51, 190, 190, 255, 255))  ('green color\_par', (50, 102, 158, 71, 200, 200))  libpng warning: Incorrect sBIT chunk length  set cam number 0  [INFO] [1648688606.222780]: Open the fixture  ^CEXIT APP  zonesion@rk3399:~/catkin\_ws/src/marm\_visual\_control/script$ |

运行main.py脚本，当机械臂动作后，退出程序

### 颜色识别标定

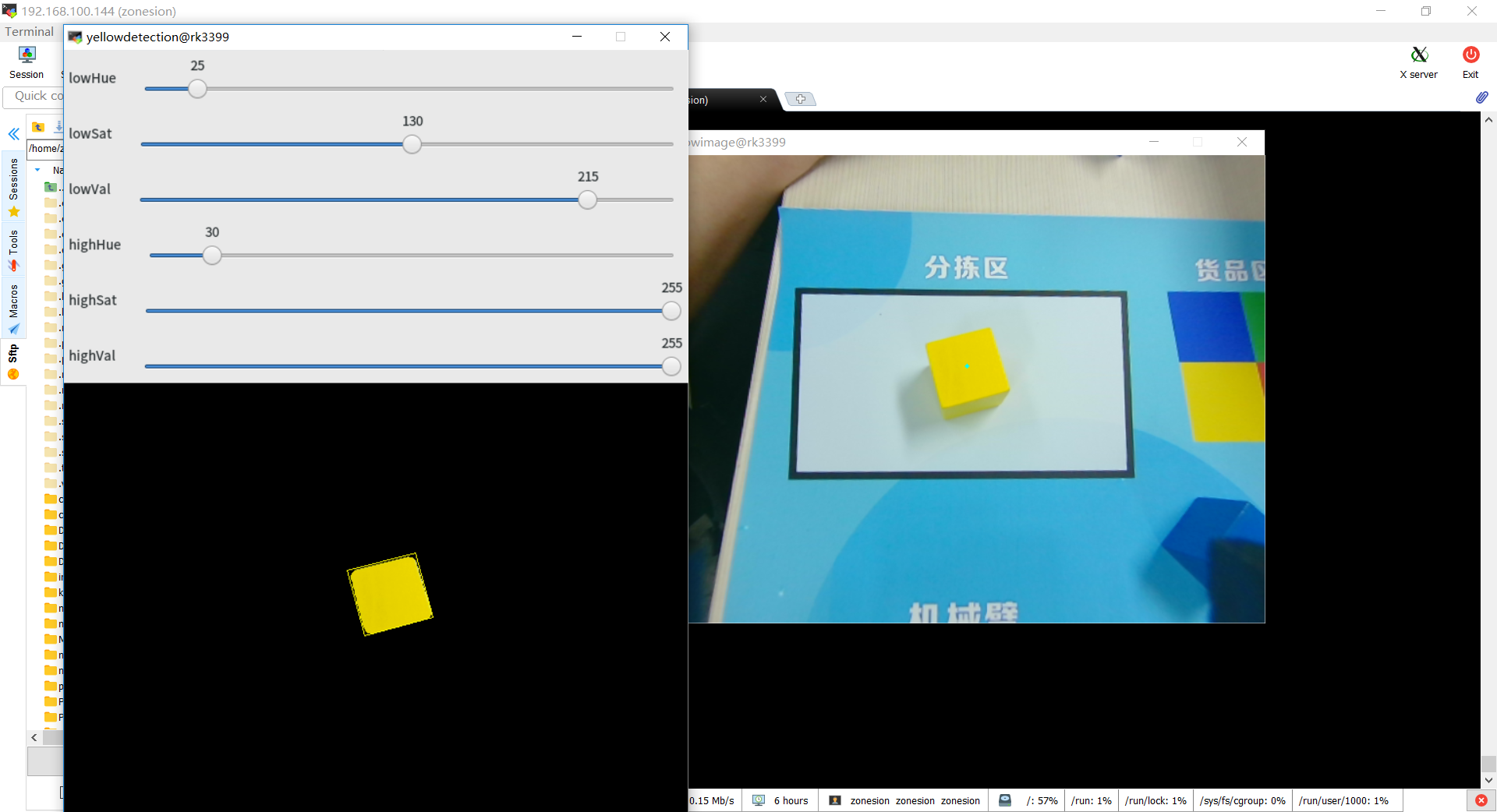
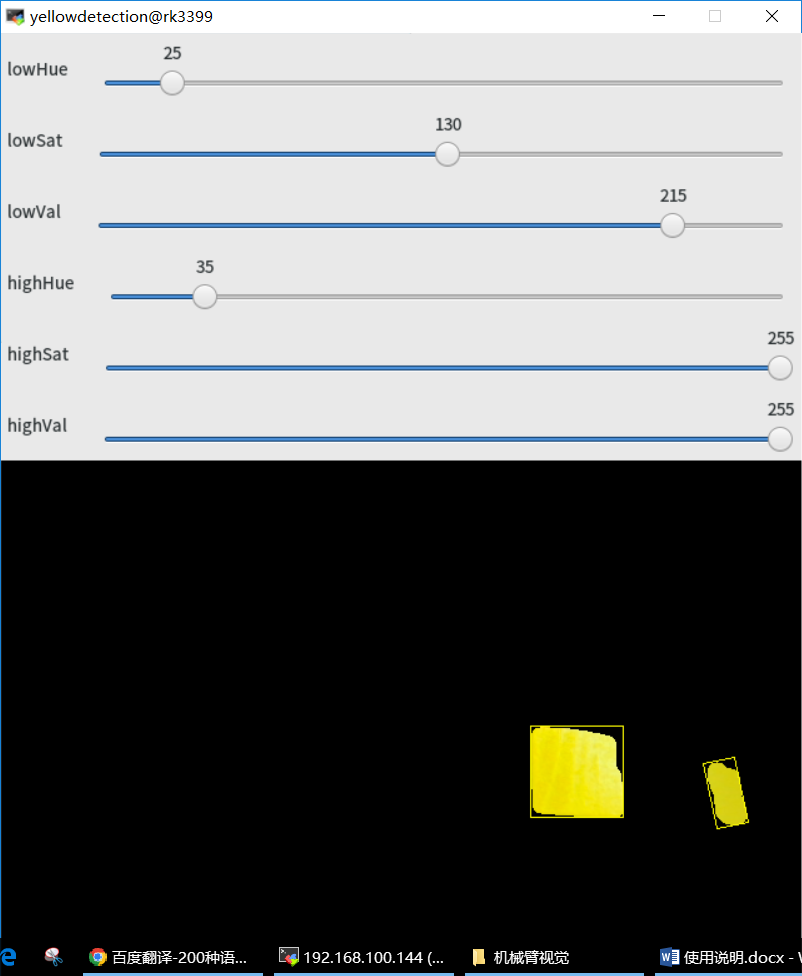
使用color\_detection.py文件

##### 1、机械臂走到拍照位

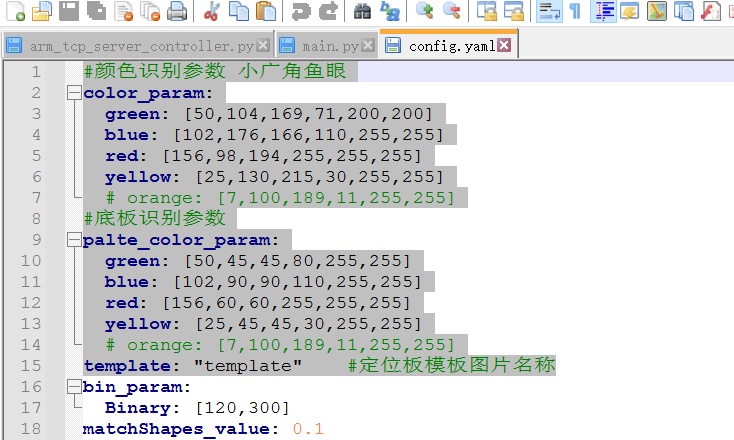
##### 2、标定颜色

zonesion@rk3399:~/catkin\_ws/src/marm\_visual\_control/script$ **python color\_detection.py -c yellow**

如标定黄色，后面跟随参数为yellow即可。如下这样标定所有颜色，

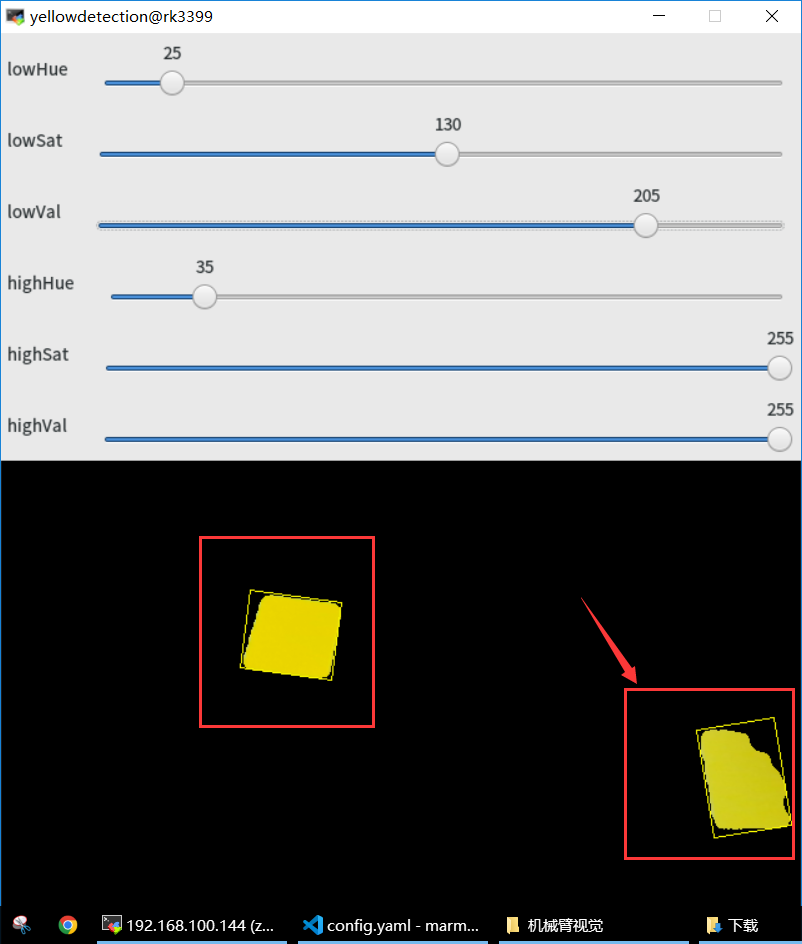
右边这样也不影响，主要黄色的木块能够识别出来即可。后面的camera应用中会将不在定位框内的进行清楚



将颜色识别参数写入到color\_param.

第一组颜色识别参数用于定位物块的位置

第二组颜色识别参数用于定位定位板框位置，为了提高定位框识别精度用的。



同时保证两个识别框数据稳定，不然会早层定位数据失败

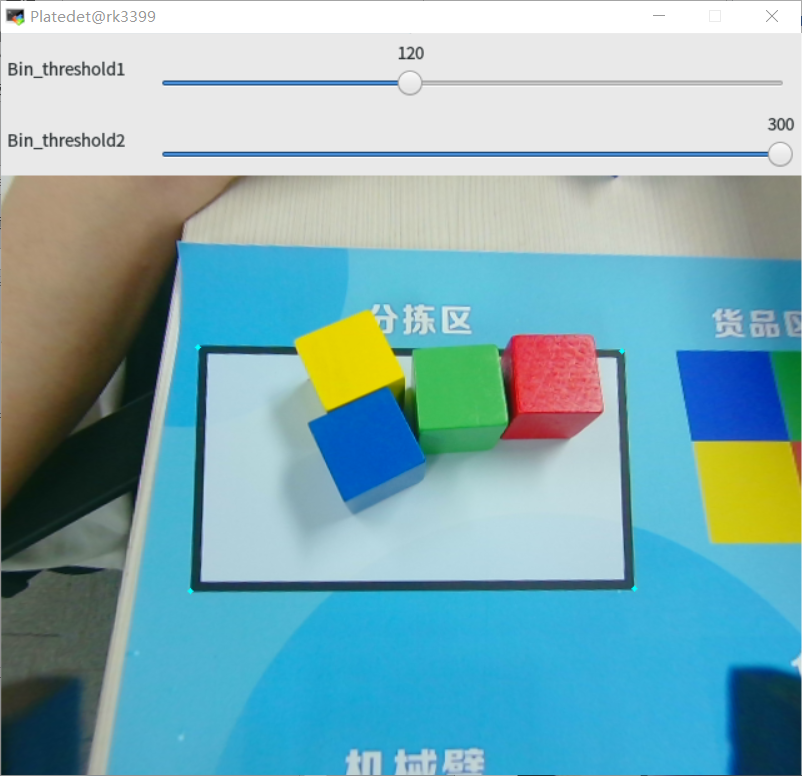
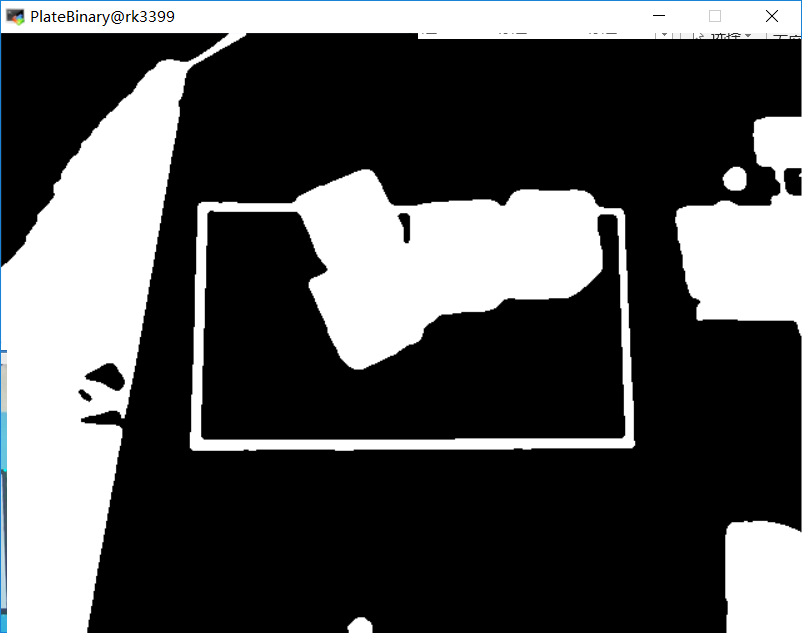
### 定位框识别标定

同样机械臂走到拍照位。

#### 1、运行plate\_detection.py

|  |
| --- |
| zonesion@rk3399:~/catkin\_ws/src/marm\_visual\_control/script**$ python plate\_detection.py**  libpng warning: Incorrect sBIT chunk length |

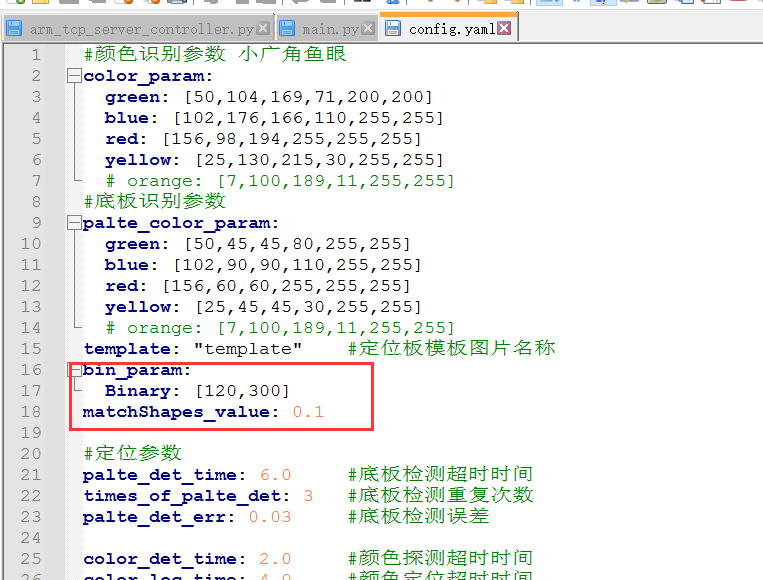
#### 2、调整参数

识别出四个角，同时物体显示为白色，说参数就可以

通常阈值2不用调，只需要调整第一个值

如上参数为：120,300

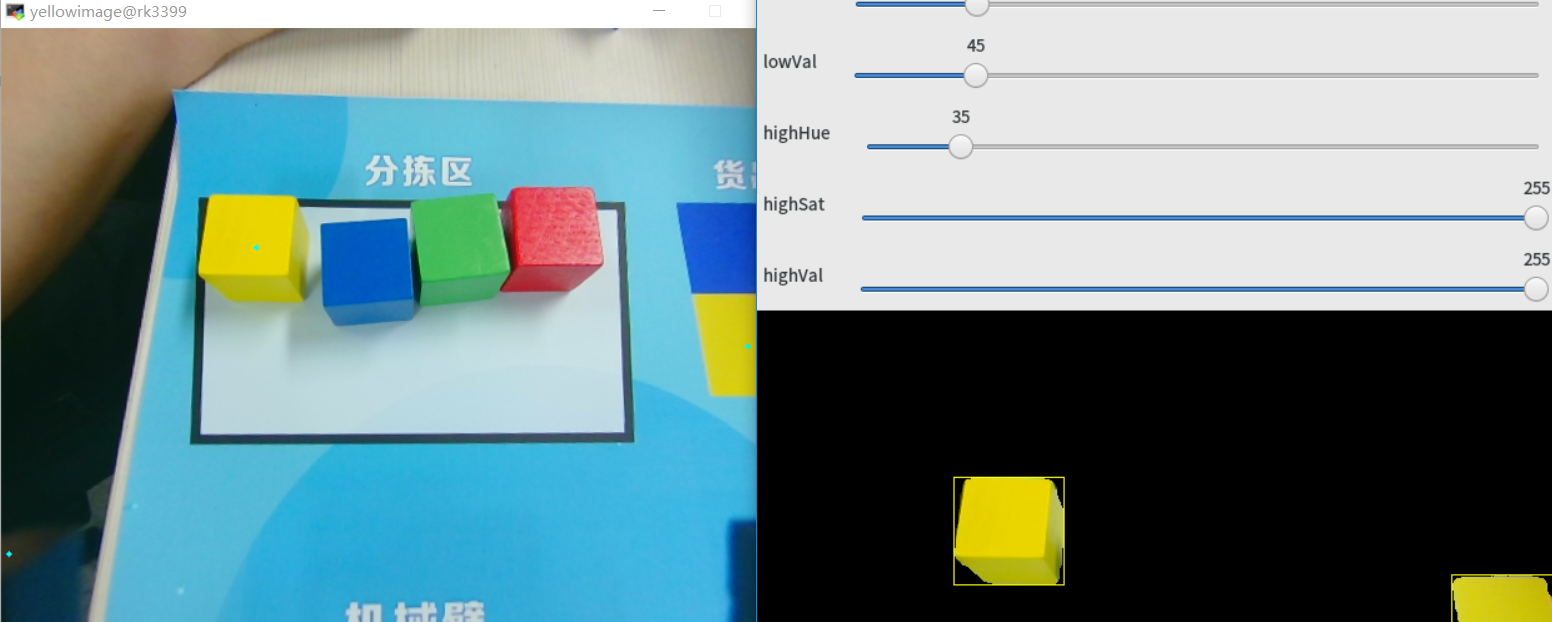


#### 3、调整色块色域参数

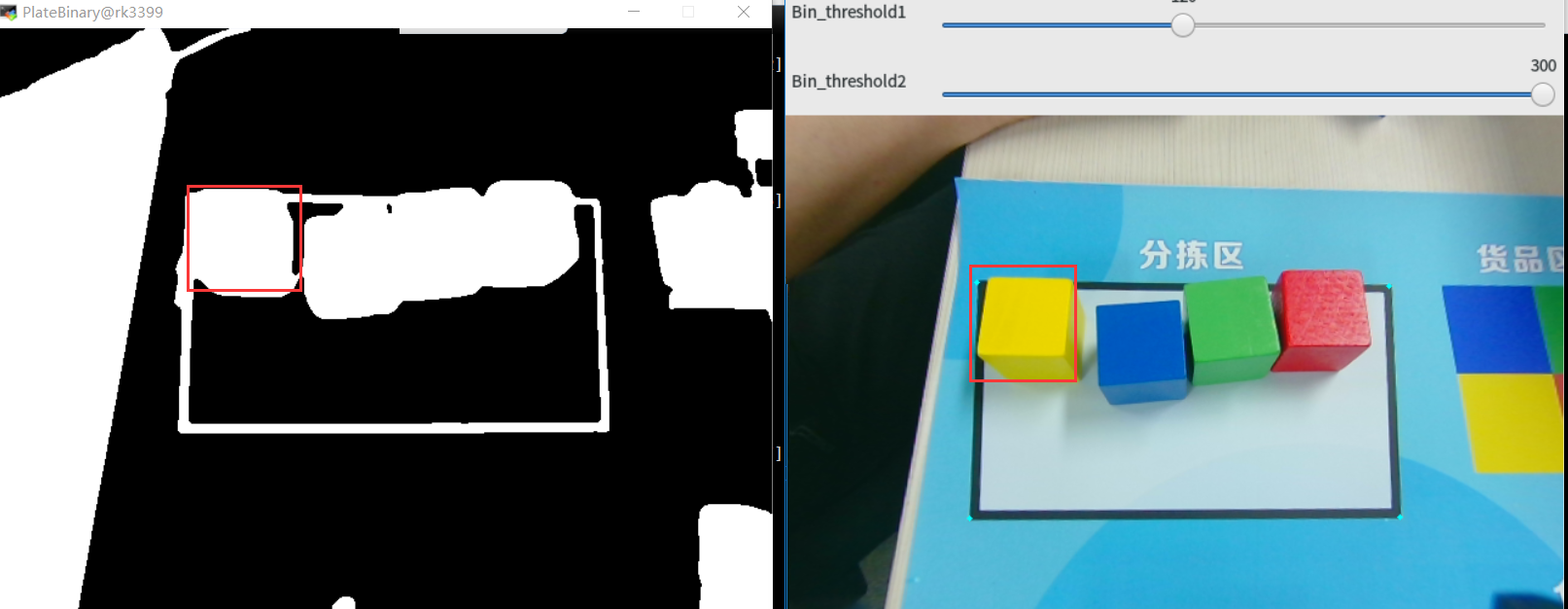
zonesion@rk3399:~/catkin\_ws/src/marm\_visual\_control/script$ **python color\_detection.py -c yellow -t 1**

('yellow color\_par', [25, 45, 45, 30, 255, 255])

('yellow', [[598, 255, 0.0], [277, 149, 24.4439544



主要黄色的木块能够整个识别即可。这样



这样就可以进行相应的取反颜色。

### 主程序说明

#### aiarmcontrol():

|  |
| --- |
| def aiarmcontrol():  # win=[98,420,103,560] #选择剪裁窗口  win = []  rospy.init\_node("AIARM\_NODE", log\_level=rospy.INFO)  aiarm=AiArm(-40, ("red","green","blue"),win,loc\_plate,loc\_plate\_act,loc\_plate\_act\_origin,loc\_x\_off\_mx,loc\_x\_off\_mi,  loc\_y\_off\_mx,this.color\_param,this.bin\_param) #初始化AIarm aiarm.all\_gohome()  aiarm.set\_joint\_value\_target(arm\_cam\_joint)  time.sleep(1)  aiarm.plate\_to\_base()  for i in aiarm.rec\_cla\_dict.keys():  point=aiarm.rec\_cla\_dict[i]["pos"]  point.insert(0,i)  aiarm.block\_pos.append(point)  while len(aiarm.block\_pos)>0 or len(aiarm.solutions)>0:  while len(aiarm.solutions)>0:  solution=aiarm.solutions[0]  # print(solution)  \_\_msg="Grabbing "+solution[0]+" wood"  rospy.loginfo(\_\_msg)  joint\_positions = solution[1]  aiarm.setGripper(False)  time.sleep(1)  aiarm.set\_joint\_value\_target(joint\_positions)  time.sleep(0.5)  if solution[3]>45:  solution[3]=90-solution[3]  aiarm.rotate\_gripper(solution[3],arm\_cam\_joint[4])  else:  aiarm.rotate\_gripper(-solution[3],arm\_cam\_joint[4])  joint\_positions = solution[2]  aiarm.set\_arm\_joint\_value\_target(joint\_positions)  time.sleep(0.5)  aiarm.setGripper(True)  time.sleep(1)  joint=aiarm.get\_joints()  arm\_trans\_joint[0]=joint[0] #不旋转1号舵机抬起  aiarm.set\_joint\_value\_target(arm\_trans\_joint)  time.sleep(0.5)  aiarm.arm\_goHome()  del aiarm.solutions[0]  time.sleep(0.5)  aiarm.all\_gohome()  time.sleep(1)  print('EXIT APP')  sys.exit() |

aiarm=AiArm(-40,("red","green"),win,loc\_plate,loc\_plate\_act,loc\_plate\_act\_origin,loc\_x\_off\_mx,loc\_x\_off\_mi,

loc\_y\_off\_mx,this.color\_param,this.bin\_param) #初始化AIarm如上，这里定义可aiarm机械臂的

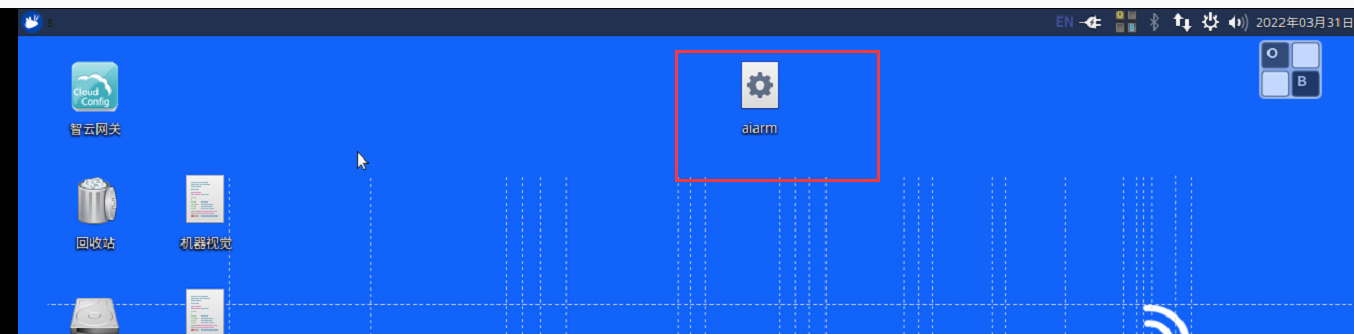
1、夹具打开角度为-40时，关闭也就是-40-80=-120

2、定义此机械臂进行"red","green"，红色和绿色两个色块的抓取。如果只抓取一个色块，定义成("red",)这样。这里的逗号不要省掉。

3、win参数，如果需要截取一部分视频窗口，可配置此参数，# win=[98,420,103,560] #选择剪裁窗口

4、其他为主要参数，见参数解释

### 启动文件



关闭所有终端，使用启动程序启动以上程序

## 注意事项

1、如果蓝色和绿色四个角识别不是很好，主要是光线不是很好，加上木块材质反光，导致四个角的颜色识别参数不一样，**演示时候可以使用红色放在四角**，红色识别比较好，**蓝色和绿色就固定放在一侧即可**。

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |