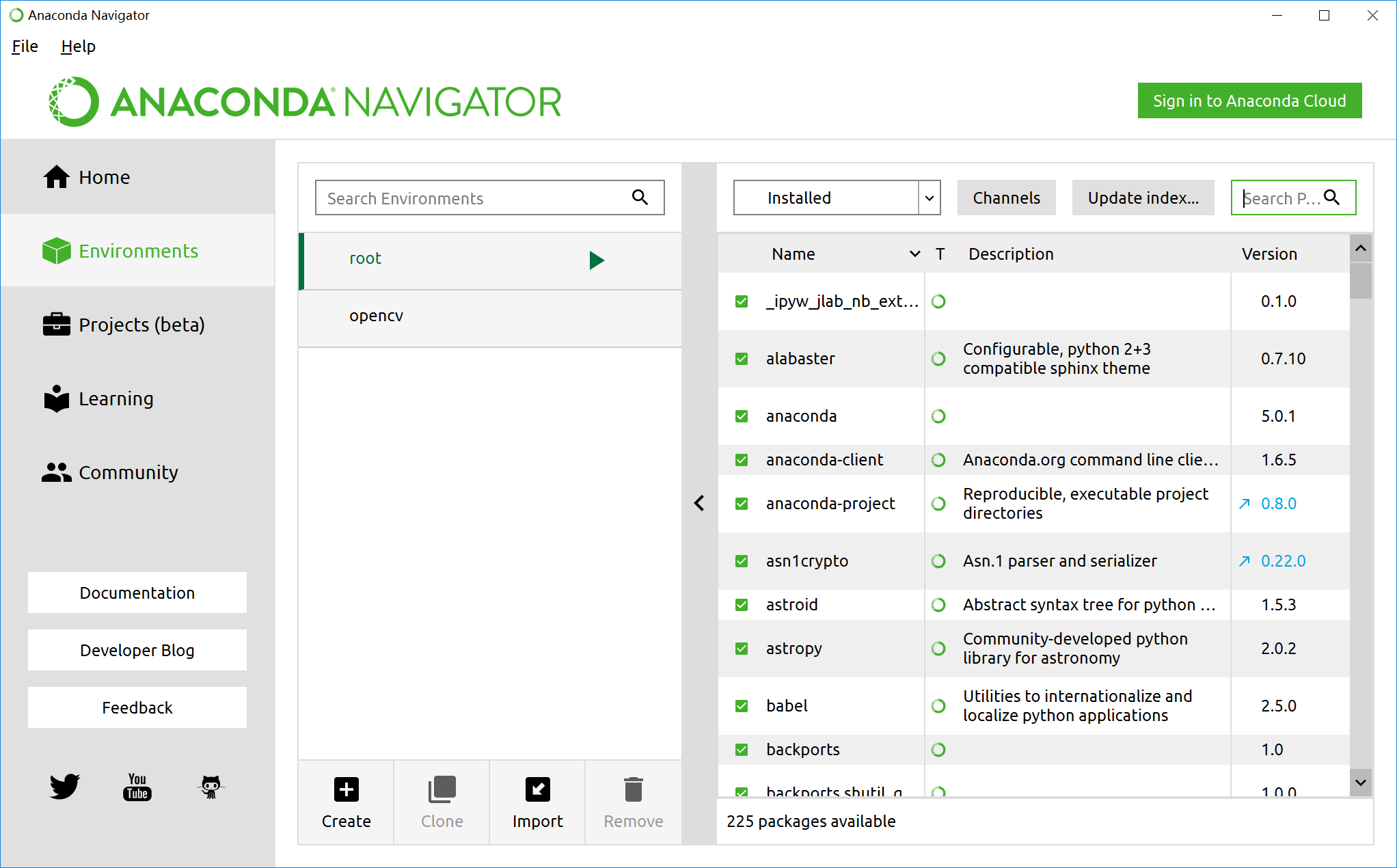
实验九

第一周实验报告

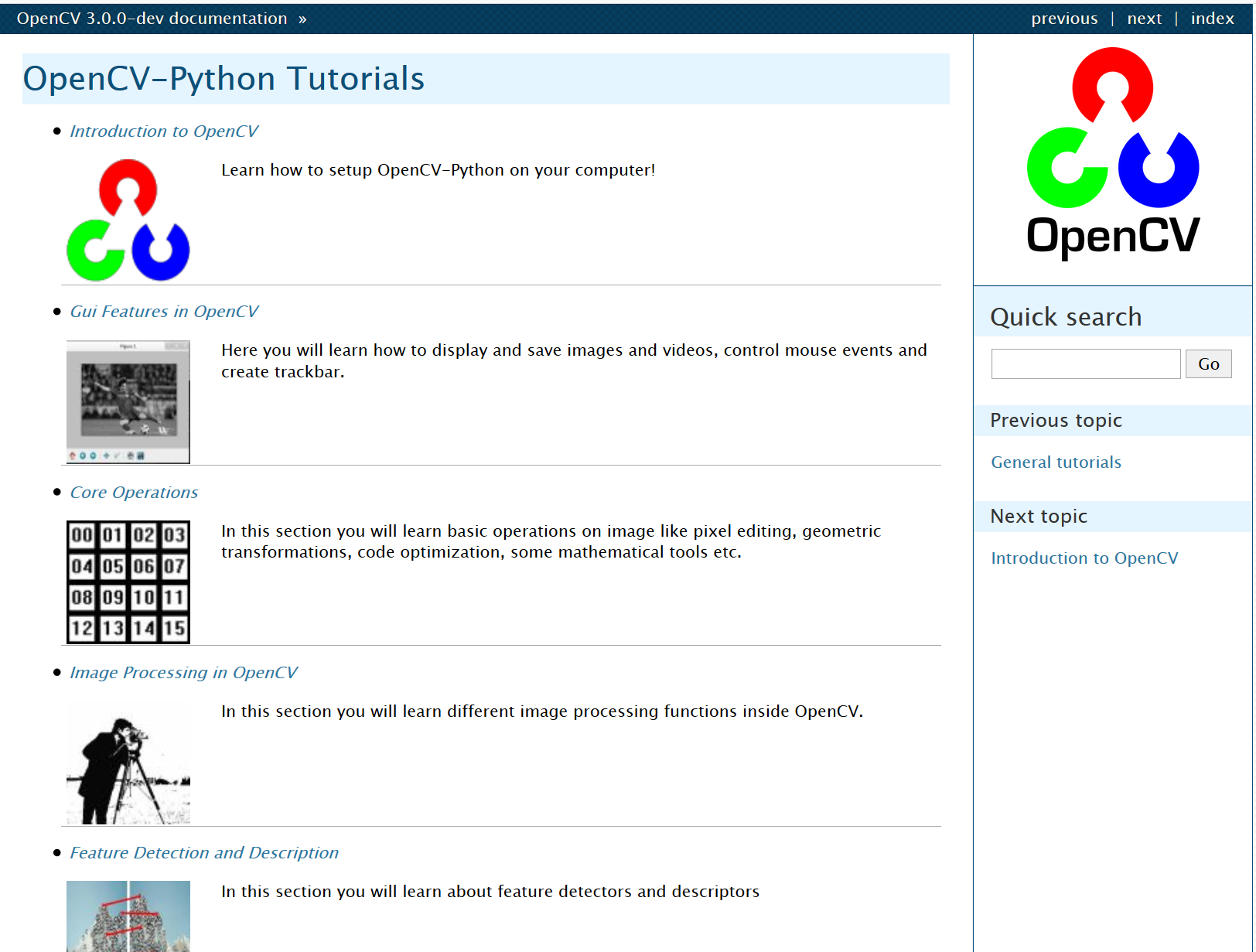
李彦瑞3140102743

1、完成了python编程软件的安装与opencv环境的搭建

安装了anaconda，建立了python开发环境，通过pip安装安装了opencv



2、上网学习了opencv-python tutorial，对opencv的一些基本操作以及基本概念进行了了解



3、对识别魔方颜色的基本方法进行了构思

打算先对图像进行一些形态学处理，之后对图像的边缘进行检测并且提取出每个小方块的轮廓，之后通过这些轮廓所包含的像素的颜色经行分析，分析出魔方表面小块的颜色。

4、采用python进行编程

以下为图像处理的几个主要语句：

img1=cv2.imread(where,1) #读取图像

img=cv2.resize(img1,(300,500)) #对图像大小进行调整

img = cv2.GaussianBlur(img,(5,5),0) #进行高斯模糊处理

#clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))

gray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY) #获得灰度图

hsv=cv2.cvtColor(img,cv2.COLOR\_BGR2HSV) #获得hsv图

binary = cv2.adaptiveThreshold(gray,255,cv2.ADAPTIVE\_THRESH\_MEAN\_C,\

cv2.THRESH\_BINARY,11,2)

#对图像进行而二值处理，采用了平均自适应的算法

#cv2.imshow("fff",binary)

face,contours, hierarchy = cv2.findContours (binary, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_NONE) #获得轮廓

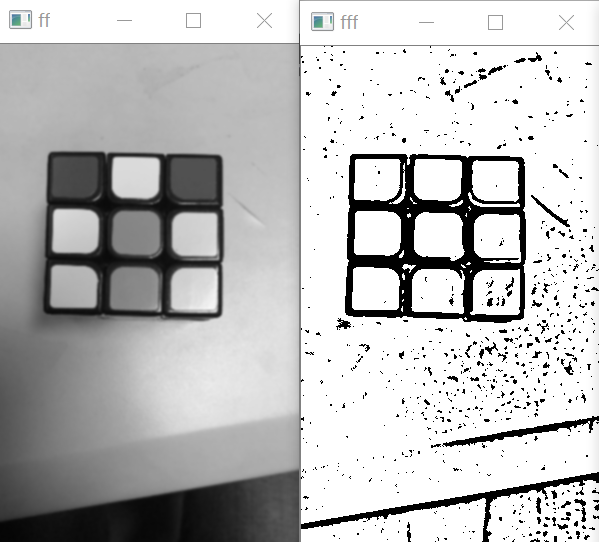
通过以上的几个语句，将轮廓从图像中提取了出来，其中说明几点：

（1）采用高斯模糊是为了增加边缘的连贯性，以及减少由于反光而导致的部分区域变白，以及排除一些噪声。

（2）转化为hsv是为了之后分析颜色的需要

（3）采用了平均自适应算法大大增高了图像轮廓查找的准确率

通过以上的处理可以获得如下图所示的效果：



左图为原图，中间图为模糊过后的灰度图，右图为二值图。

之后对而制图进行轮廓查找，并对获得的轮廓进行筛除以及检验确定9个小块颜色的位置。具体的筛选方法如下

**for i in range(len(contours)):**

**cnt=contours[i]**

**x,y,w,h = cv2.boundingRect(cnt)**

**area = cv2.contourArea(cnt)**

**rect\_area = w\*h**

**extent = float(area)/rect\_area**

**if area>10 and extent>0.8 and area<10000:**

**temp=cubespot(x,y,cnt)**

解释以上程序：

对所有的轮廓曲线进行遍历，通过**x,y,w,h = cv2.boundingRect(cnt)**函数获得外接此轮廓的长方形的位置与大小

通过**area = cv2.contourArea(cnt)**获得轮廓所包围的面积

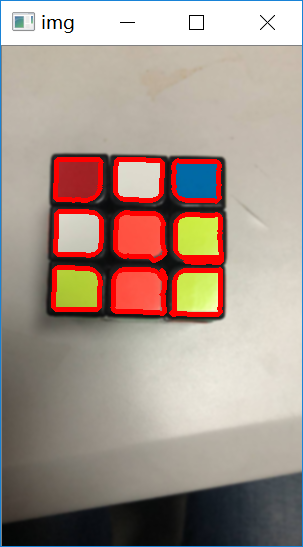
之后**extent = float(area)/rect\_area**将extent设置为轮廓面积与外接矩形的面积比值，由于先验条件我们已知所要是别的轮廓为接近方形，并且面积不会太小也不会太大，于是用

**if area>10 and extent>0.8 and area<10000:**

**temp=cubespot(x,y,cnt)**

**cube.append(temp)**

将符合条件的轮廓加入一个新的集合。



上图为找到9个小块后用红色标出的效果