数字图像处理

第三周课堂练习

李竹

杭州电子科技大学

电子信息学院

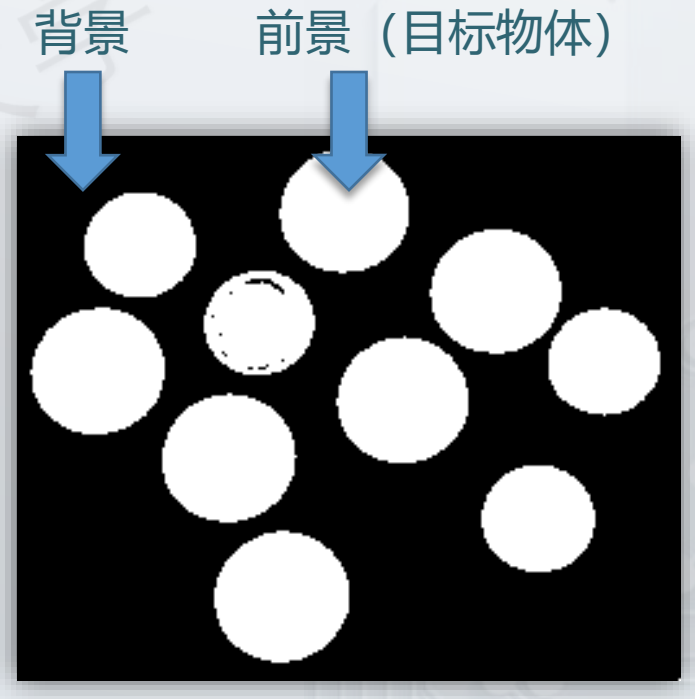
讨论

1. 关于RGB通道分离的问题

讨论

2.完成二值化后是否可以实现自动计件

原图



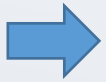
二值图

自动计件

讨论

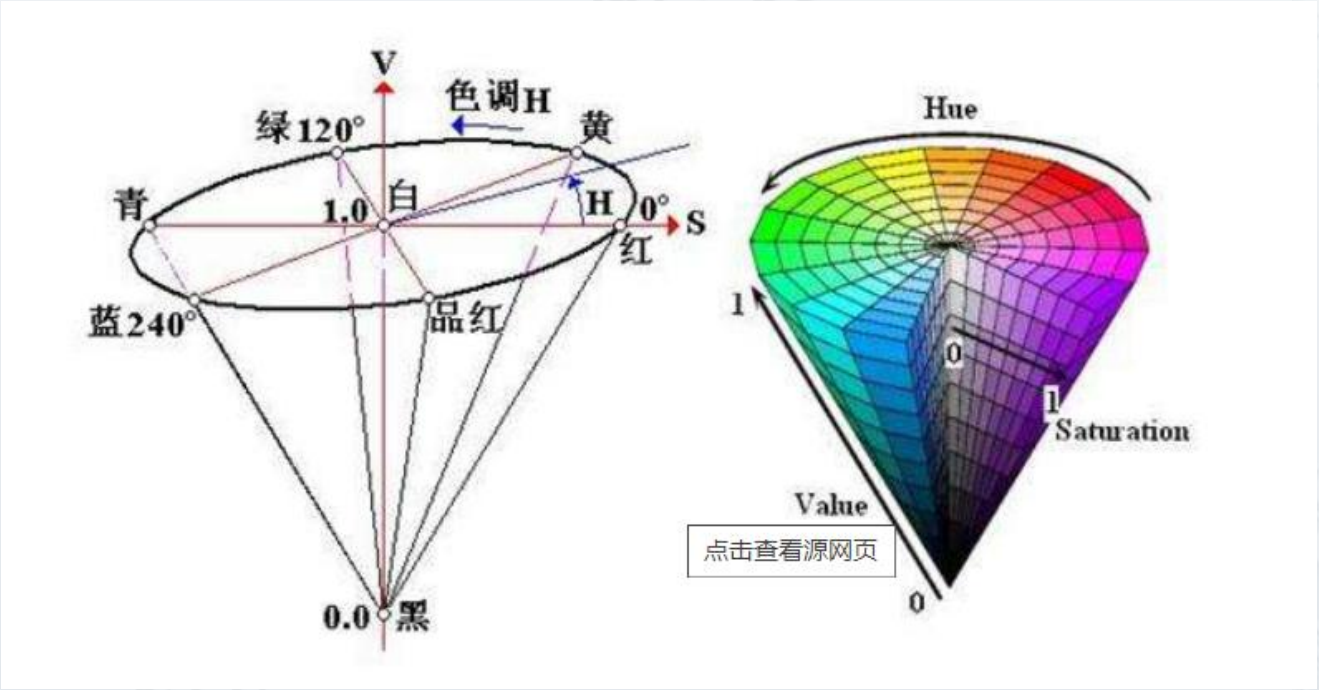
连通域标记

			1					
	1	1	1	1				
	1	1	1	1				
			1			1		
					1	1		
						1		

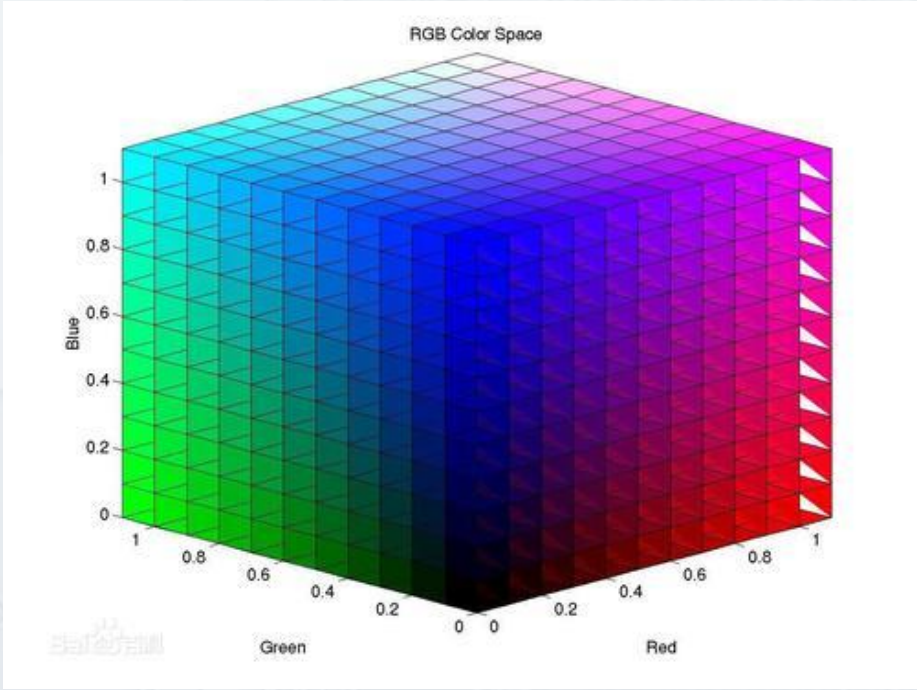


			1					
	1	1	1	1				
	1	1	1	1				
			1			2		
					2	2		
						2		

练习1


V
色调H
绿120°
黄
青
1.0
白
H
0°
S
红
蓝240°
品红
0.0
黑
Hue
1
0
1
Saturation
Value
0
点击查看源网页

HSV


RGB Color Space
Blue
Green
Red
0
0.2
0.4
0.6
0.8
1

RGB

练习1

利用不同的物体在HSV色彩空间上的不同色域，实现目标像素的提取。

主要利用iRange函数。

需要注意的是opencv中h的定义与理论值不同。

	红	黄	绿	青	蓝	品红
理论值	0	60	120	180	240	300
opencv	120	90	60	30	0	150

练习1

```cpp
#include <iostream>
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;

int main()
{
    VideoCapture cap(0);

    double scale=0.5;

    //0-180
    //肤色
    double i_minH = 0;
    double i_maxH = 20;
    //0-255
    double i_minS = 43;
    double i_maxS = 255;
    //0-255
    double i_minV = 55;
    double i_maxV = 255;

    while (1)
    {
        Mat frame;
        Mat hsvMat;
        Mat detectMat;

        cap >> frame;
        Size ResImgSiz = Size(frame.cols*scale, frame.rows*scale);
        Mat rFrame= Mat(ResImgSiz, frame.type());
        resize(frame, rFrame, ResImgSiz, INTER_LINEAR);

        cvtColor(rFrame, hsvMat, COLOR_BGR2HSV);

        rFrame.copyTo(detectMat);

        cv::inRange(hsvMat, Scalar(i_minH, i_minS, i_minV), Scalar(i_maxH, i_maxS, i_maxV), detectMat);

        imshow("whie: in the range", detectMat);
        imshow("frame", rFrame);

        waitKey(30);
    }
}
```

练习1

头文件，和命名空间

```
#include <iostream>
#include <opencv2/opencv.hpp>
using namespace cv;
using namespace std;
```

调用摄像头，并对摄像头取得的帧进行缩放，

```
VideoCapture cap(0);

double scale=0.5;
```

利用resize

```
cap >> frame;
Size ResImgSiz = Size(frame.cols*scale, frame.rows*scale);
Mat rFrame= Mat(ResImgSiz, frame.type());
resize(frame, rFrame, ResImgSiz, INTER_LINEAR);
```

练习1

首先分别定义并赋值HSV各自的范围，可以通过调节不同的范围值获得不同的结果。

```
//0-180
//肤色
double i_minH = 0;
double i_maxH = 20;
//0-255
double i_minS = 43;
double i_maxS = 255;
//0-255
double i_minV = 55;
double i_maxV = 255;
```

利用cvtColor函数，将原图转成hsv格式，需要设置最后一位参数为COLOR_BGR2HSV

```
cvtColor(rFrame, hsvMat, COLOR_BGR2HSV);

rFrame.copyTo(detectMat);
```

练习1

首先分别定义并赋值HSV各自的范围

```
//0-180
//肤色
double i_minH = 0;
double i_maxH = 20;
//0-255
double i_minS = 43;
double i_maxS = 255;
//0-255
double i_minV = 55;
double i_maxV = 255;
```

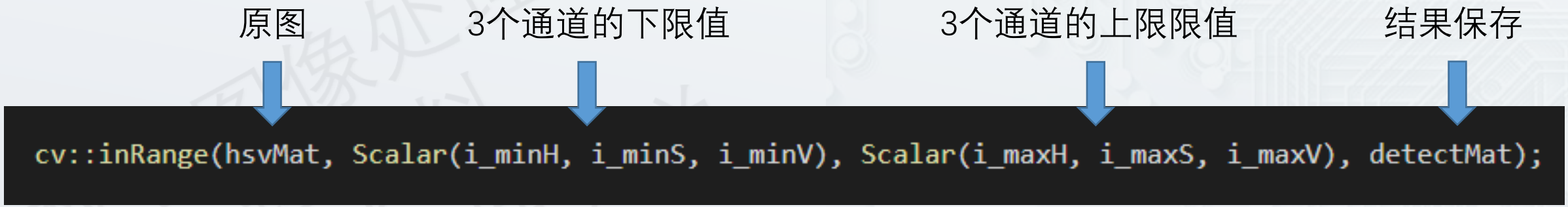
利用cvtColor函数，将原图转成hsv格式，需要设置最后一位参数为COLOR_BGR2HSV

```
cvtColor(rFrame, hsvMat, COLOR_BGR2HSV);
```

赋值一个原图，作为显示用

```
rFrame.copyTo(detectMat);
```

练习1



Scalar 数据格式：

```
typedef struct Scalar
{
    double val[4];
}Scalar;
```

Scalar是一个由长度为4的数组作为元素构成的结构体，Scalar最多可以存储四个值，没有提供的值默认是0。

练习2

调用2种二值化的函数，实现二值化，本练习不提供完整示例。

提示：opencv的二值化函数是可以对彩色图像进行处理的，但是我们一般不会对彩色图像进行二值化，所以在调用二值化函数之前，先将原图转为灰度图。

练习2

区域自适应二值化。

1.原图

2.结果图

3.预设满足条件的最大值

5.二进制阈值或反二进制阈值

7.该参数和算法有关

```
adaptiveThreshold(srcMat,result,255,ADAPTIVE_THRESH_GAUSSIAN_C,THRESH_BINARY_INV,15,10);
```

4.自适应阈值算法

6.局部区域的尺寸，一般选择为3、5、7.....等。

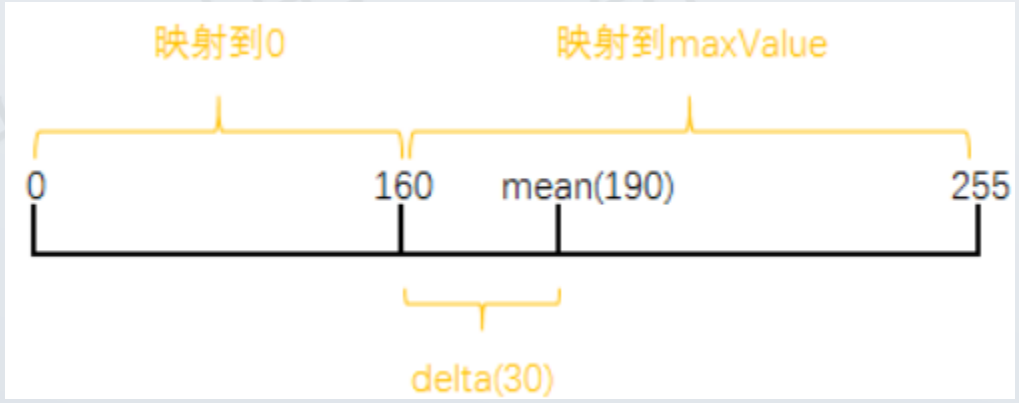
练习2

关于参数4和参数5

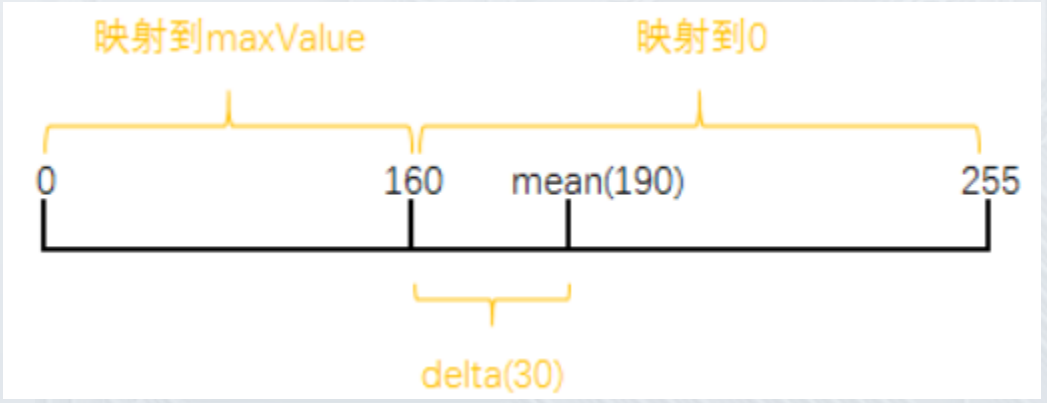
参数4中：

ADAPTIVE_THRESH_MEAN_C，为局部邻域块的平均值。该算法是先求出块中的均值，再减去常数C。

ADAPTIVE_THRESH_GAUSSIAN_C，为局部邻域块的高斯加权和。该算法是在区域中（x，y）周围的像素根据高斯函数按照他们离中心点的距离进行加权计算，再减去常数C。



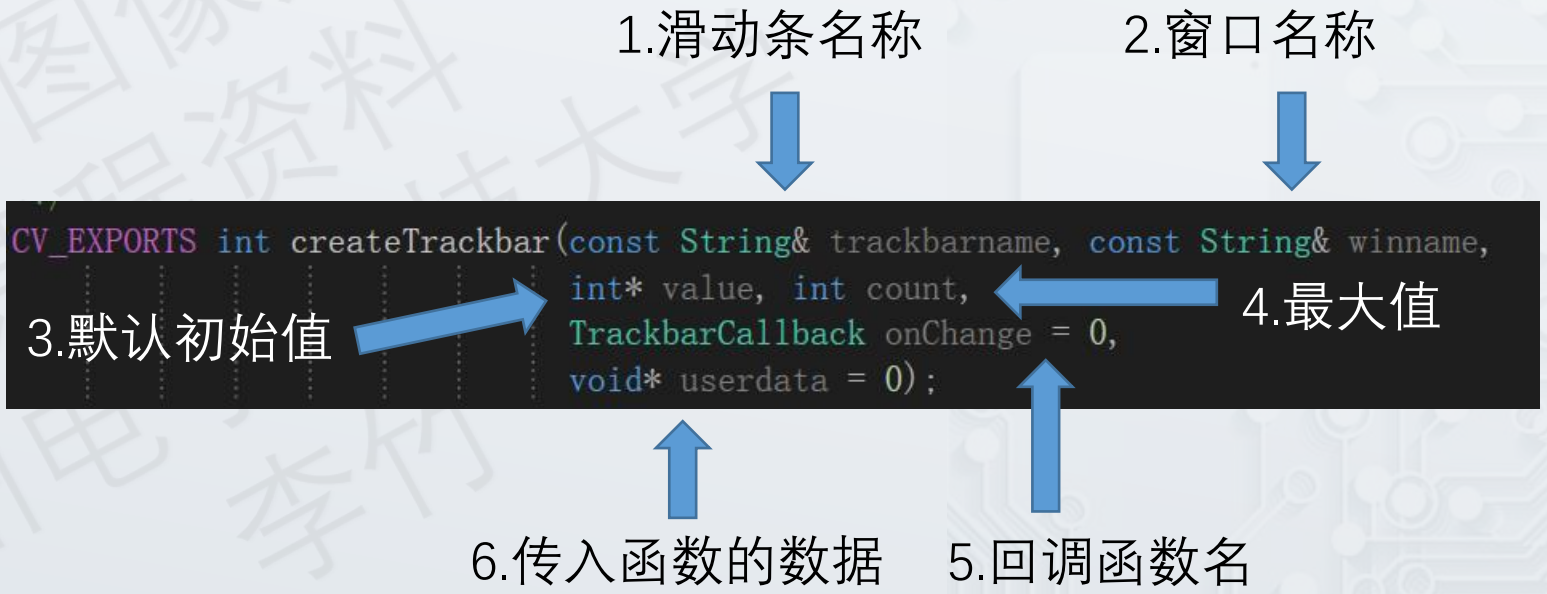
THRESH_BINARY



THRESH_BINARY_INV

练习3

回调函数



练习3

```
string window_name = "binaryMat";

void threshod_Mat(int th, void* data)
{

    Mat src = *(Mat*)(data);

    Mat dst;
    // 二值化
    threshold(src, dst, th, 255, 0);

    imshow(window_name, dst);

}
```

```
int main()
{

    Mat srcMat;
    Mat gryMat;
    int lowTh = 30;
    int maxTh = 255;

    srcMat = imread("D:\\test2.jpg");
    if (!srcMat.data) //判断图像是否载入
    {
        cout << "图像载入失败！" << endl;
        return 0;
    }
    //  imshow(window_name, Image);
    cvtColor(srcMat, gryMat, CV_BGR2GRAY);
    imshow(window_name, gryMat);
    createTrackbar("threshold",
                   window_name,
                   &lowTh,
                   maxTh,
                   threshod_Mat,
                   &gryMat);
    waitKey(0);

    return 0;
}
```