在 OpenCV2 中，图像的读取和显示通过 imread()和 imshow()函数实现。 、

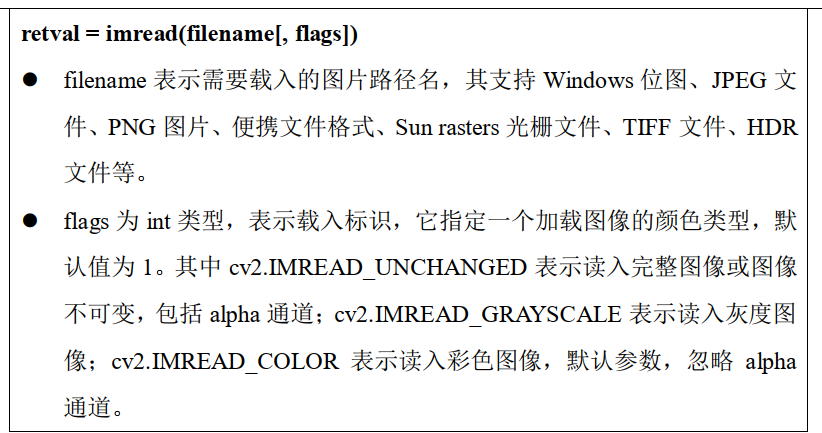
读取图像的 imread()函数将从指定的文件加载图像并返回矩阵，如果无法读取图像（因为缺少文件、权限不正确、格式不支持或图像无效等），则返回空矩阵。

cv2.imread()接口读图像，读进来直接是BGR 格式数据格式在 0~255

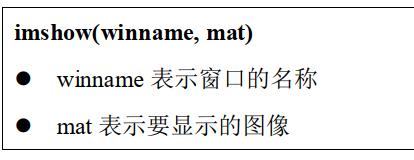
需要特别注意的是图片读出来的格式是BGR，不是常见的RGB格式，颜色有区别。

opencv的接口使用BGR，而matplotlib.pyplot 则是RGB模式，使用matplotlib.pyplot输出时需要

img = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)转成RGB模式

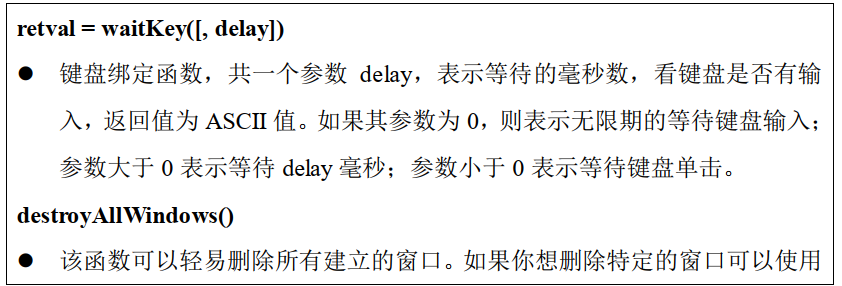


显示图像调用 imshow()函数，它将在指定窗口中显示一幅图像，窗口会自动调整为图像大小



图像显示过程中，如果代码中没有 waitKey(0)函数，其运行结果可能会出现错误

在显示图像过程中，通常还会调用两个操作窗口的函数，它们分别是waitKey()和 destroyAllWindows()。





OpenCV 中读取图像的像素值可以直接通过遍历图像的位置实现，如果是灰度图像则返回其灰度值，如果是彩色图像则返回蓝色（ B）、绿色（G）、红色（G）三个分量值。

灰度图像：返回值 = 图像[位置参数]

彩色图像：返回值 = 图像[位置元素, 0 | 1 | 2 ]获取 BGR 三个通道像素

在图像处理中， NumPy 读取像素调用 item()函数实现，修改像素调用 itemset()实现

返回值 = 图像.item(位置参数)

由于在 OpenCV2 中没有 CreateImage 函数，如果需要创建图像，则需要使用 Numpy 库函数实现。如下述代码，调用 np.zeros()函数创建空图像，创建的新图像使用 Numpy 数组的属性来表示图像的尺寸和通道信息，其中参数 img.shape

表示原始图像的形状， np.uint8 表示类型。

emptyImage = np.zeros(img.shape, np.uint8)

例如 img.shape 为(500, 300, 3)，它表示 500×300 像素的图像， 3 表示这是一个 RGB 图像。

复制原有图像来获取一幅新图像，可以调用 copy()函数实现

emptyImage = img.copy()

在 OpenCV 中，输出图像到文件使用的函数为 imwrite()

