2、Open Source CV几何变换

2.1、OpenCV图片缩放

2.1.1 、 在OpenCV中，实现图像缩放的函数是：cv2.resize(InputArray src, OutputArray dst, Size, fx, fy, interpolation)

参数含义：

InputArray src：输入图片

OutputArray ds：输出图片

Size：输出图片尺寸

fx,fy：沿x轴，y轴的缩放系数

interpolation：插入方式，可选择INTER_NEAREST（最近邻插值），INTER_LINEAR（双线性插值（默认设置）），INTER_AREA（使用像素区域关系进行重采样），INTER_CUBIC（4x4像素邻域的双三次插值），INTER_LANCZOS4（8x8像素邻域的Lanczos插值）

需要注意：

1.输出尺寸格式为（宽，高）

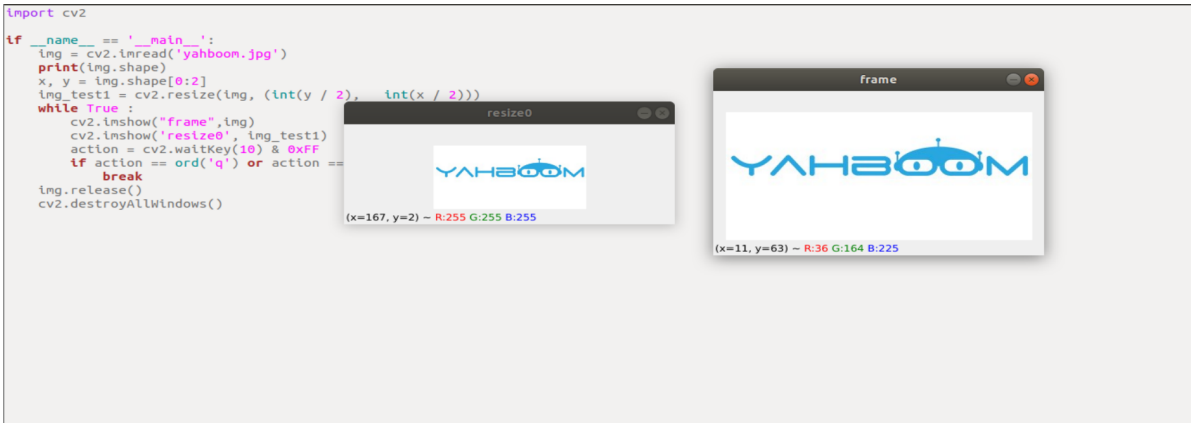
2.默认的插值方法为：双线性插值

2.1.2、代码与实际效果展示

运行程序

```
python ~/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/2_1.py
```

```
import cv2
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    print(img.shape)
    x, y = img.shape[0:2]
    img_test1 = cv2.resize(img, (int(y / 2),   int(x / 2)))
    while True :
        cv2.imshow("frame",img)
        cv2.imshow('resize0', img_test1)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```



2.2、OpenCV图片剪裁

2.2.1、图片剪切

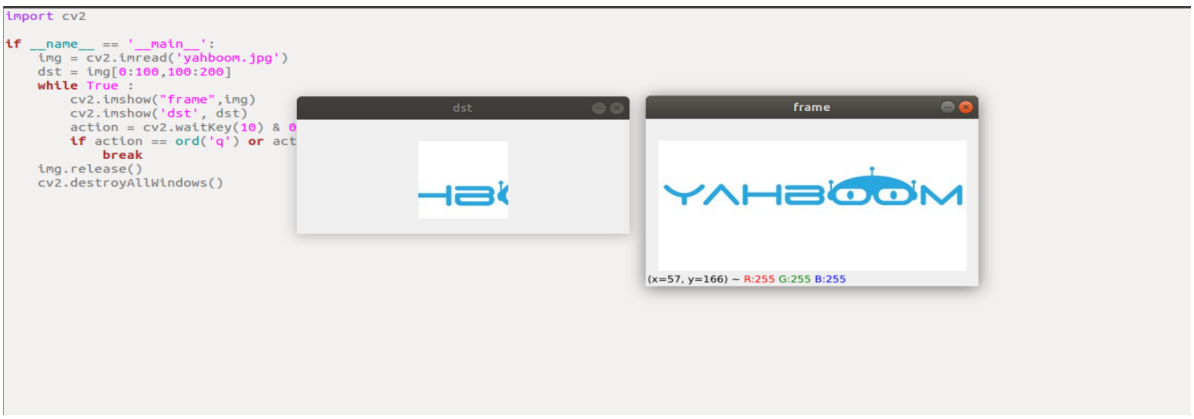
首先读取图像，然后再数组中获取像素点区域。下面代码中选取形区域X：300-500 Y：500-700，注意图像尺寸是800*800，所以选择的区域不要超过此分辨率。

2.2.2、代码与实际效果展示

运行程序

```
python ~/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/2_2.py
```

```
import cv2
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    dst = img[0:100,100:200]
    while True :
        cv2.imshow("frame",img)
        cv2.imshow('dst', dst)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```



2.3、OpenCV图片平移

2.3.1、在OpenCV中，通过仿射变换来实现图片平移，用到的方法就是cv2.warpAffine(src, M, dsize[,dst[, flags[, borderMode[, borderValue]]]])

参数含义：

src - 输入图像。
M - 变换矩阵。
dsize - 输出图像的大小。
flags - 插值方法的组合（int 类型！）
borderMode - 边界像素模式（int 类型！）
borderValue - （重点！）边界填充值; 默认情况下，它为0。

上述参数中：M作为仿射变换矩阵，一般反映平移或旋转的关系，为InputArray类型的2×3的变换矩阵。在日常进行仿射变换是，只设置前三个参数的情况下，如cv2.warpAffine(img,M,(rows,cols))就可以实现基本的仿射变换效果。

2.3.2、如何得到转换矩阵M？下列举个例子说明，

通过转换矩阵M实现将原始图像src转换为目标图像dst：

$$dst(x, y) = src(M11x + M12y+M13, M21x+M22y+M23)$$

将原始图像src向右侧移动200、向下移动100个像素，则其对应关系为：

$$dst(x, y) = src(x+200, y+100)$$

将上述表达式补充完整，即：

$$dst(x, y) = src(1 \cdot x + 0 \cdot y + 200, 0 \cdot x + 1 \cdot y + 100)$$

根据上述表达式，可以确定对应的转换矩阵M中各个元素的值为：

M11=1

M12=0

M13=200

M21=0

M22=1

M23=100

将上述值代入转换矩阵M，得到：

$$M = \begin{bmatrix} & \end{bmatrix}$$

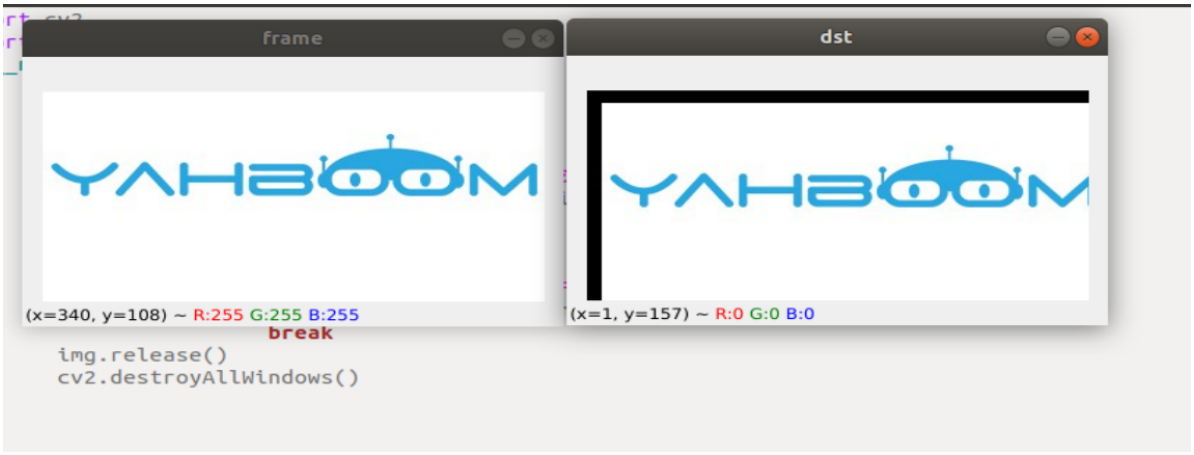
2.3.3、代码与实际效果展示

运行程序

```
python ~/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/2_3.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    imgInfo = img.shape
```

```
    height = imgInfo[0]
    width = imgInfo[1]
    matShift = np.float32([[1,0,10],[0,1,10]])# 2*3
    dst = cv2.warpAffine(img, matShift, (width,height))
    while True :
        cv2.imshow("frame",img)
        cv2.imshow('dst', dst)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```



2.4、OpenCV图片镜像

2.4.1、图片镜像的原理

图像的镜像变换分为两种：水平镜像和垂直镜像。水平镜像以图像垂直中线为轴，将图像的像素进行对换，也就是将图像的左半部和右半部对调。垂直镜像则是以图像的水平中线为轴，将图像的上半部分和下班部分对调。

变换原理：

设图像的宽度为width，长度为height。(x,y)为变换后的坐标，(x0,y0)为原图像的坐标

水平镜像变换

向前映射：x=width-x0-1,y=y0

向后映射：x0=width-x-1,y0=y

垂直镜像变换

向上映射：x=x0,y=height-y0-1

向下映射：x0=x，y0=height-y-1

总结：

在水平镜像变换时，遍历了整个图像，然后根据映射关系对每个像素都做了处理。实际上，水平镜像变换就是将图像坐标的列换到右边，右边的列换到左边，是可以以列为单位做变换的。同样垂直镜像变换也如此，可以以行为单位进行变换。

2.4.2、以垂直变换为例，来看下Python是如何写的

运行程序

```
python ~/yahboomcar_ws/src/yahboomcar_astra/scripts/opencv/2_4.py
```

```
import cv2
import numpy as np
if __name__ == '__main__':
    img = cv2.imread('yahboom.jpg')
    imgInfo = img.shape
    height = imgInfo[0]
    width = imgInfo[1]
    deep = imgInfo[2]
    newImgInfo = (height*2,width,deep)
    dst = np.zeros(newImgInfo,np.uint8)#uint8
    for i in range(0,height):
        for j in range(0,width):
            dst[i,j] = img[i,j]
            dst[height*2-i-1,j]   = img[i,j]
    while True :
        cv2.imshow("frame",img)
        cv2.imshow('dst', dst)
        action = cv2.waitKey(10) & 0xFF
        if action == ord('q') or action == 113:
            break
    img.release()
    cv2.destroyAllWindows()
```