信 息 科 学 与 技 术 学 院
SCHOOL OF INFORMATION SCIENCE&TECHNOLOGY

西北大学
Northwest University

计算机视觉

彭盛霖

西北大学信息学院

pengshenglin@nwu.edu.cn

特征检测

3.1 特征检测-边缘检测

罗伯茨(Roberts)算子

```
def getRoberts(img):
    #Roberts算子
    kernelx = np.array([[-1,0],[0,1]], dtype=int)
    kernely = np.array([[0,-1],[1,0]], dtype=int)
    x = cv2.filter2D(img, cv2.CV_16S, kernelx)
    y = cv2.filter2D(img, cv2.CV_16S, kernely)
    absX = cv2.convertScaleAbs(x)
    absY = cv2.convertScaleAbs(y)
    Roberts = cv2.addWeighted(absX,0.5,absY,0.5,0)
    return Roberts
```

3.1 特征检测-边缘检测

普雷维特(Prewitt)算子

```python
def getPrewitt(img):
    #Prewitt算子
    kernelx = np.array([[1,1,1],[0,0,0],[-1,-1,-1]], dtype=int)
    kernely = np.array([[-1,0,1],[-1,0,1],[-1,0,1]], dtype=int)
    x = cv2.filter2D(img, cv2.CV_16S, kernelx)
    y = cv2.filter2D(img, cv2.CV_16S, kernely)
    #转uint8
    absX = cv2.convertScaleAbs(x)
    absY = cv2.convertScaleAbs(y)
    # Prewitt = cv2.addWeighted(absX,0.5,absY,0.5,0)
    Prewitt = (0.5*absX**2.0+0.5*absY**2.0)**0.5
    return cv2.convertScaleAbs(np.uint8(Prewitt))
```

3.1 特征检测-边缘检测

索贝尔(Sobel)算子

```
def getSobel(img):
    #Sobel算子
    kernelx = np.array([[1,2,1],[0,0,0],[-1,-2,-1]], dtype=int)
    kernely = np.array([[-1,0,1],[-2,0,2],[-1,0,1]], dtype=int)
    x = cv2.filter2D(img, cv2.CV_16S, kernelx)
    y = cv2.filter2D(img, cv2.CV_16S, kernely)
    #转uint8
    absX = cv2.convertScaleAbs(x)
    absY = cv2.convertScaleAbs(y)
    Prewitt = (0.5*absX**2.0+0.5*absY**2.0)**0.5
    return cv2.convertScaleAbs(np.uint8(Prewitt))
```

3.1 特征检测-边缘检测

拉普拉斯(Laplacian)算子

```python
def getLaplacian(img):
    #Laplacian算子
    # kernel = np.array([[0,1,0],[1,-4,1],[0,1,0]], dtype=int)
    kernel = np.array([[1,1,1],[1,-8,1],[1,1,1]], dtype=int)
    # kernel = np.array([[-1,2,-1],[2,-4,2],[-1,2,-1]], dtype=int)
    laplacian = cv2.filter2D(img, cv2.CV_16S, kernel)
    return cv2.convertScaleAbs(laplacian)
```

3.1 特征检测-边缘检测

拉普拉斯-高斯(LoG)算子

```
def getLoG(img):
    #LoG算子
    kernel = -np.array([[-2, -4, -4, -4, -2],
    [-4,  0,  8,  0, -4],
    [-4,  8,  24, 8, -4],
    [-4,  0,  8,  0, -4],
    [-2, -4, -4, -4, -2 ]], dtype=int)
    laplacian = cv2.filter2D(img, cv2.CV_16S, kernel)
    return cv2.convertScaleAbs(laplacian)
```

3.1 特征检测-边缘检测

坎尼(Canny)算子

```
img = cv2.imread('catdogN.jpg')
out = cv2.Canny(img,100,200)
cv2.imshow('img',img)
cv2.imshow('out',out)
cv2.waitKey(0)
```

3.1 特征检测-角点检测

ORB角点检测

```
img = cv2.imread('catdog.jpg')
out=np.copy(img)
gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
#SIFT对象创建
orb=cv2.ORB_create()
#进行检测,其中第二个参数为None，表示对整张图进行检测
kp=orb.detect(gray,None)
#进行特征匹配
# kp,des=surf.compute(gray,kp)
kp,des=orb.detectAndCompute(gray,None)
print(des)
#绘制角点
cv2.drawKeypoints(image=out,keypoints=kp,outImage=out,color=(0,255,0))
cv2.imshow('img',img)
cv2.imshow('dst',out)
cv2.waitKey(0)
```

3.1 特征检测-霍夫检测

标准霍夫变换

```python
def line_detection(image):
    # 变换为灰度图
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    # 进行Canny边缘检测
    edge = cv2.Canny(gray, 50, 100, apertureSize=3)
    cv2.imshow("edge", edge)
    # 进行霍夫直线运算
    lines = cv2.HoughLines(edge, 1, np.pi/180., 200)
    for line in lines:# 对检测到的每一条线段
        # 霍夫变换返回的是 r 和 theta 值
        rho, theta = line[0]
        a = np.cos(theta)
        b = np.sin(theta)
        # 确定x0 和 y0
        x0 = a * rho
        y0 = b * rho
        # 构建（x1,y1）,(x2, y2)
        x1 = int(x0 + 1000 * (-b))
        y1 = int(y0 + 1000 * a)
        x2 = int(x0 - 1000 * (-b))
        y2 = int(y0 - 1000 * a)
        # 用cv2.line( )函数在image上画直线
        cv2.line(image, (x1, y1), (x2, y2), (0, 0, 255), 2)
    cv2.imshow("line_detection", image)
    cv2.waitKey(0)
```

3.1 特征检测-霍夫检测

渐进概率式霍夫变换

```python
def line_detectionP(image):
    # 变换为灰度图
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    # 进行Canny边缘检测
    edge = cv2.Canny(gray, 50, 100, apertureSize=3)
    cv2.imshow("edge", edge)
    # 进行霍夫直线运算
    lines = cv2.HoughLinesP(edge, 1.0, np.pi/180., 200, minLineLength=150,maxLineGap=20)
    # 对检测到的每一条线段
    for line in lines:
        for x1,y1,x2,y2 in line:
            cv2.line(image,(x1,y1),(x2,y2),(0,0,255),2)
    cv2.imshow("line_detection", image)
    cv2.waitKey(0)
```

3.1 特征检测-霍夫检测

霍夫圆检测

```python
def HoughCircles(src):
    image = np.array(src)
    cimage = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)  # 灰度图
    circles = cv2.HoughCircles(cimage, cv2.HOUGH_GRADIENT, 1, 40, param1=260, param2=50, minRadius=10)
    cv2.imshow("in", cimage)
    circles = np.uint16(np.around(circles))  # 取整
    for i in circles[0, :]:
        cv2.circle(image, (i[0], i[1]), i[2], (0, 0, 255), 2)  # 在原图上画圆，圆心，半径，颜色，线框
        cv2.circle(image, (i[0], i[1]), 2, (255, 0, 0), 2)  # 画圆心
    # cv2.putText(image, "param1=250, param2=58", (20, 20), cv2.FONT_HERSHEY_SIMPLEX, 0.75, (0, 0, 255), 2)
    cv2.imshow("row_circles", image)
```

◆ 接下来的时间：上机实验并完成实验报告

《计算机视觉》实验报告

- 实验 03：特征检测：边缘、角点与线

姓名		学号	
实验地点		实验日期	

一、实验内容

【1】任选图片，通过 python 编程检测其轮廓，探索如何使其轮廓获得很好的效果。

【2】任选图片，通过 python 编程检测其角点，探索怎样提高其准确率。

【3】任选图片，用霍夫变换检测直线和圆，探索怎样使得获得的效果较佳。