Tedu.cn
达内教育

机器学习

计算机视觉基本理论

DAY02

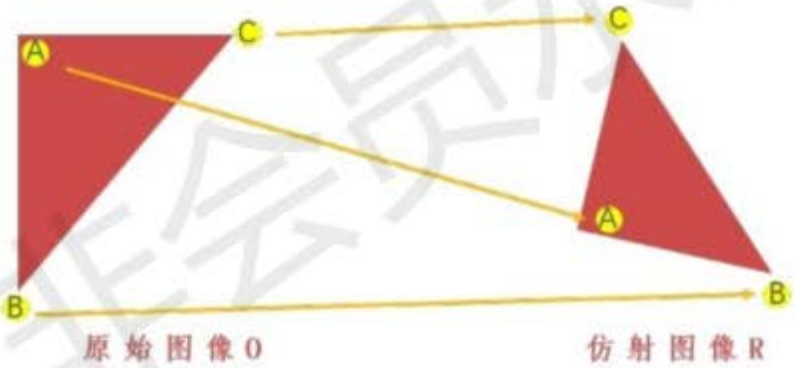
图像形态处理


图像形态处理
仿射与透视变换
图像腐蚀与膨胀
仿射变换
透视变换
图像腐蚀
图像膨胀
图像开运算
图像闭运算
礼帽运算
黑帽运算

仿射与透视变换

仿射变换

- 仿射变换是指图像可以通过一系列的几何变换来实现平移、旋转等多种操作。该变换能够保持图像的平直性和平行性。平直性是指图像经过仿射变换后，直线仍然是直线；平行性是指图像在完成仿射变换后，平行线仍然是平行线。

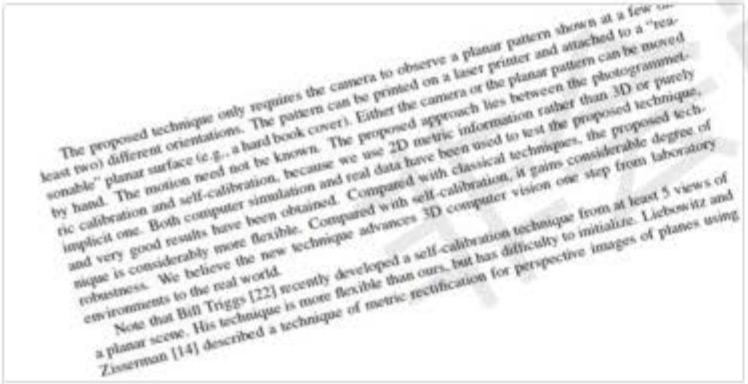


知识讲解

透视变换

- 透视变换是将图片投影到一个新的视平面，也称作投影映射．它是二维（x,y）到三维(X,Y,Z)，再到另一个二维(x′,y′)空间的映射．
- 相对于仿射变换，它提供了更大的灵活性，将一个四边形区域映射到另一个四边形区域（不一定是平行四边形）．透视变换可用于图像形状校正。

知识讲解



The proposed technique only requires the camera to observe a planar pattern shown at a few (at least two) different orientations. The pattern can be printed on a laser printer and attached to a "reasonable" planar surface (e.g., a hard book cover). Either the camera or the planar pattern can be moved by hand. The motion need not be known. The proposed approach lies between the photogrammetric calibration and self-calibration, because we use 2D metric information rather than 3D or purely implicit one. Both computer simulation and real data have been used to test the proposed technique, and very good results have been obtained. Compared with classical techniques, the proposed technique is considerably more flexible. Compared with self-calibration, it gains considerable degree of robustness. We believe the new technique advances 3D computer vision one step from laboratory environments to the real world.

Note that Bill Triggs [22] recently developed a self-calibration technique from at least 5 views of a planar scene. His technique is more flexible than ours, but has difficulty to initialize. Liebowitz and Zisserman [14] described a technique of metric rectification for perspective images of planes using

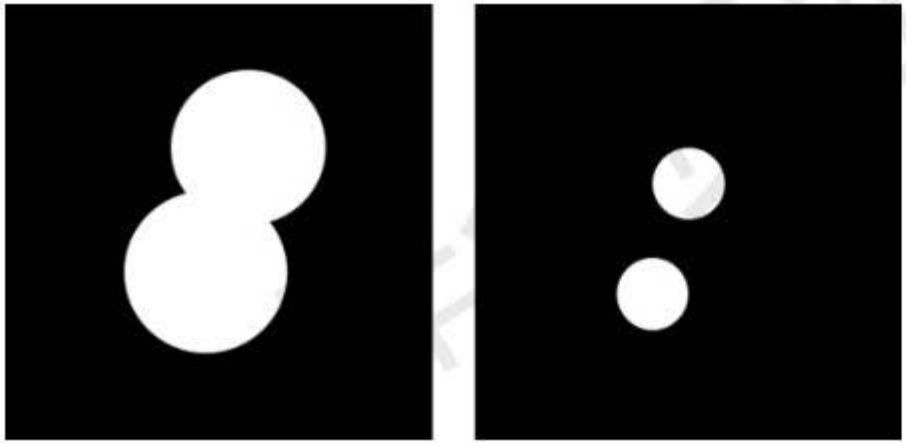
The proposed technique only requires the camera to observe a planar pattern shown at a few (at least two) different orientations. The pattern can be printed on a laser printer and attached to a "reasonable" planar surface (e.g., a hard book cover). Either the camera or the planar pattern can be moved by hand. The motion need not be known. The proposed approach lies between the photogrammetric calibration and self-calibration, because we use 2D metric information rather than 3D or purely implicit one. Both computer simulation and real data have been used to test the proposed technique, and very good results have been obtained. Compared with classical techniques, the proposed technique is considerably more flexible. Compared with self-calibration, it gains considerable degree of robustness. We believe the new technique advances 3D computer vision one step from laboratory environments to the real world.

Note that Bill Triggs [22] recently developed a self-calibration technique from at least 5 views of a planar scene. His technique is more flexible than ours, but has difficulty to initialize. Liebowitz and Zisserman [14] described a technique of metric rectification for perspective images of planes using

图像腐蚀与膨胀

图像腐蚀

- 腐蚀是最基本的形态学操作之一，它能够将图像的边界点消除，使图像沿着边界向内收缩，也可以将小于指定结构体元素的部分去除。腐蚀用来“收缩”或者“细化”二值图像中的前景，借此实现去除噪声、元素分割等功能。



图像膨胀

- 图像膨胀(dilate)是指根据原图像的形状，向外进行扩充。如果图像内两个对象的距离较近，那么在膨胀的过程中，两个对象可能会连通在一起。膨胀操作对填补图像分割后图像内所存在的空白相当有帮助。

图像开运算

- 开运算进行的操作是先将图像腐蚀，再对腐蚀的结果进行膨胀。开运算可以用于去噪、计数等。

知识讲解

图像开运算（续）

- 开运算可用于取出主题图像之间细小的连接

知识讲解

图像闭运算

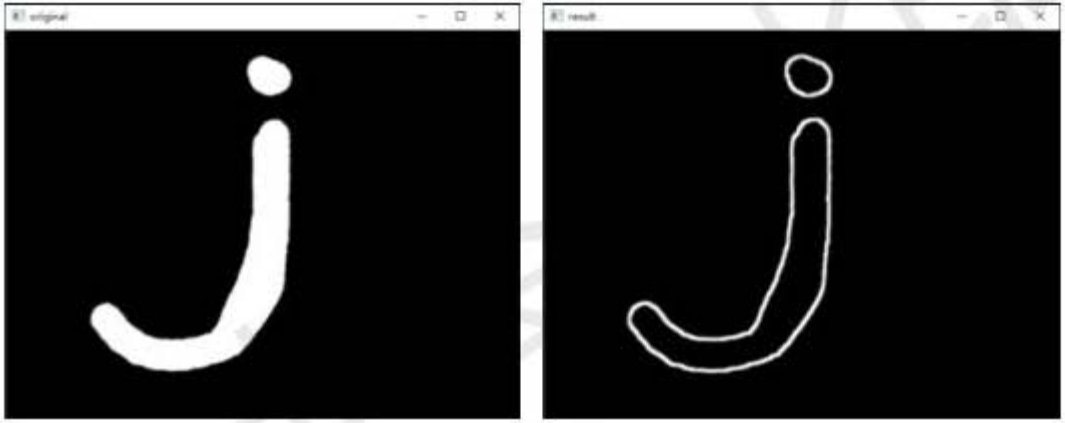
- 闭运算是先膨胀、后腐蚀的运算，它有助于关闭前景物体内部的小孔，或去除物体上的小黑点，还可以将不同的前景图像进行连接。

图像闭运算（续）

知识讲解

形态学梯度

- 形态学梯度运算是用图像的膨胀图像减腐蚀图像的操作，该操作可以获取原始图像中前景图像的边缘。



礼帽运算

- 礼帽运算是用原始图像减去其开运算图像的操作。礼帽运算能够获取图像的噪声信息，或者得到比原始图像的边缘更亮的边缘信息。

知识讲解

黑帽运算

- 黑帽运算是用闭运算图像减去原始图像的操作。黑帽运算能够获取图像内部的小孔，或前景色中的小黑点，或者得到比原始图像的边缘更暗的边缘部分。

知识讲解

图像轮廓

图像轮廓

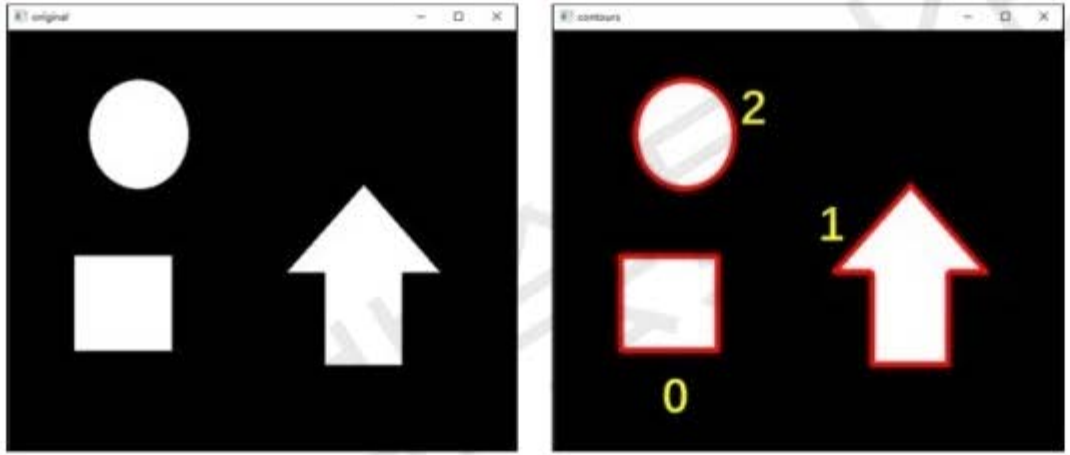
什么是图像轮廓

知识讲解

- 边缘检测虽然能够检测出边缘，但边缘是不连续的，检测到的边缘并不是一个整体。图像轮廓是指将边缘连接起来形成的一个整体，用于后续的计算。
- 图像轮廓是图像中非常重要的一个特征信息，通过对图像轮廓的操作，我们能够获取目标图像的大小、位置、方向等信息。
- 图像轮廓操作包括：查找轮廓、绘制轮廓、轮廓拟合等

查找和绘制轮廓

- 一个轮廓对应着一系列的点，这些点以某种方式表示图像中的一条曲线，将这些点绘制成不同样式的线条，就是轮廓查找与绘制



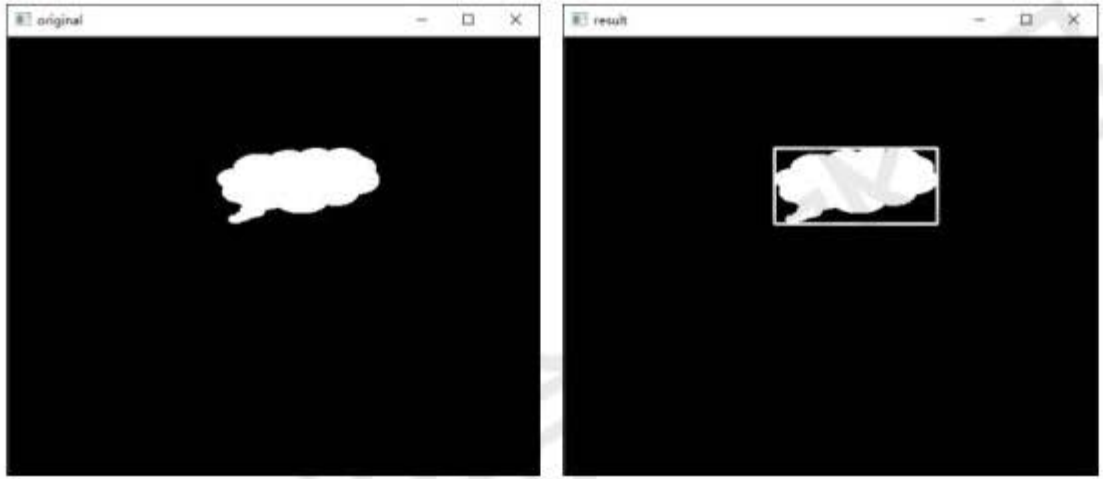
知识讲解

轮廓拟合

- 在计算轮廓时，可能并不需要实际的轮廓，而仅需要一个接近于轮廓的近似多边形，绘制这个近似多边形称之为轮廓拟合

知识讲解

矩形包围框



知识讲解

最小包围圆形

知识讲解

最优拟合椭圆

知识讲解

逼近多边形

(a)

(b)

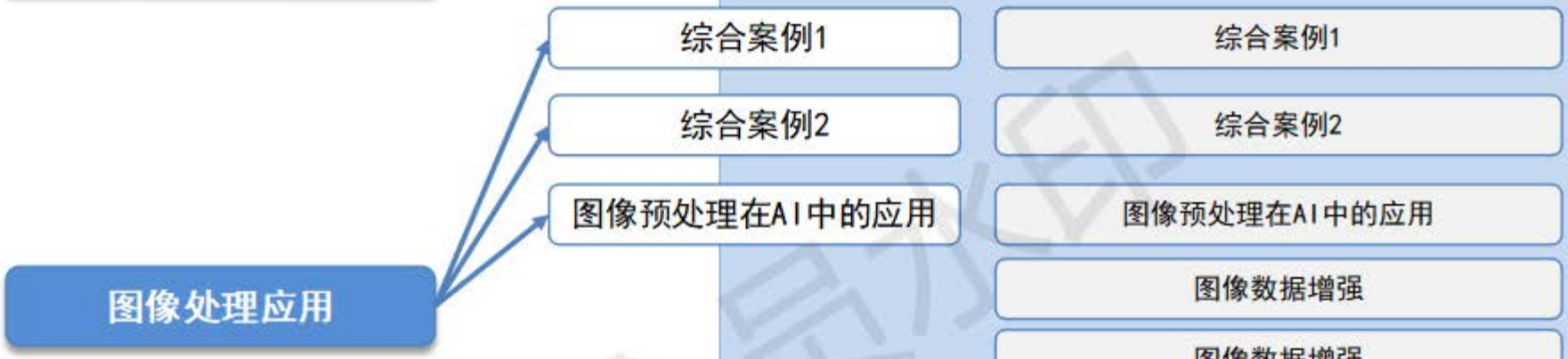
(c)

(d)

(e)

(f)

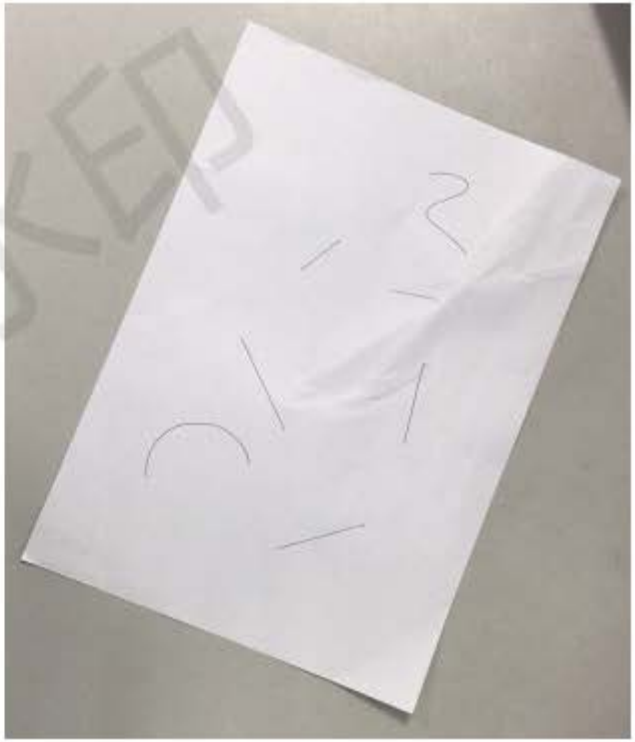
图像处理应用


图像处理应用
综合案例1
综合案例2
图像预处理在AI中的应用
综合案例1
综合案例2
图像预处理在AI中的应用
图像数据增强
图像数据增强
纯图像技术的缺陷

综合案例

综合案例

- 任务描述：我们对图像中的目标进行分析和检测时，目标往往具有一定的倾斜角度，自然条件下拍摄的图像，完全平正是很少的。因此，需要将倾斜的目标“扶正”的过程就就叫做图像矫正。该案例中使用的原始图像如右图所示。



课堂练习

综合案例（续一）

课堂练习

```
# 图像校正示例
import cv2
import imutils
import numpy as np

im = cv2.imread("../data/paper.jpg")
gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
cv2.imshow('im', im)

# 模糊
blurred = cv2.GaussianBlur(gray, (5, 5), 0)
# 膨胀
dilate = cv2.dilate(blurred,
                    cv2.getStructuringElement(cv2.MORPH_RECT, (3, 3)))
# 检测边沿
edged = cv2.Canny(dilate,  # 原始图像
                  30, 120,  # 滞后阈值、模糊度
                  3)  # 孔径大小
```

综合案例（续二）

课堂练习

```
19 # 轮廓检测
20 cnts = cv2.findContours(edged.copy(),
21                         cv2.RETR_EXTERNAL,  # 只检测外轮廓
22                         cv2.CHAIN_APPROX_SIMPLE)  # 只保留该方向的终点坐标
23 cnts = cnts[0] if imutils.is_cv2() else cnts[1]  # 判断是opencv2还是opencv3
24 docCnt = None
25
26 # 绘制轮廓
27 # im_cnt = cv2.drawContours(im,  # 绘制图像
28 #                           cnts,  # 轮廓点列表
29 #                           -1,  # 绘制全部轮廓
30 #                           (0, 0, 255),  # 轮廓颜色：红色
31 #                           2)  # 轮廓粗细
32 # cv2.imshow("im_cnt", im_cnt)
```

综合案例（续三）

课堂练习

```
34 # 计算轮廓面积，并排序
35 if len(cnts) > 0:
36     cnts = sorted(cnts,  # 数据
37                   key=cv2.contourArea,  # 排序依据，根据contourArea函数结果排序
38                   reverse=True)
39     for c in cnts:
40         peri = cv2.arcLength(c, True)  # 计算轮廓周长
41         approx = cv2.approxPolyDP(c, 0.02 * peri, True)  # 轮廓多边形拟合
42         # 轮廓为4个点表示找到纸张
43         if len(approx) == 4:
44             docCnt = approx
45             break
46
47 print(docCnt)
```

综合案例（续四）

课堂练习

```
49 # 用圆圈标记处角点
50 points = []
51 for peak in docCnt:
52     peak = peak[0]
53     # 绘制圆
54     cv2.circle(im,  # 绘制图像
55                tuple(peak), 10,  # 圆心、半径
56                (0, 0, 255), 2)  # 颜色、粗细
57     points.append(peak)  # 添加到列表
58 print(points)
```

综合案例（续五）

课堂练习

```
60 # 校正
61 src = np.float32([points[0], points[1], points[2], points[3]])  # 原来逆时针方向四个点
62 dst = np.float32([[0, 0], [0, 488], [337, 488], [337, 0]])  # 对应变换后逆时针方向四个点
63 m = cv2.getPerspectiveTransform(src, dst)  # 生成透视变换矩阵
64 result = cv2.warpPerspective(gray.copy(), m, (337, 488))  # 透视变换
65 cv2.imshow("result", result)  # 显示透视变换结果
66
67 cv2.waitKey()
68 cv2.destroyAllWindows()
```

综合案例（续六）

- 校正效果

课堂练习

图像预处理在AI中的应用

图像预处理在AI中的应用

- 图像预处理的目的，是让图像数据更适合AI模型进行处理，例如调整大小、颜色
- 通过图像预处理技术，实现数据集的扩充，这种方法称为数据增强。数据增强主要方法有：缩放，拉伸，加入噪点，翻转，旋转，平移，剪切，对比度调整，通道变化。

知识讲解

图像数据增强

知识讲解

原图

通道调整

水平翻转

缩放

拉伸

旋转

噪声

裁剪

纯图像技术的缺陷

- 到目前为止，我们使用的基本是纯图像技术，对图像大小、颜色、形状、轮廓、边沿进行变换和处理，但这些技术都有一个共同的缺点，即无法理解图像内容和场景，要实现这个目标，必须借助于深度学习技术。

知识讲解

今日总结

- 图像形态处理
- 图像轮廓
- 计算机图像技术应用