**เส้น: เพิ่มเติมฟังก์ชัน**

เป้าหมาย

ในบทนี้เราจะเรียนรู้

* ข้อบกพร่องนูนและวิธีการหาพวกเขา
* ค้นหาระยะทางที่สั้นที่สุดจากจุดหนึ่งไปยังรูปหลายเหลี่ยม
* การจับคู่รูปร่างต่างๆ

ทฤษฎีและรหัส

1. ข้อบกพร่องนูน

เราเห็นลำตัวนูนในบทที่สองคืออะไรเกี่ยวกับรูปทรง การเบี่ยงเบนใด ๆ ของวัตถุจากลำตัวนี้ถือได้ว่าเป็นข้อบกพร่องของนูน

OpenCV มาพร้อมกับฟังก์ชั่นพร้อมทำเพื่อหาสิ่งนี้cv2.convexityDefects () สายฟังก์ชันพื้นฐานจะมีลักษณะดังนี้:

hull = cv2.convexHull(cnt,returnPoints = False)

defects = cv2.convexityDefects(cnt,hull)

**บันทึก :** จำได้ว่าเราต้องส่งreturnPoints = Falseในขณะที่หาลำตัวนูนเพื่อหาข้อบกพร่องของนูน

ก็จะส่งกลับอาร์เรย์ที่แต่ละแถวมีค่าเหล่านี้ - [จุดจุดสิ้นสุดจุดที่ไกลที่สุด, ระยะทางโดยประมาณไปยังจุดที่ไกลที่สุดเริ่มต้น] เราสามารถเห็นภาพได้โดยใช้รูปภาพ เราวาดเส้นตรงจุดเริ่มต้นและจุดสิ้นสุดจากนั้นวาดวงกลมที่จุดสุด โปรดจำไว้ว่าค่าสามคนแรกกลับเป็นดัชนีของCNT ดังนั้นเราจึงต้องนำค่าเหล่านั้นจากCNT

**import** **cv2**

**import** **numpy** **as** **np**

img = cv2.imread('star.jpg')

img\_gray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)

ret, thresh = cv2.threshold(img\_gray, 127, 255,0)

contours,hierarchy = cv2.findContours(thresh,2,1)

cnt = contours[0]

hull = cv2.convexHull(cnt,returnPoints = False)

defects = cv2.convexityDefects(cnt,hull)

**for** i **in** range(defects.shape[0]):

s,e,f,d = defects[i,0]

start = tuple(cnt[s][0])

end = tuple(cnt[e][0])

far = tuple(cnt[f][0])

cv2.line(img,start,end,[0,255,0],2)

cv2.circle(img,far,5,[0,0,255],-1)

cv2.imshow('img',img)

cv2.waitKey(0)

cv2.destroyAllWindows()

และดูผล:



2. การทดสอบจุด Polygon

ฟังก์ชันนี้หาระยะห่างที่สั้นที่สุดระหว่างจุดในภาพและเส้นขอบ มันจะส่งกลับระยะทางที่เป็นลบเมื่อจุดอยู่นอกเส้นรูปร่างบวกเมื่อจุดอยู่ภายในและศูนย์ถ้าจุดอยู่ใน contour

ตัวอย่างเช่นเราสามารถตรวจสอบจุด (50,50) ดังนี้:

dist = cv2.pointPolygonTest(cnt,(50,50),True)

ในฟังก์ชั่นที่สามคือการโต้แย้งmeasureDist ถ้าเป็นTrueพบระยะทางที่เซ็นชื่อ ถ้าเป็นเท็จพบว่าจุดนั้นอยู่ด้านในหรือด้านนอกหรือบนเส้น (จะส่งกลับ +1, -1, 0 ตามลำดับ)

**บันทึก :** ถ้าคุณไม่ต้องการหาระยะทางให้แน่ใจว่าอาร์กิวเมนต์ที่สามเป็นเท็จเพราะเป็นกระบวนการที่ต้องใช้เวลามาก ดังนั้นทำให้เท็จให้ประมาณ 2-3X speedup

3. จับคู่รูปทรง

OpenCV มาพร้อมกับฟังก์ชัน**cv2.matchShapes ()**ซึ่งช่วยให้เราสามารถเปรียบเทียบรูปทรงสองรูปหรือสองรูปทรงและแสดงเมตริกที่แสดงถึงความคล้ายคลึงกัน ผลที่ได้จะต่ำกว่าจะดีกว่า คำนวณจากค่า hu-moment วิธีการวัดต่าง ๆ จะอธิบายไว้ในเอกสาร

**import** **cv2**

**import** **numpy** **as** **np**

img1 = cv2.imread('star.jpg',0)

img2 = cv2.imread('star2.jpg',0)

ret, thresh = cv2.threshold(img1, 127, 255,0)

ret, thresh2 = cv2.threshold(img2, 127, 255,0)

contours,hierarchy = cv2.findContours(thresh,2,1)

cnt1 = contours[0]

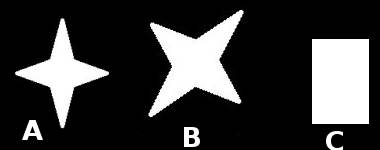
contours,hierarchy = cv2.findContours(thresh2,2,1)

cnt2 = contours[0]

ret = cv2.matchShapes(cnt1,cnt2,1,0.0)

**print** ret

ฉันพยายามจับคู่รูปร่างที่มีรูปร่างแตกต่างกันดังนี้



ฉันได้ผลตาม:

* จับคู่รูปภาพ A กับตัวเอง = 0.0
* จับคู่รูปภาพ A กับรูปภาพ B = 0.001946
* จับคู่รูปภาพ A กับรูปภาพ C = 0.326911

ดูแม้การหมุนภาพจะไม่ส่งผลต่อการเปรียบเทียบนี้มากนัก