**การตรวจจับมุมแฮร์รี่**

เป้าหมาย

ในบทนี้,

* เราจะเข้าใจแนวคิดที่อยู่เบื้องหลังการตรวจจับมุมแฮร์ริส
* เราจะเห็นฟังก์ชัน: **cv2.cornerHarris ()** , **cv2.cornerSubPix ()**

ทฤษฎี

ในบทสุดท้ายเราเห็นว่ามุมเป็นบริเวณในภาพที่มีความรุนแรงมากในทุกทิศทาง คริสแฮร์ริสและไมค์สตีเฟนส์ได้ค้นพบมุมเหล่านี้ในหนังสือพิมพ์มุมรวมและเครื่องตรวจจับขอบในปี 1988 ดังนั้นตอนนี้จึงเรียกว่า Harris Corner Detector เขาใช้ความคิดแบบนี้เป็นแบบคณิตศาสตร์ โดยทั่วไปจะพบความแตกต่างของความเข้มในการเคลื่อนที่ของ(U, V)ทุกทิศทาง นี่แสดงเป็นด้านล่าง:

e (u, v) = \ sum_ {x, y} \ underbrace {w (x, y)} _ \ text {window function} \, [\ underbrace {I (x + u, y + v)} _ \ ข้อความ {ความเข้มขยับขยาย} - \ underbrace {I (x, y)} _ \ text {intensity}] ^ 2

ฟังก์ชันหน้าต่างเป็นหน้าต่างสี่เหลี่ยมผืนผ้าหรือหน้าต่างแบบเกาส์ซึ่งจะให้น้ำหนักกับพิกเซลด้านล่าง

เราต้องเพิ่มฟังก์ชั่นนี้E (U, V)เพื่อการตรวจจับมุม นั่นหมายถึงเราต้องเพิ่มระยะที่สอง ใช้การขยายเทย์เลอร์ไปสู่สมการข้างต้นและใช้ขั้นตอนทางคณิตศาสตร์บางอย่าง (โปรดอ่านหนังสือมาตรฐานที่คุณต้องการสำหรับการมาเต็มรูปแบบ) เราจะได้สมการสุดท้ายดังนี้

E (u, v) \ approx \ begin {bmatrix} u & v \ end {bmatrix} M \ begin {bmatrix} u \\\\ v \ end {bmatrix}

ที่ไหน

M = \ sum_ {x, y} w (x, y) \ begin {bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \ end {bmatrix}

ที่นี่I_xและI_yเป็นอนุพันธ์ภาพในทิศทาง x และ y ตามลำดับ (สามารถหาได้โดยใช้cv2.Sobel () )

จากนั้นมาส่วนหลัก หลังจากนี้พวกเขาสร้างคะแนนโดยทั่วไปสมการซึ่งจะเป็นตัวกำหนดว่าหน้าต่างสามารถมีมุมได้หรือไม่

R = det (M) - k (ร่องรอย (M)) ^ 2

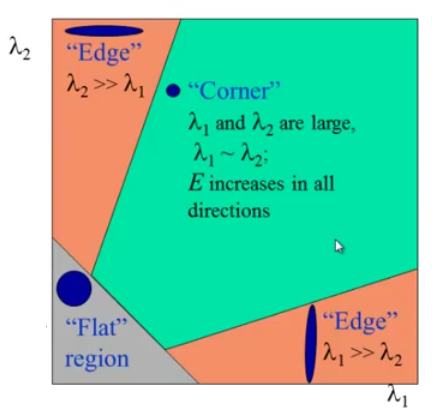
ที่ไหน

* det (M) = \ lambda_1 \ lambda_2
* trace (M) = \ lambda_1 + \ lambda_2
* \ lambda_1และ\ lambda_2เป็นค่า eigen ของ M

ดังนั้นค่าของค่า eigen เหล่านี้จะเป็นตัวกำหนดว่าบริเวณมุมขอบหรือแบน

* เมื่อ| R |มีขนาดเล็กซึ่งเกิดขึ้นเมื่อ\ lambda_1และ\ lambda_2มีขนาดเล็กในภูมิภาคจะแบน
* เมื่อR <0ใดที่เกิดขึ้นเมื่อ\ lambda_1 >> \ lambda_2หรือกลับกันภูมิภาคคือขอบ
* เมื่อRมีขนาดใหญ่ซึ่งเกิดขึ้นเมื่อ\ lambda_1และ\ lambda_2มีขนาดใหญ่และ\ lambda_1 \ sim \ lambda_2ภูมิภาคคือมุม

สามารถแสดงภาพที่สวยงามได้ดังนี้



ดังนั้นผลของ Harris Corner Detection จึงเป็นภาพระดับสีเทาที่มีคะแนนเหล่านี้ การกำหนดเกณฑ์เพื่อให้เหมาะกับมุมของภาพ เราจะทำมันด้วยภาพที่เรียบง่าย

เครื่องตรวจจับมุมแฮร์ริสใน OpenCV

OpenCV มีฟังก์ชันcv2.cornerHarris ()เพื่อการนี้ ข้อโต้แย้งของมันคือ

* img - ใส่ภาพควรเป็นสีเทาและประเภท float32
* blockSize - เป็นขนาดของพื้นที่ใกล้เคียงที่พิจารณาสำหรับการตรวจจับมุม
* ksize - ใช้พารามิเตอร์รูรับแสงของอนุพันธ์ของ Sobel
* k - แฮร์ริสฟรีพารามิเตอร์ของตัวตรวจวัดในสมการ

ดูตัวอย่างด้านล่าง:

**import** **cv2**

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

filename = 'chessboard.jpg'

img = cv2.imread(filename)

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

gray = np.float32(gray)

dst = cv2.cornerHarris(gray,2,3,0.04)

*#result is dilated for marking the corners, not important*

dst = cv2.dilate(dst,None)

*# Threshold for an optimal value, it may vary depending on the image.*

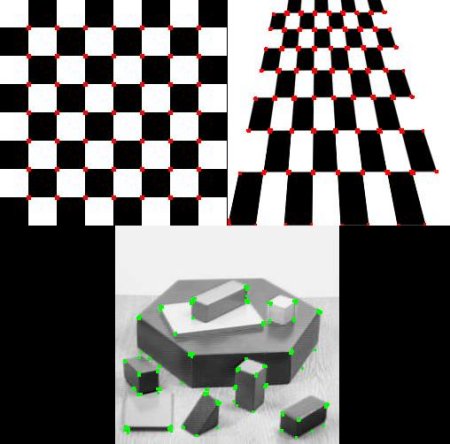
img[dst>0.01\*dst.max()]=[0,0,255]

cv2.imshow('dst',img)

**if** cv2.waitKey(0) & 0xff == 27:

cv2.destroyAllWindows()

ด้านล่างมีสามผลลัพธ์:



มุมที่มีความแม่นยำ SubPixel

บางครั้งคุณอาจต้องหามุมที่มีความแม่นยำสูงสุด OpenCV มาพร้อมกับฟังก์ชั่น**cv2.cornerSubPix ()**ที่ปรับแต่งมุมต่างๆที่ตรวจพบด้วยความถูกต้องของ sub-pixel ด้านล่างเป็นตัวอย่าง ตามปกติแล้วเราต้องหามุมแฮร์ริสก่อน จากนั้นเราจะผ่าน centroids ของมุมเหล่านี้ (อาจมีพวงของพิกเซลที่มุมเราใช้เซนทรอยด์) เพื่อปรับแต่ง มุมแฮร์ริสมีเครื่องหมายพิกเซลสีแดงและมุมที่ผ่านการกลั่นจะมีเครื่องหมายเป็นพิกเซลสีเขียว สำหรับฟังก์ชันนี้เราจะต้องกำหนดเกณฑ์เมื่อจะหยุดการทำซ้ำ เราจะหยุดการทำงานหลังจากทำซ้ำจำนวนครั้งที่ระบุหรือมีความถูกต้องบางอย่างแล้วแต่ว่าจะเกิดขึ้นก่อน เราจำเป็นต้องกำหนดขนาดของพื้นที่ใกล้เคียงที่จะค้นหามุม

**import** **cv2**

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

filename = 'chessboard2.jpg'

img = cv2.imread(filename)

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

*# find Harris corners*

gray = np.float32(gray)

dst = cv2.cornerHarris(gray,2,3,0.04)

dst = cv2.dilate(dst,None)

ret, dst = cv2.threshold(dst,0.01\*dst.max(),255,0)

dst = np.uint8(dst)

*# find centroids*

ret, labels, stats, centroids = cv2.connectedComponentsWithStats(dst)

*# define the criteria to stop and refine the corners*

criteria = (cv2.TERM\_CRITERIA\_EPS + cv2.TERM\_CRITERIA\_MAX\_ITER, 100, 0.001)

corners = cv2.cornerSubPix(gray,np.float32(centroids),(5,5),(-1,-1),criteria)

*# Now draw them*

res = np.hstack((centroids,corners))

res = np.int0(res)

img[res[:,1],res[:,0]]=[0,0,255]

img[res[:,3],res[:,2]] = [0,255,0]

cv2.imwrite('subpixel5.png',img)

ด้านล่างนี้เป็นผลที่ตำแหน่งที่สำคัญบางแห่งจะแสดงในหน้าต่างที่ซูมเพื่อให้เห็นภาพ:

