**อัลกอริธึม FAST สำหรับการตรวจจับมุม**

เป้าหมาย

ในบทนี้,

* เราจะเข้าใจพื้นฐานของขั้นตอนวิธี FAST
* เราจะหามุมโดยใช้ฟังก์ชัน OpenCV สำหรับอัลกอริทึม FAST

ทฤษฎี

เราได้เห็นเครื่องตรวจจับคุณลักษณะหลายอย่างและหลายเครื่องสามารถตรวจจับได้ดีมาก แต่เมื่อมองจากมุมมองของแอพพลิเคชันแบบเรียลไทม์พวกเขายังไม่เร็วพอ ตัวอย่างที่ดีที่สุดประการหนึ่งคือหุ่นยนต์เคลื่อนที่แบบสไลด์ (Simultaneous Localization and Mapping) ที่มีทรัพยากรการคำนวณ จำกัด

จากนั้นจึงนำเสนออัลกอริทึม FAST (คุณลักษณะจากการทดสอบส่วนขยาย) โดย Edward Rosten และ Tom Drummond ในเอกสาร "การเรียนรู้เกี่ยวกับเครื่องสำหรับการตรวจหามุมที่มีความเร็วสูง" ในปี 2006 (แก้ไขในภายหลังในปี 2010) สรุปขั้นพื้นฐานของอัลกอริธึมดังต่อไปนี้ ดูเอกสารต้นฉบับสำหรับรายละเอียดเพิ่มเติม (ภาพทั้งหมดถูกนำมาจากกระดาษต้นฉบับ)

การตรวจจับคุณลักษณะโดยใช้ FAST

1. เลือกพิกเซลพีในรูปภาพซึ่งจะระบุเป็นจุดสนใจหรือไม่ I_pให้ความรุนแรงของมันจะเป็น
2. เสื้อเลือกค่าเกณฑ์ที่เหมาะสม
3. พิจารณาวงกลม 16 พิกเซลรอบ ๆ พิกเซลที่อยู่ระหว่างทดสอบ (ดูภาพด้านล่าง)

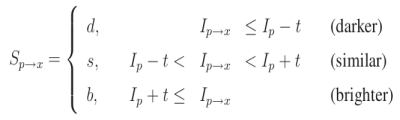


1. ตอนนี้พิกเซลพีเป็นมุมถ้ามีชุดของnพิกเซลที่ต่อเนื่องกันในวงกลม (16 พิกเซล) ซึ่งมีทั้งหมดสว่างกว่าหรือเข้มกว่าทุกI_p + t I_p - t(แสดงเป็นเส้นประสีขาวในภาพด้านบน) nได้รับเลือกให้เป็น 12
2. **ทดสอบความเร็วสูง**ได้รับการเสนอให้ยกเว้นเป็นจำนวนมากไม่ใช่มุม- การทดสอบนี้จะตรวจสอบเฉพาะสี่พิกเซลที่ 1, 9, 5 และ 13 (ก่อน 1 และ 9 จะได้รับการทดสอบถ้าสว่างหรือมืดมากขึ้นถ้าเป็นเช่นนั้นให้ตรวจสอบ 5 และ 13) ถ้าพีเป็นมุมแล้วอย่างน้อยสามเหล่านี้ทั้งหมดจะต้องสว่างกว่าหรือเข้มกว่าI_p + t I_p - tหากไม่เป็นอย่างนี้ก็พีไม่ควรเป็นมุม เกณฑ์การทดสอบกลุ่มแบบเต็มจะสามารถนำไปใช้กับผู้สมัครที่ผ่านการตรวจสอบพิกเซลทั้งหมดในแวดวง เครื่องตรวจจับนี้มีประสิทธิภาพสูง แต่มีจุดอ่อนหลายประการ:
   * ไม่ปฏิเสธผู้สมัครจำนวนมากสำหรับ n <12
   * การเลือกพิกเซลไม่เหมาะสมเพราะประสิทธิภาพขึ้นอยู่กับการเรียงลำดับคำถามและการกระจายรูปลักษณ์ภายนอก
   * ผลของการทดสอบความเร็วสูงจะถูกโยนออกไป
   * ตรวจพบคุณสมบัติหลายอย่างที่อยู่ติดกัน

3 จุดแรกจะถูกจัดการด้วยวิธีการเรียนรู้ด้วยเครื่อง คำตอบสุดท้ายถูกใช้โดยใช้การปราบปรามที่ไม่ใช่แบบสูงสุด

เครื่องตรวจจับมุม

1. เลือกชุดของภาพสำหรับการฝึกอบรม (โดยเฉพาะจากโดเมนแอ็พพลิเคชันเป้าหมาย)
2. เรียกใช้อัลกอริทึม FAST ในทุกรูปเพื่อหาจุดสนใจ
3. สำหรับจุดสนใจทุกจุดเก็บพิกเซล 16 จุดไว้รอบ ๆ เป็นเวกเตอร์ Pทำมันให้ภาพทั้งหมดจะได้รับคุณลักษณะเวกเตอร์
4. แต่ละพิกเซล (พูดx) ใน 16 พิกเซลเหล่านี้อาจมีหนึ่งในสามสถานะต่อไปนี้:



1. ทั้งนี้ขึ้นอยู่กับรัฐเหล่านี้เวกเตอร์คุณลักษณะPแบ่งเป็น 3 P_dส่วนย่อย, P_s, P_b,
2. กำหนดตัวแปรบูลีนใหม่K_pซึ่งเป็นความจริงถ้าพีเป็นมุมและเท็จมิฉะนั้น
3. ใช้อัลกอริธึม ID3 (ตัวจำแนกประเภทต้นไม้ตัดสินใจ) เพื่อค้นหาชุดย่อยแต่ละตัวโดยใช้ตัวแปรK_pสำหรับความรู้เกี่ยวกับคลาสที่แท้จริง มันเลือกที่ทำให้ข้อมูลมากที่สุดเกี่ยวกับว่าพิกเซลผู้สมัครที่เป็นมุมที่วัดโดยเอนโทรปีของxK_p
4. นี่คือ recursively นำไปประยุกต์ใช้กับส่วนย่อยทั้งหมดจนกว่าจะเอนโทรปีเป็นศูนย์
5. โครงสร้างการตัดสินใจที่สร้างขึ้นจะใช้สำหรับการตรวจจับภาพอย่างรวดเร็วในภาพอื่น ๆ

การปราบปรามสูงสุด

การตรวจจับจุดสนใจหลายจุดในบริเวณที่อยู่ติดกันเป็นอีกปัญหาหนึ่ง ได้รับการแก้ไขโดยใช้การปราบปรามแบบไม่สูงสุด

1. คำนวณคะแนนVสำหรับทุกจุดที่ตรวจพบ Vคือผลรวมของค่าสัมบูรณ์ที่แตกต่างระหว่างพี16 และค่าพิกเซลแวดล้อม
2. พิจารณาสอง keypoints ที่อยู่ติดกันและคำนวณVค่าของพวกเขา
3. ทิ้งหนึ่งที่มีVค่าต่ำกว่า

สรุป

เร็วกว่าเครื่องตรวจจับมุมอื่น ๆ ที่มีอยู่หลายเท่า

แต่จะไม่ทำให้ระดับเสียงสูง ขึ้นอยู่กับเกณฑ์

เครื่องตรวจจับคุณลักษณะ FAST ใน OpenCV

เรียกว่าเป็นเครื่องตรวจจับคุณลักษณะอื่น ๆ ใน OpenCV หากต้องการคุณสามารถระบุเกณฑ์ว่าจะใช้การปราบปรามที่ไม่ใช้งานสูงสุดหรือไม่รวมถึงบริเวณที่จะใช้ ฯลฯ

สำหรับพื้นที่ใกล้เคียงสามธงมีการกำหนดcv2.FAST\_FEATURE\_DETECTOR\_TYPE\_5\_8 , cv2.FAST\_FEATURE\_DETECTOR\_TYPE\_7\_12และ cv2.FAST\_FEATURE\_DETECTOR\_TYPE\_9\_16 ด้านล่างเป็นรหัสง่ายๆในการตรวจจับและวาดจุดคุณลักษณะ FAST

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

**import** **cv2**

**from** **matplotlib** **import** pyplot **as** plt

img = cv2.imread('simple.jpg',0)

*# Initiate FAST object with default values*

fast = cv2.FastFeatureDetector()

*# find and draw the keypoints*

kp = fast.detect(img,None)

img2 = cv2.drawKeypoints(img, kp, color=(255,0,0))

*# Print all default params*

**print** "Threshold: ", fast.getInt('threshold')

**print** "nonmaxSuppression: ", fast.getBool('nonmaxSuppression')

**print** "neighborhood: ", fast.getInt('type')

**print** "Total Keypoints with nonmaxSuppression: ", len(kp)

cv2.imwrite('fast\_true.png',img2)

*# Disable nonmaxSuppression*

fast.setBool('nonmaxSuppression',0)

kp = fast.detect(img,None)

**print** "Total Keypoints without nonmaxSuppression: ", len(kp)

img3 = cv2.drawKeypoints(img, kp, color=(255,0,0))

cv2.imwrite('fast\_false.png',img3)

ดูผลการค้นหา ภาพแรกแสดง FAST ด้วย nonmaxSuppression และ second โดยไม่มี nonmaxSuppression:

