**Meanshift และ Camshift**

เป้าหมาย

ในบทนี้,

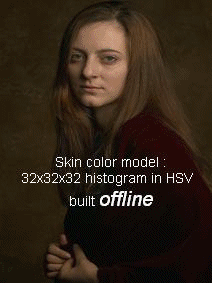
* เราจะเรียนรู้เกี่ยวกับอัลกอริทึม Meanshift และ Camshift เพื่อค้นหาและติดตามวัตถุในวิดีโอ

Meanshift

สัญชาตญาณเบื้องหลังความเรียบง่ายเป็นเรื่องง่าย พิจารณาว่าคุณมีชุดของจุด (อาจเป็นการกระจายพิกเซลเช่น histogram backprojection) คุณจะได้หน้าต่างเล็ก ๆ (อาจเป็นวงกลม) และคุณต้องย้ายหน้าต่างดังกล่าวไปยังพื้นที่ที่มีความหนาแน่นสูงสุดของพิกเซล (หรือจำนวนจุดสูงสุด) มันแสดงให้เห็นในภาพง่ายๆที่ระบุด้านล่าง:



หน้าต่างเริ่มต้นจะแสดงเป็นวงกลมสีฟ้าที่ชื่อ "C1" ศูนย์เดิมถูกทำเครื่องหมายเป็นสี่เหลี่ยมผืนผ้าสีฟ้าชื่อ "C1\_o" แต่ถ้าคุณพบจุดศูนย์กลางของจุดภายในหน้าต่างดังกล่าวคุณจะได้รับ "C1\_r" (ทำเครื่องหมายในวงกลมสีน้ำเงินขนาดเล็ก) ซึ่งเป็นเซนทรอยด์จริงของหน้าต่าง แน่นอนพวกเขาไม่ตรงกับ ดังนั้นย้ายหน้าต่างของคุณเช่นวงกลมของหน้าต่างใหม่ที่ตรงกับ centroid ก่อนหน้านี้ พบ centroid ใหม่อีกครั้ง ส่วนใหญ่แล้วมันจะไม่ตรงกัน ดังนั้นย้ายอีกครั้งและดำเนินการซ้ำตามที่ศูนย์กลางของหน้าต่างและ centroid ตกอยู่ในตำแหน่งเดียวกัน (หรือมีข้อผิดพลาดที่ต้องการเพียงเล็กน้อย) ดังนั้นสิ่งที่คุณได้รับคือหน้าต่างที่มีการกระจายพิกเซลสูงสุด มีเครื่องหมายวงกลมสีเขียวชื่อ "C2" ดังที่คุณเห็นในรูปภาพจะมีจำนวนจุดสูงสุด ขั้นตอนทั้งหมดจะแสดงในภาพนิ่งด้านล่าง:



ดังนั้นเรามักจะผ่าน histogram backprojected image และตำแหน่งเป้าหมายเริ่มต้น เมื่อวัตถุเคลื่อนย้ายได้ชัดการเคลื่อนไหวจะสะท้อนให้เห็นในภาพที่ฉายภาพย้อนหลัง ด้วยเหตุนี้อัลกอริธึมการให้ความคมชัดจะย้ายหน้าต่างของเราไปยังตำแหน่งใหม่ที่มีความหนาแน่นสูงสุด

Meanshift ใน OpenCV

เมื่อต้องการใช้ค่าความหมายใน OpenCV อันดับแรกเราต้องตั้งค่าเป้าหมายค้นหาฮิสโตแกรมเพื่อให้เราสามารถแบ็คโปรเจคเป้าหมายบนเฟรมแต่ละอันเพื่อคำนวณค่า meanhift เราจำเป็นต้องระบุตำแหน่งเริ่มต้นของหน้าต่าง สำหรับฮิสโตแกรมเฉพาะที่นี่เท่านั้น นอกจากนี้เพื่อหลีกเลี่ยงค่าเท็จเนื่องจากแสงน้อยค่าแสงน้อยจะถูกยกเลิกการใช้cv2.inRange ()ฟังก์ชั่น

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

**import** **cv2**

cap = cv2.VideoCapture('slow.flv')

*# take first frame of the video*

ret,frame = cap.read()

*# setup initial location of window*

r,h,c,w = 250,90,400,125 *# simply hardcoded the values*

track\_window = (c,r,w,h)

*# set up the ROI for tracking*

roi = frame[r:r+h, c:c+w]

hsv\_roi = cv2.cvtColor(roi, cv2.COLOR\_BGR2HSV)

mask = cv2.inRange(hsv\_roi, np.array((0., 60.,32.)), np.array((180.,255.,255.)))

roi\_hist = cv2.calcHist([hsv\_roi],[0],mask,[180],[0,180])

cv2.normalize(roi\_hist,roi\_hist,0,255,cv2.NORM\_MINMAX)

*# Setup the termination criteria, either 10 iteration or move by atleast 1 pt*

term\_crit = ( cv2.TERM\_CRITERIA\_EPS | cv2.TERM\_CRITERIA\_COUNT, 10, 1 )

**while**(1):

ret ,frame = cap.read()

**if** ret == True:

hsv = cv2.cvtColor(frame, cv2.COLOR\_BGR2HSV)

dst = cv2.calcBackProject([hsv],[0],roi\_hist,[0,180],1)

*# apply meanshift to get the new location*

ret, track\_window = cv2.meanShift(dst, track\_window, term\_crit)

*# Draw it on image*

x,y,w,h = track\_window

img2 = cv2.rectangle(frame, (x,y), (x+w,y+h), 255,2)

cv2.imshow('img2',img2)

k = cv2.waitKey(60) & 0xff

**if** k == 27:

**break**

**else**:

cv2.imwrite(chr(k)+".jpg",img2)

**else**:

**break**

cv2.destroyAllWindows()

cap.release()

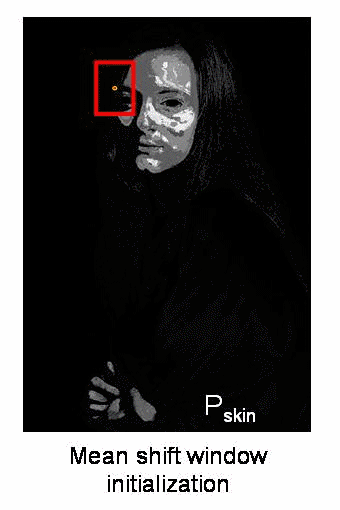
สามเฟรมในวิดีโอที่ฉันใช้มีดังนี้:



Camshift

คุณได้ดูผลสุดท้ายหรือไม่? มีปัญหา หน้าต่างของเรามีขนาดเท่ากันเมื่อรถอยู่ไกลออกไปและอยู่ใกล้กับกล้องมาก ที่ไม่ดี เราจำเป็นต้องปรับขนาดหน้าต่างให้มีขนาดและการหมุนของเป้าหมาย อีกครั้งหนึ่งโซลูชันมาจาก "OpenCV Labs" และเรียกว่า CAMshift (Continuously Adaptive Meanshift) ที่เผยแพร่โดย Gary Bradsky ในบทความเรื่อง "Computer Vision Face Tracking สำหรับใช้ในรูปแบบการรับรู้ของผู้ใช้" ในปี 1988

ใช้ความหมายแรกก่อน เมื่อลู่ meanshift s = 2 \ times \ sqrt {\ frac {M_ {00}} {256}}ก็ปรับปรุงขนาดของหน้าต่างเป็น, นอกจากนี้ยังคำนวณการวางแนวของรูปวงรีที่เหมาะสมกับมัน อีกครั้งใช้ความหมายด้วยหน้าต่างการค้นหาที่ปรับขนาดใหม่และตำแหน่งหน้าต่างก่อนหน้า กระบวนการนี้ดำเนินต่อไปจนกว่าจะมีการตรวจจับความถูกต้อง



Camshift ใน OpenCV

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

**import** **cv2**

cap = cv2.VideoCapture('slow.flv')

*# take first frame of the video*

ret,frame = cap.read()

*# setup initial location of window*

r,h,c,w = 250,90,400,125 *# simply hardcoded the values*

track\_window = (c,r,w,h)

*# set up the ROI for tracking*

roi = frame[r:r+h, c:c+w]

hsv\_roi = cv2.cvtColor(roi, cv2.COLOR\_BGR2HSV)

mask = cv2.inRange(hsv\_roi, np.array((0., 60.,32.)), np.array((180.,255.,255.)))

roi\_hist = cv2.calcHist([hsv\_roi],[0],mask,[180],[0,180])

cv2.normalize(roi\_hist,roi\_hist,0,255,cv2.NORM\_MINMAX)

*# Setup the termination criteria, either 10 iteration or move by atleast 1 pt*

term\_crit = ( cv2.TERM\_CRITERIA\_EPS | cv2.TERM\_CRITERIA\_COUNT, 10, 1 )

**while**(1):

ret ,frame = cap.read()

**if** ret == True:

hsv = cv2.cvtColor(frame, cv2.COLOR\_BGR2HSV)

dst = cv2.calcBackProject([hsv],[0],roi\_hist,[0,180],1)

*# apply meanshift to get the new location*

ret, track\_window = cv2.CamShift(dst, track\_window, term\_crit)

*# Draw it on image*

pts = cv2.boxPoints(ret)

pts = np.int0(pts)

img2 = cv2.polylines(frame,[pts],True, 255,2)

cv2.imshow('img2',img2)

k = cv2.waitKey(60) & 0xff

**if** k == 27:

**break**

**else**:

cv2.imwrite(chr(k)+".jpg",img2)

**else**:

**break**

cv2.destroyAllWindows()

cap.release()

สามเฟรมของผลที่ได้แสดงด้านล่าง:

