**การสกัดด้วย Foreground แบบโต้ตอบโดยใช้ GrabCut Algorithm**

เป้าหมาย

ในบทนี้

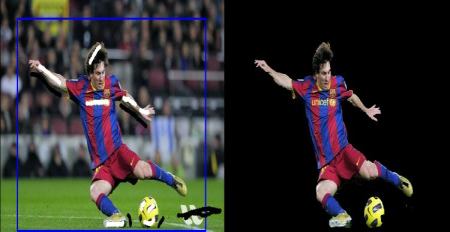
* เราจะเห็นอัลกอริทึม GrabCut เพื่อดึงข้อมูลเด่นในรูปภาพ
* เราจะสร้างแอปพลิเคชันแบบโต้ตอบสำหรับสิ่งนี้

ทฤษฎี

ขั้นตอนวิธี GrabCut ได้รับการออกแบบโดย Carsten Rother, Vladimir Kolmogorov และ Andrew Blake จาก Microsoft Research Cambridge ประเทศอังกฤษ ในกระดาษของพวกเขา[“GrabCut”:](http://dl.acm.org/citation.cfm?id=1015720)การสกัดเบื้องหน้าโต้ตอบโดยใช้การตัดกราฟซ้ำ จำเป็นต้องมีอัลกอริทึมสำหรับการสกัดขั้นต้นโดยมีการโต้ตอบกับผู้ใช้น้อยที่สุดและผลลัพธ์คือ GrabCut

วิธีการทำงานจากมุมมองของผู้ใช้? ในตอนแรกผู้ใช้วาดสี่เหลี่ยมผืนผ้ารอบพื้นที่เบื้องหน้า (พื้นที่ครอบโครงพื้นที่เบื้องหน้าจะอยู่ภายในสี่เหลี่ยมผืนผ้า) จากนั้นอัลกอริทึมจะแบ่งส่วนซ้ำเพื่อให้ได้ผลลัพธ์ที่ดีที่สุด เสร็จสิ้น แต่ในบางกรณีการแบ่งส่วนจะไม่ดีเช่นนี้อาจมีการทำเครื่องหมายพื้นที่เบื้องหน้าเป็นพื้นหลังและในทางกลับกัน ในกรณีนี้ผู้ใช้ต้องทำ touch-ups ปรับ เพียงแค่ให้บางจังหวะในภาพที่มีผลผิดพลาดบางอย่างจะมี จังหวะโดยทั่วไปกล่าวว่า*"เฮ้ภูมิภาคนี้ควรเป็นเบื้องหน้าคุณทำเครื่องหมายว่าเป็นพื้นหลังแก้ไขในการทำซ้ำถัดไป"*หรือตรงกันข้ามกับพื้นหลัง จากนั้นในขั้นตอนถัดไปคุณจะได้ผลลัพธ์ที่ดีขึ้น

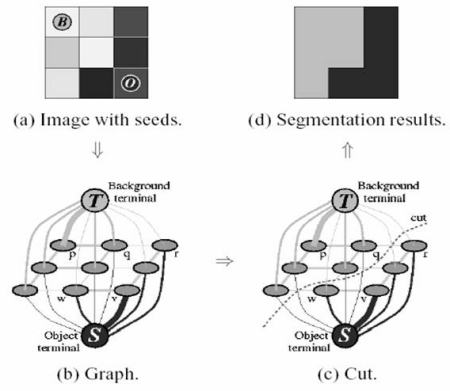
ดูภาพด้านล่าง ผู้เล่นคนแรกและฟุตบอลถูกล้อมรอบด้วยสี่เหลี่ยมผืนผ้าสีน้ำเงิน จากนั้นจะมีการทัชดาวน์ขั้นสุดท้ายด้วยจังหวะขาว (denoting foreground) และ stroke สีดำ (denoting background) และเราได้ผลดี



ดังนั้นสิ่งที่เกิดขึ้นในพื้นหลัง?

* ใส่ข้อมูลผู้ใช้สี่เหลี่ยมผืนผ้า ทุกสิ่งทุกอย่างที่อยู่นอกสี่เหลี่ยมผืนผ้านี้จะถูกนำมาเป็นพื้นหลังที่แน่นอน (นั่นคือเหตุผลที่กล่าวถึงก่อนว่าสี่เหลี่ยมผืนผ้าของคุณควรรวมออบเจ็กต์ทั้งหมดไว้ด้วย) ทุกอย่างภายในสี่เหลี่ยมผืนผ้าไม่เป็นที่รู้จัก ในทำนองเดียวกันการป้อนข้อมูลของผู้ใช้ที่ระบุพื้นหน้าและพื้นหลังจะถือว่าเป็นข้อความที่ติดฉลากยากซึ่งหมายความว่าพวกเขาจะไม่เปลี่ยนแปลงในกระบวนการ
* คอมพิวเตอร์ทำการติดฉลากเริ่มแรกบนข้อมูลที่เราให้ไว้ กำหนดป้ายกำกับเบื้องหน้าและพิกเซลพื้นหลัง (หรือเป็นป้ายกำกับยาก)
* ขณะนี้โมเดล Mixture Gaussian (GMM) ใช้เพื่อจำลองเบื้องหน้าและพื้นหลัง
* GMM ได้เรียนรู้และสร้างการกระจายพิกเซลขึ้นอยู่กับข้อมูลที่เราให้ไว้ นั่นคือพิกเซลที่ไม่รู้จักมีความหมายว่าเป็นเบื้องหน้าหรือพื้นหลังที่เป็นไปได้น่าจะขึ้นอยู่กับความสัมพันธ์กับพิกเซลที่ติดป้ายยากอื่น ๆ ในแง่ของสถิติสี (เช่นเดียวกับการจัดกลุ่ม)
* กราฟถูกสร้างขึ้นจากการกระจายพิกเซลนี้ โหนดในกราฟเป็นพิกเซล สองโหนดเพิ่มเติมมีการเพิ่มโหนดที่มาและโหนดอ่างล้างจาน พิกเซลเบื้องหน้าทั้งหมดเชื่อมต่อกับโหนดต้นทางและทุกพิกเซลพื้นหลังเชื่อมต่อกับโหนด Sink
* น้ำหนักของขอบที่เชื่อมต่อพิกเซลกับโหนดต้นทาง / โหนดปลายทางถูกกำหนดโดยความน่าจะเป็นของพิกเซลที่อยู่เบื้องหน้า / พื้นหลัง น้ำหนักระหว่างพิกเซลจะถูกกำหนดโดยข้อมูลขอบหรือความคล้ายคลึงกันของพิกเซล หากมีสีพิกเซลแตกต่างกันมากขอบระหว่างพวกเขาจะมีน้ำหนักต่ำ
* จากนั้นใช้อัลกอริทึม mincut เพื่อแบ่งส่วนกราฟ จะตัดกราฟออกเป็นโหนดต้นทาง 2 โหนดและโหนดจมกับฟังก์ชันต้นทุนต่ำสุด ฟังก์ชันค่าใช้จ่ายคือผลรวมของน้ำหนักทั้งหมดของขอบที่ถูกตัด หลังจากตัดแล้วพิกเซลทั้งหมดที่เชื่อมต่อกับโหนดต้นทางจะกลายเป็นจุดเริ่มต้นและผู้ที่เชื่อมต่อกับโหนด Sink จะกลายเป็นพื้นหลัง
* กระบวนการนี้ดำเนินไปเรื่อย ๆ จนกระทั่งการจำแนกการจำแนก

ภาพนี้แสดงไว้ในภาพด้านล่าง (รูปภาพมารยาท: <http://www.cs.ru.ac.za/research/g02m1682/> )



Demo

ตอนนี้เราไปสำหรับ algorithm grabcut กับ OpenCV OpenCV มีฟังก์ชันcv2.grabCut ()สำหรับข้อมูลนี้ เราจะเห็นข้อคิดเห็นก่อน:

* img - ใส่รูปภาพ
* หน้ากาก - เป็นภาพหน้ากากที่เราระบุว่าพื้นที่ใดเป็นพื้นหลังพื้นหน้าหรือพื้นหลังน่าจะ / เบื้องหน้า ฯลฯ จะทำโดยธงต่อไปนี้cv2.GC\_BGD, cv2.GC\_FGD, cv2.GC\_PR\_BGD, cv2.GC\_PR\_FGDหรือเพียงแค่ส่งผ่าน 0,1,2,3 ถึงภาพ
* rect - เป็นพิกัดของรูปสี่เหลี่ยมผืนผ้าซึ่งรวมถึงวัตถุเบื้องหน้าในรูปแบบ (x, y, w, h)
* bdgModel , fgdModel - เป็นอาร์เรย์ที่ใช้โดยอัลกอริทึมภายใน คุณเพิ่งสร้างอาร์เรย์เป็นศูนย์ zero.float64 จำนวนสองขนาด (1,65)
* iterCount - จำนวนการทำซ้ำที่อัลกอริธึมควรรัน
* โหมด - ควรcv2.GC\_INIT\_WITH\_RECTหรือcv2.GC\_INIT\_WITH\_MASKหรือรวมกันซึ่งตัดสินใจว่าเรากำลังวาดรูปสี่เหลี่ยมผืนผ้าหรือจังหวะ touchup ขั้นสุดท้าย

ขั้นแรกให้ดูที่รูปสี่เหลี่ยมผืนผ้า เราโหลดภาพสร้างภาพหน้ากากที่คล้ายกัน เราสร้างfgdModelและbgdModel เรากำหนดพารามิเตอร์สี่เหลี่ยมผืนผ้า ทั้งหมดตรงไปข้างหน้า ให้ใช้อัลกอริทึมสำหรับการวนซ้ำ 5 ครั้ง โหมดควรเป็นcv2.GC\_INIT\_WITH\_RECTเนื่องจากเราใช้รูปสี่เหลี่ยมผืนผ้า จากนั้นใช้ grabcut แก้ไขภาพมาสก์ ในภาพหน้ากากใหม่พิกเซลจะถูกทำเครื่องหมายด้วยสี่ธงที่แสดงถึงพื้นหลัง / เบื้องหน้าตามที่ระบุไว้ด้านบน ดังนั้นเราจึงปรับเปลี่ยนรูปแบบมาสก์เพื่อให้ 0 พิกเซลและ 2 พิกเซลถูกวางเป็น 0 (เช่นพื้นหลัง) และทั้งหมด 1 พิกเซลและ 3 พิกเซลจะถูกกำหนดเป็น 1 (พิกเซลเบื้องล่าง) ตอนนี้หน้ากากสุดท้ายของเราพร้อมแล้ว เพียงแค่คูณด้วยภาพนำเข้าเพื่อรับภาพที่แบ่ง

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

**import** **cv2**

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

img = cv2.imread('messi5.jpg')

mask = np.zeros(img.shape[:2],np.uint8)

bgdModel = np.zeros((1,65),np.float64)

fgdModel = np.zeros((1,65),np.float64)

rect = (50,50,450,290)

cv2.grabCut(img,mask,rect,bgdModel,fgdModel,5,cv2.GC\_INIT\_WITH\_RECT)

mask2 = np.where((mask==2)|(mask==0),0,1).astype('uint8')

img = img\*mask2[:,:,np.newaxis]

plt.imshow(img),plt.colorbar(),plt.show()

ดูผลลัพธ์ด้านล่าง:



อ๊ะผมของเมสซี่หายไปแล้ว ใครชอบเมสซี่โดยไม่มีเส้นผม? เราจำเป็นต้องนำมันกลับมา ดังนั้นเราจะให้มีสัมผัสที่ดีกับ 1 พิกเซล (แน่ใจว่าเบื้องหน้า) ในเวลาเดียวกันบางส่วนของพื้นดินได้มาถึงภาพที่เราไม่ต้องการและยังมีโลโก้บางส่วน เราจำเป็นต้องถอดออก ที่นี่เราให้ touchup 0-pixel (พื้นหลังที่แน่นอน) ดังนั้นเราจึงปรับเปลี่ยนหน้ากากที่เกิดขึ้นของเราในกรณีก่อน ๆ ตามที่เราได้แจ้งไว้ในตอนนี้

สิ่งที่ฉันทำจริงก็คือฉันเปิดภาพที่ป้อนเข้าในแอพพลิเคชันการทาสีและเพิ่มเลเยอร์อื่นลงในภาพ การใช้แปรงทาสีผมทำเครื่องหมายเบื้องหน้าเบื้องหลัง (ผมรองเท้าบอล ฯลฯ ) ที่มีพื้นหลังสีขาวและไม่เป็นที่ต้องการ (เช่นโลโก้พื้น ฯลฯ ) พร้อมกับสีดำในเลเยอร์ใหม่นี้ จากนั้นเติมสีพื้นหลังที่เหลือด้วยสีเทา จากนั้นโหลดภาพหน้ากากนั้นใน OpenCV แก้ไขภาพหน้ากากต้นฉบับที่เราได้รับพร้อมกับค่าที่สอดคล้องกันในภาพมาสก์ที่เพิ่มใหม่ ตรวจสอบรหัสด้านล่าง:

*# newmask is the mask image I manually labelled*

newmask = cv2.imread('newmask.png',0)

*# whereever it is marked white (sure foreground), change mask=1*

*# whereever it is marked black (sure background), change mask=0*

mask[newmask == 0] = 0

mask[newmask == 255] = 1

mask, bgdModel, fgdModel = cv2.grabCut(img,mask,None,bgdModel,fgdModel,5,cv2.GC\_INIT\_WITH\_MASK)

mask = np.where((mask==2)|(mask==0),0,1).astype('uint8')

img = img\*mask[:,:,np.newaxis]

plt.imshow(img),plt.colorbar(),plt.show()

ดูผลลัพธ์ด้านล่าง:



ดังนั้นนั่นแหล่ะ ที่นี่แทนการเริ่มต้นในโหมดสี่เหลี่ยมผืนผ้าคุณสามารถเข้าโหมดมาสก์ได้โดยตรง เพียงทำเครื่องหมายบริเวณสี่เหลี่ยมผืนผ้าในรูปหน้ากากด้วยภาพ 2 มิติหรือ 3 พิกเซล (พื้นหลังน่าจะเป็น / เบื้องหน้า) จากนั้นทำเครื่องหมาย sure\_foreground ของเราด้วย 1 พิกเซลตามที่เราได้ทำในตัวอย่างที่สอง จากนั้นใช้ฟังก์ชัน grabCut โดยตรงกับโหมดมาสก์