**Epipolar Geometry**

เป้าหมาย

ในส่วนนี้,

* เราจะเรียนรู้เกี่ยวกับพื้นฐานของเรขาคณิตแบบหลายชั้น
* เราจะเห็นว่า epipole, epipolar lines, epipolar constraint เป็นต้น

แนวคิดพื้นฐาน

เมื่อเราถ่ายภาพด้วยกล้องพินรูทำให้เราหลวมข้อมูลที่สำคัญเช่นความลึกของภาพ หรือระยะห่างของแต่ละจุดในภาพจากกล้องเนื่องจากแต่ละแปลงเป็น 3D-to-2D ดังนั้นจึงเป็นคำถามที่สำคัญว่าเราสามารถหาข้อมูลเชิงลึกโดยใช้กล้องเหล่านี้ได้หรือไม่ และคำตอบคือการใช้กล้องมากกว่าหนึ่งกล้อง ดวงตาของเราทำงานในลักษณะเดียวกันกับที่เราใช้กล้องสองตัว (สองดวง) ซึ่งเรียกว่าสเตอริโอวิชั่น ลองดูว่า OpenCV มีอะไรบ้างในฟิลด์นี้

( การเรียนรู้ OpenCVโดย Gary Bradsky มีข้อมูลมากมายในฟิลด์นี้)

ก่อนที่จะถ่ายภาพเชิงลึกให้เข้าใจแนวคิดขั้นพื้นฐานเบื้องต้นในรูปเรขาคณิตหลายมุม ในส่วนนี้เราจะจัดการกับเรขาคณิต epipolar ดูภาพด้านล่างซึ่งแสดงการตั้งค่าพื้นฐานด้วยกล้องสองตัวที่ถ่ายภาพในฉากเดียวกัน



ถ้าเราใช้เฉพาะกล้องด้านซ้ายเราไม่สามารถหาจุด 3D ตรงกับจุดxในภาพได้เนื่องจากทุกจุดบนเส้นวัวจะพาดไปที่จุดเดียวกันบนระนาบภาพ แต่พิจารณาภาพที่ถูกต้องด้วย ตอนนี้จุดที่แตกต่างกันในวัวโครงการสายไปยังจุดที่แตกต่างกัน ( x') ในเครื่องบินที่ถูกต้อง ดังนั้นด้วยภาพสองภาพนี้เราจึงสามารถคำนวณจุด 3D ที่ถูกต้องได้ นี่คือความคิดทั้งหมด

การฉายจุดต่าง ๆ ในวัวรูปแบบเส้นบนระนาบด้านขวา (เส้นL') เราเรียกมันว่าepilinexสอดคล้องกับจุด หมายถึงเพื่อหาจุดxบนภาพขวาค้นหาตาม epiline นี้ ควรจะอยู่ที่ไหนสักแห่งในบรรทัดนี้ (คิดอย่างนี้เพื่อหาจุดจับคู่ในภาพอื่น ๆ คุณไม่จำเป็นต้องค้นหาภาพทั้งหมดค้นหาเพียงตาม epiline ดังนั้นจึงให้ประสิทธิภาพและความถูกต้องดีขึ้น) นี้เรียกว่าEpipolar จำกัด ในทำนองเดียวกันทุกจุดจะมี epilines ที่สอดคล้องกันในภาพอื่น ๆ เครื่องบินXOO'ที่เรียกว่าEpipolar เครื่องบิน

OและO'เป็นศูนย์กลางของกล้องถ่ายรูป จากการตั้งค่าที่กำหนดข้างต้นคุณสามารถดูการฉายภาพของกล้องที่เหมาะสมที่จะเห็นในภาพซ้ายที่จุด,O' อีมันถูกเรียกว่าepipole Epipole คือจุดตัดกันของเส้นผ่านศูนย์กล้องและเครื่องบินภาพ ในทำนองเดียวกันe'คือ epipole ของกล้องด้านซ้าย ในบางกรณีคุณจะไม่สามารถที่จะหา epipole ในภาพที่พวกเขาอาจจะอยู่นอกภาพ (ซึ่งหมายถึงหนึ่งกล้องไม่เห็นที่อื่น ๆ )

epilines ทั้งหมดผ่าน epipole ของมัน ดังนั้นเพื่อหาตำแหน่งของ epipole เราสามารถหา epilines จำนวนมากและหาจุดตัดกันของพวกเขา

ดังนั้นในเซสชั่นนี้เรามุ่งเน้นไปที่การหาเส้น epipolar และ epipoles แต่การที่จะพบพวกเขาเราต้องสองส่วนผสมเพิ่มเติมMatrix พื้นฐาน (F)และเมทริกซ์ที่จำเป็น (E) Essential Matrix ประกอบด้วยข้อมูลเกี่ยวกับการแปลและการหมุนซึ่งอธิบายถึงตำแหน่งของกล้องตัวที่สองเทียบกับพิกัดส่วนกลางแรก ดูภาพด้านล่าง (ภาพโดย: เรียนรู้ OpenCV โดย Gary Bradsky):



แต่เราชอบการวัดที่จะทำในพิกเซลพิกัดใช่ไหม? เมทริกซ์พื้นฐานมีข้อมูลเช่นเดียวกับเมทริกซ์ที่จำเป็นนอกเหนือไปจากข้อมูลเกี่ยวกับความบังเอิญของกล้องทั้งสองแบบเพื่อให้เราสามารถจับภาพกล้องสองตัวนี้ในพิกเซลพิกัดได้ (ถ้าเรากำลังใช้ภาพที่ได้รับการแก้ไขและปรับจุดให้เป็นปกติโดยหารด้วยความยาวโฟกัสF = E) ในคำง่ายพื้นฐาน Matrix F, แผนที่จุดในภาพหนึ่งไปยังบรรทัด (epiline) ในภาพอื่น ๆ นี่คำนวณจากการจับคู่จุดจากทั้งสองภาพ ต้องใช้จุดอย่างน้อย 8 จุดเพื่อค้นหาเมทริกซ์พื้นฐาน (ในขณะที่ใช้อัลกอริทึม 8 จุด) จุดมากขึ้นเป็นที่ต้องการและใช้ RANSAC เพื่อให้ได้ผลที่มีประสิทธิภาพมากขึ้น

รหัส

ก่อนอื่นเราต้องหาข้อมูลที่ตรงกันระหว่างสองภาพเพื่อค้นหาเมทริกซ์พื้นฐาน สำหรับนี้เราใช้ descriptors SIFT กับ matcher FLANN ตามและการทดสอบอัตราส่วน

**import** **cv2**

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

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

img1 = cv2.imread('myleft.jpg',0) *#queryimage # left image*

img2 = cv2.imread('myright.jpg',0) *#trainimage # right image*

sift = cv2.SIFT()

*# find the keypoints and descriptors with SIFT*

kp1, des1 = sift.detectAndCompute(img1,None)

kp2, des2 = sift.detectAndCompute(img2,None)

*# FLANN parameters*

FLANN\_INDEX\_KDTREE = 0

index\_params = dict(algorithm = FLANN\_INDEX\_KDTREE, trees = 5)

search\_params = dict(checks=50)

flann = cv2.FlannBasedMatcher(index\_params,search\_params)

matches = flann.knnMatch(des1,des2,k=2)

good = []

pts1 = []

pts2 = []

*# ratio test as per Lowe's paper*

**for** i,(m,n) **in** enumerate(matches):

**if** m.distance < 0.8\*n.distance:

good.append(m)

pts2.append(kp2[m.trainIdx].pt)

pts1.append(kp1[m.queryIdx].pt)

ตอนนี้เรามีรายการของการจับคู่ที่ดีที่สุดจากทั้งสองภาพ ลองหาพื้นฐานเมทริกซ์

pts1 = np.int32(pts1)

pts2 = np.int32(pts2)

F, mask = cv2.findFundamentalMat(pts1,pts2,cv2.FM\_LMEDS)

*# We select only inlier points*

pts1 = pts1[mask.ravel()==1]

pts2 = pts2[mask.ravel()==1]

ต่อไปเราจะพบ epilines epilines ที่สอดคล้องกับจุดในภาพแรกจะวาดในภาพที่สอง ดังนั้นการกล่าวถึงภาพที่ถูกต้องมีความสำคัญที่นี่ เราได้รับแถวของเส้น ดังนั้นเราจึงกำหนดฟังก์ชันใหม่เพื่อวาดเส้นเหล่านี้ลงบนภาพ

**def** drawlines(img1,img2,lines,pts1,pts2):

*''' img1 - image on which we draw the epilines for the points in img2*

*lines - corresponding epilines '''*

r,c = img1.shape

img1 = cv2.cvtColor(img1,cv2.COLOR\_GRAY2BGR)

img2 = cv2.cvtColor(img2,cv2.COLOR\_GRAY2BGR)

**for** r,pt1,pt2 **in** zip(lines,pts1,pts2):

color = tuple(np.random.randint(0,255,3).tolist())

x0,y0 = map(int, [0, -r[2]/r[1] ])

x1,y1 = map(int, [c, -(r[2]+r[0]\*c)/r[1] ])

img1 = cv2.line(img1, (x0,y0), (x1,y1), color,1)

img1 = cv2.circle(img1,tuple(pt1),5,color,-1)

img2 = cv2.circle(img2,tuple(pt2),5,color,-1)

**return** img1,img2

ตอนนี้เราพบ epilines ทั้งภาพและวาดภาพ

*# Find epilines corresponding to points in right image (second image) and*

*# drawing its lines on left image*

lines1 = cv2.computeCorrespondEpilines(pts2.reshape(-1,1,2), 2,F)

lines1 = lines1.reshape(-1,3)

img5,img6 = drawlines(img1,img2,lines1,pts1,pts2)

*# Find epilines corresponding to points in left image (first image) and*

*# drawing its lines on right image*

lines2 = cv2.computeCorrespondEpilines(pts1.reshape(-1,1,2), 1,F)

lines2 = lines2.reshape(-1,3)

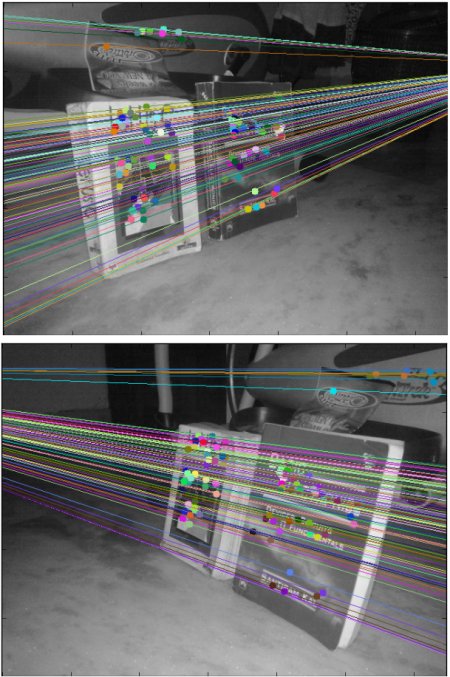
img3,img4 = drawlines(img2,img1,lines2,pts2,pts1)

plt.subplot(121),plt.imshow(img5)

plt.subplot(122),plt.imshow(img3)

plt.show()

ด้านล่างนี้เป็นผลลัพธ์ที่เราได้รับ:



คุณสามารถเห็นในภาพซ้ายว่า epilines ทั้งหมดจะบรรจบกันที่จุดนอกภาพที่ด้านขวา จุดนัดพบคือ epipole

เพื่อให้ได้ผลลัพธ์ที่ดีขึ้นควรใช้ภาพที่มีความละเอียดดีและจุดต่างๆที่ไม่ใช่แนวระนาบ