**การประมาณ**

เป้าหมาย

ในส่วนนี้,

* เราจะเรียนรู้การใช้ประโยชน์จากโมดูล calib3d เพื่อสร้างผล 3D ในภาพ

พื้นฐาน

นี่เป็นส่วนเล็ก ๆ ในช่วงเซสชั่นล่าสุดในการสอบเทียบกล้องคุณได้พบกล้องเมทริกซ์ค่าสัมประสิทธิ์การบิดเบือน ฯลฯ ให้ภาพรูปแบบที่เราสามารถใช้ข้อมูลข้างต้นในการคำนวณท่าทางของตนหรือวิธีการที่วัตถุตั้งอยู่ในพื้นที่เช่นวิธีการที่จะหมุน, วิธีการเคลื่อนย้าย ฯลฯ สำหรับวัตถุระนาบเราสามารถสมมติ Z = 0 เพื่อให้ปัญหากลายเป็นวิธีที่กล้องถูกวางไว้ในอวกาศเพื่อดูรูปแบบของเรา ถ้าเรารู้ว่าวัตถุอยู่ในอวกาศเราสามารถวาดไดอะแกรม 2D เพื่อจำลองผล 3D ได้ ลองดูวิธีการทำ

ปัญหาคือเราต้องการวาดแกนพิกัด 3 มิติของเรา (แกน X, Y, Z) บนมุมแรกของหมากรุกของเรา แกน X เป็นสีน้ำเงินแกน Y ในสีเขียวและแกน Z มีสีแดง ดังนั้นในผลแกน Z ควรรู้สึกเหมือนกับว่ามันตั้งฉากกับเครื่องบินกระดานหมากรุกของเรา

ขั้นแรกให้โหลดเมทริกซ์กล้องและค่าความผิดเพี้ยนจากผลการสอบเทียบก่อนหน้านี้

**import** **cv2**

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

**import** **glob**

*# Load previously saved data*

**with** np.load('B.npz') **as** X:

mtx, dist, \_, \_ = [X[i] **for** i **in** ('mtx','dist','rvecs','tvecs')]

ตอนนี้ขอสร้างฟังก์ชันวาดที่ใช้มุมในกระดานหมากรุก (ได้ใช้cv2.findChessboardCorners () ) และจุดแกนเพื่อวาดแกน 3 มิติ

**def** draw(img, corners, imgpts):

corner = tuple(corners[0].ravel())

img = cv2.line(img, corner, tuple(imgpts[0].ravel()), (255,0,0), 5)

img = cv2.line(img, corner, tuple(imgpts[1].ravel()), (0,255,0), 5)

img = cv2.line(img, corner, tuple(imgpts[2].ravel()), (0,0,255), 5)

**return** img

จากนั้นในกรณีก่อนหน้านี้เราจะสร้างเกณฑ์การสิ้นสุดจุดวัตถุ (มุม 3D ของมุมในกระดานหมากรุก) และจุดแกน จุดแกนคือจุดในพื้นที่ 3D สำหรับการวาดแกน เราวาดแกนความยาว 3 (หน่วยจะอยู่ในรูปแบบของตารางหมากรุกเนื่องจากเราปรับเทียบตามขนาดนั้น) ดังนั้นแกน X ของเราจะถูกดึงจาก (0,0,0) ถึง (3,0,0) ดังนั้นสำหรับแกน Y สำหรับแกน Z จะถูกวาดจาก (0,0,0) ถึง (0,0, -3) ภาพลบบ่งชี้ว่ากล้องถูกลากไปทางกล้อง

criteria = (cv2.TERM\_CRITERIA\_EPS + cv2.TERM\_CRITERIA\_MAX\_ITER, 30, 0.001)

objp = np.zeros((6\*7,3), np.float32)

objp[:,:2] = np.mgrid[0:7,0:6].T.reshape(-1,2)

axis = np.float32([[3,0,0], [0,3,0], [0,0,-3]]).reshape(-1,3)

ขณะนี้เราโหลดภาพแต่ละภาพตามปกติ ค้นหาตาราง 7x6 หากพบเราปรับแต่งด้วยพิกเซล subcorner จากนั้นในการคำนวณการหมุนและการแปลเราจะใช้ฟังก์ชั่นcv2.solvePnPRansac () เมื่อเราเปลี่ยนการฝึกอบรมการแปลงเหล่านี้เราใช้พวกเขาเพื่อจุดแกนของเราไปยังระนาบภาพ ในคำพูดง่ายๆเราจะหาจุดบนระนาบภาพที่สอดคล้องกับแต่ละ (3,0,0), (0,3,0), (0,0,3) ในพื้นที่ 3D เมื่อเราได้รับแล้วเราวาดเส้นจากมุมแรกไปยังจุดต่างๆเหล่านี้โดยใช้ฟังก์ชันdraw () ของเรา ทำ !!!

**for** fname **in** glob.glob('left\*.jpg'):

img = cv2.imread(fname)

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

ret, corners = cv2.findChessboardCorners(gray, (7,6),None)

**if** ret == True:

corners2 = cv2.cornerSubPix(gray,corners,(11,11),(-1,-1),criteria)

*# Find the rotation and translation vectors.*

rvecs, tvecs, inliers = cv2.solvePnPRansac(objp, corners2, mtx, dist)

*# project 3D points to image plane*

imgpts, jac = cv2.projectPoints(axis, rvecs, tvecs, mtx, dist)

img = draw(img,corners2,imgpts)

cv2.imshow('img',img)

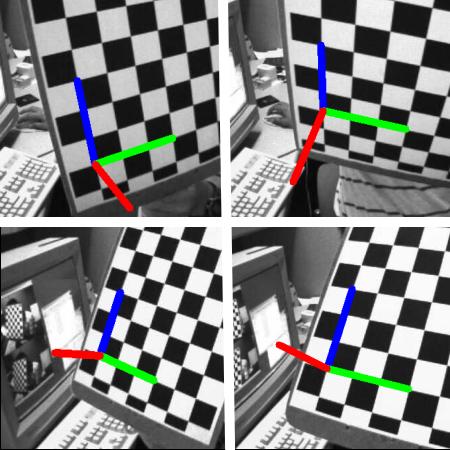
k = cv2.waitKey(0) & 0xff

**if** k == 's':

cv2.imwrite(fname[:6]+'.png', img)

cv2.destroyAllWindows()

ดูผลการค้นหาด้านล่าง สังเกตว่าแต่ละแกนมี 3 สี่เหลี่ยมยาว:



ทำให้ Cube

ถ้าคุณต้องการวาดลูกบาศก์ให้แก้ไขฟังก์ชัน draw () และจุดแกนดังนี้

แก้ไขฟังก์ชันการวาด ():

**def** draw(img, corners, imgpts):

imgpts = np.int32(imgpts).reshape(-1,2)

*# draw ground floor in green*

img = cv2.drawContours(img, [imgpts[:4]],-1,(0,255,0),-3)

*# draw pillars in blue color*

**for** i,j **in** zip(range(4),range(4,8)):

img = cv2.line(img, tuple(imgpts[i]), tuple(imgpts[j]),(255),3)

*# draw top layer in red color*

img = cv2.drawContours(img, [imgpts[4:]],-1,(0,0,255),3)

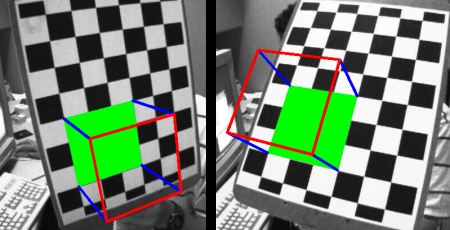
**return** img

แก้ไขจุดแกน พวกเขาเป็น 8 มุมของก้อนในพื้นที่ 3D:

axis = np.float32([[0,0,0], [0,3,0], [3,3,0], [3,0,0],

[0,0,-3],[0,3,-3],[3,3,-3],[3,0,-3] ])

และดูผลลัพธ์ด้านล่าง:



หากคุณสนใจกราฟิก ฯลฯ เติมความเป็นจริงคุณสามารถใช้ OpenGL เพื่อแสดงตัวเลขที่ซับซ้อนมากขึ้นได้