CSCI677 Homework4

1 Introduction

This homework is about SFM. The procedure of SFM has a plenty of processes. Computing keypoint features, computing matches, selecting matches, computing essential matrix, decomposing the essential matrix into a rotation matrix and translation vector, computing 3D point positions. This assignment follows this procedure. It has 3 pairs of images ((a1,a2), (b1,b2), (c1,c2)). The intrinsic matrix is

([[518.86, 0., 285.58],
[0., 519.47, 213.74],
[0., 0., 1.]])

2 Procedure

2.1SIFT feature matching

Description:

Using SIFT method to find the common features and match those features.

We would like to choose the good features by filtering the distances.

```
##############################
#1----SIFT feature matching---#
##############################

#detect sift features for both images
sift = cv2.xfeatures2d.SIFT_create()
kp1, des1 = sift.detectAndCompute(img1,None)
kp2, des2 = sift.detectAndCompute(img2,None)

#use flann to perform feature matching
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)

# store all the good matches as per Lowe's ratio test.
good = []
for m,n in matches:
    if m.distance < 0.7*n.distance:
        good.append(m)

if len(good)>MIN_MATCH_COUNT:
    p1 = np.float32([ kp1[m.queryIdx].pt for m in good ]).reshape(-1,1,2)
    p2 = np.float32([ kp2[m.trainIdx].pt for m in good ]).reshape(-1,1,2)

draw_params = dict(matchColor = (0,255,0), # draw matches in green color
                   singlePointColor = None,
                   flags = 2)

img_siftmatch = cv2.drawMatches(img1,kp1,img2,kp2,good,None,**draw_params)
# cv2.imwrite('../results/sift_match.png',img_siftmatch)
cv2.imwrite('HW4_data/sift_match.png',img_siftmatch)
```

2.2Calculating essential matrix

Description:
CV2.findEssentialMat can be used to calculate the essential matrix. Use RANSAC to calculate the matrix.

```
########################
#2----essential matrix--#
########################
E, mask = cv2.findEssentialMat(p1, p2, K, cv2.RANSAC, 0.999, 1.0);

matchesMask = mask.ravel().tolist()

draw_params = dict(matchColor = (0,255,0), # draw matches in green color
                   singlePointColor = None,
                   matchesMask = matchesMask, # draw only inliers
                   flags = 2)

img_inliermatch = cv2.drawMatches(img1,kp1,img2,kp2,good,None,**draw_params)
cv2.imwrite('HW4_data/inlier_match.png',img_inliermatch)
print "Essential matrix:"
print E
```

2.3Decomposing
Description:
As the requirements mentioned, we need to decompose the essential matrix into Rotation matrix and Translation matrix.

```
####################
#3----recoverpose--#
####################

points, R, t, mask = cv2.recoverPose(E, p1, p2)
print "Rotation:"
print R
print "Translation:"
print t
# p1_tmp = np.expand_dims(np.squeeze(p1), 0)
p1_tmp = np.ones([3, p1.shape[0]])
p1_tmp[:2,:] = np.squeeze(p1).T
p2_tmp = np.ones([3, p2.shape[0]])
p2_tmp[:2,:] = np.squeeze(p2).T
# print (np.dot(R, p2_tmp) + t) - p1_tmp
```

2.4Calculating point position
Description:
This is used to calculate the point position.

```
######################
#4----triangulation---#
######################

#calculate projection matrix for both camera
M_r = np.hstack((R, t))
M_l = np.hstack((np.eye(3, 3), np.zeros((3, 1))))

P_l = np.dot(K,  M_l)
P_r = np.dot(K,  M_r)

# undistort points
p1 = p1[np.asarray(matchesMask)==1,:,:]
p2 = p2[np.asarray(matchesMask)==1,:,:]
p1_un = cv2.undistortPoints(p1,K,None)
p2_un = cv2.undistortPoints(p2,K,None)
```

```
p1_un = np.squeeze(p1_un)
p2_un = np.squeeze(p2_un)

#triangulate points this requires points in normalized coordinate
point_4d_hom = cv2.triangulatePoints(P_l, P_r, p1_un.T, p2_un.T)
point_3d = point_4d_hom / np.tile(point_4d_hom[-1, :], (4, 1))
point_3d = point_3d[:3, :].T
```

2.5Points display

Description:

Ax.scatter can be used to plot all points restored in point_3d list.

```
############################
#5----output 3D pointcloud--#
############################
#TODO: Display 3D points


fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

for point in point_3d:
    ax.scatter(point[0], point[1], point[2], c='r', marker='o')

ax.set_xlabel('X Label')
ax.set_ylabel('Y Label')
ax.set_zlabel('Z Label')

plt.show()
```

3 Results

There are three requirements for this homework

3.1 Output the computed camera matrices, rotation and translation parameters and point positions.

A1 and A2

Essential matrix:

[[0.03528635 -0.64288702 0.00591349]
 [0.49341659 0.02860999 -0.50550657]
 [-0.0264357   0.29126999 0.00798598]]

Rotation:

[[0.93091564 0.04776478 -0.3620975]
 [-0.04716958 0.99883182 0.01048912]
 [0.36217552 0.0073155   0.93208121]]

Translation:

[[-0.41331871]
 [-0.01921728]
 [-0.91038362]]

Point position:

```
[[ 1.0800054   0.95881647 -2.027178  ]
 [ 1.079903    0.95909095 -2.0272336 ]
 [ 1.079904    0.959093   -2.0272405 ]
 [ 1.0799174   0.95916253 -2.0272768 ]
 [ 1.0798175   0.9587379  -2.0271814 ]
 [ 1.0799352   0.9575363  -2.0272884 ]
 [ 1.0799352   0.9575363  -2.0272884 ]
 [ 1.0798333   0.95788336 -2.0272353 ]
 [ 1.0796282   0.95894474 -2.027106  ]
 [ 1.0797504   0.9578568  -2.0272212 ]
 [ 1.0797344   0.9577462  -2.0271788 ]
 [ 1.0795419   0.95881456 -2.0270524 ]
 [ 1.0796636   0.95766515 -2.0271418 ]
 [ 1.0796854   0.9580441  -2.0272384 ]
 [ 1.0794802   0.9586966  -2.0269868 ]
 [ 1.0796473   0.95764023 -2.0271327 ]
 [ 1.0795157   0.95892733 -2.0270922 ]
 [ 1.0795085   0.9591466  -2.0271115 ]
 [ 1.0796179   0.9577089  -2.0271344 ]
 [ 1.0796263   0.95782024 -2.0271742 ]
 [ 1.0795325   0.9596313  -2.0272534 ]
 [ 1.0796207   0.95786667 -2.0271938 ]
 [ 1.0794592   0.9591045  -2.0270784 ]
 [ 1.0794984   0.95951724 -2.0272028 ]
 [ 1.0794729   0.9592996  -2.0271418 ]
 [ 1.0795702   0.95766973 -2.0271125 ]
 [ 1.0793864   0.9587027  -2.0269387 ]
 [ 1.0794622   0.95935154 -2.0271494 ]
 [ 1.0794249   0.9590524  -2.0270555 ]
 [ 1.079549    0.9576112  -2.0271008 ]
 [ 1.0794073   0.9591395  -2.027079  ]
 [ 1.079383    0.95909697 -2.0270464 ]
 [ 1.0793785   0.95915496 -2.0270603 ]
 [ 1.0793785   0.95915496 -2.0270603 ]
 [ 1.0793663   0.9588868  -2.027018  ]
 [ 1.0793537   0.9590192  -2.0270145 ]
 [ 1.079523    0.95781314 -2.0271542 ]
 [ 1.0794929   0.95766735 -2.0270996 ]
 [ 1.0793207   0.9588105  -2.0269637 ]
 [ 1.0793477   0.9591193  -2.0270586 ]
 [ 1.0793673   0.95940983 -2.0271418 ]
 [ 1.0793407   0.9594559  -2.027152  ]
 [ 1.079386    0.9576636  -2.0270407 ]
 [ 1.0791975   0.95829374 -2.0268087 ]
 [ 1.0791825   0.9583388  -2.02679   ]
 [ 1.0791825   0.9583388  -2.02679   ]
 [ 1.0794213   0.9581657  -2.0271754 ]
 [ 1.0792345   0.95906013 -2.0270035 ]
 [ 1.0793638   0.95776224 -2.0270686 ]
 [ 1.0791339   0.9582773  -2.0267687 ]
 [ 1.0791831   0.9587883  -2.0269206 ]
 [ 1.0792642   0.95949775 -2.027145  ]
 [ 1.0793232   0.95804846 -2.0270827 ]
 [ 1.0792966   0.9576685  -2.0270073 ]
 [ 1.0792748   0.9576482  -2.0269938 ]
 [ 1.079279    0.9577068  -2.0270114 ]
 [ 1.0792896   0.95776427 -2.027051  ]
 [ 1.0792109   0.9595061  -2.0271313 ]
 [ 1.0792577   0.9580091  -2.02706   ]
 [ 1.0792465   0.9576825  -2.0269992 ]
 [ 1.0792601   0.9576508  -2.0270205 ]
 [ 1.0792294   0.9578345  -2.0269983 ]
 [ 1.0790616   0.95846164 -2.026794  ]
 [ 1.0790914   0.9587793  -2.0269012 ]
 [ 1.0790915   0.9588402  -2.0269198 ]
 [ 1.0791255   0.95928085 -2.0270443 ]
 [ 1.0790908   0.95903    -2.026989  ]
 [ 1.0791665   0.9580299  -2.0270185 ]
 [ 1.079171    0.9580881  -2.0270495 ]
 [ 1.079106    0.9596393  -2.0271494 ]
 [ 1.0790128   0.9588067  -2.0269    ]
 [ 1.0790128   0.9588067  -2.0269    ]
 [ 1.0791162   0.96004385 -2.0272381 ]
 [ 1.0791162   0.96004385 -2.0272381 ]
 [ 1.078998    0.9584326  -2.0268548 ]
 [ 1.0791261   0.9600979  -2.0273037 ]
 [ 1.079082    0.95773154 -2.0269513 ]
 [ 1.0789851   0.9583921  -2.0269186 ]
 [ 1.0789255   0.95871776 -2.0268562 ]
 [ 1.0789531   0.9591175  -2.0269647 ]
 [ 1.0788909   0.95860654 -2.026824  ]
 [ 1.0788947   0.95868796 -2.026847  ]
 [ 1.0790262   0.9579061  -2.0269935 ]
 [ 1.0788467   0.95851624 -2.0268009 ]
 [ 1.079       0.9578035  -2.0269847 ]
 [ 1.079       0.9578035  -2.0269847 ]
 [ 1.0789701   0.95800537 -2.0269613 ]
 [ 1.0788336   0.9585581  -2.0267959 ]
 [ 1.0789726   0.9580161  -2.026986  ]
 [ 1.0788726   0.9590347  -2.0269382 ]
 [ 1.078878    0.95825434 -2.026893  ]
 [ 1.0788547   0.95844865 -2.0268748 ]
```

[1.078931 0.95793915 -2.0269473]
[1.0789428 0.9577878 -2.0269604]
[1.0789056 0.9576614 -2.026896]
[1.0789056 0.95787233 -2.0269346]
[1.0788332 0.9578564 -2.0269191]
[1.0788332 0.9578564 -2.0269191]
[1.0787975 0.95764726 -2.0268548]
[1.0787618 0.95713276 -2.0268016]
[1.0787414 0.9577814 -2.0268843]
[1.0787749 0.9580569 -2.0269797]
[1.0787269 0.95802414 -2.0269303]
[1.078705 0.9579975 -2.0269165]
[1.0786761 0.9577006 -2.0268667]
[1.0786693 0.9580424 -2.0269136]
[1.0786693 0.9580424 -2.0269136]
[1.0786057 0.9583203 -2.02685   ]
[1.0785997 0.95853215 -2.02688   ]
[1.0786645 0.95811135 -2.0269492]
[1.0786145 0.95773023 -2.0268548]
[1.0786158 0.95773023 -2.0268595]
[1.0785731 0.9584152 -2.0268767]
[1.0786294 0.95813113 -2.02697   ]
[1.078521 0.95846474 -2.0268576]
[1.078559 0.9576504 -2.0268266]
[1.0785651 0.9580699 -2.026917]
[1.078493 0.95843315 -2.0268621]
[1.0784949 0.9581393 -2.0268939]
[1.0784411 0.958036 -2.0269127]
[1.0783794 0.9580311 -2.0269215]
[1.0782496 0.9588087 -2.0268972]
[1.0782514 0.95833695 -2.026864]
[1.0782338 0.9579809 -2.0268252]
[1.078246 0.957676 -2.0269272]
[1.078151 0.9584116 -2.0268583]
[1.078112 0.9586501 -2.02686   ]
[1.0780728 0.9587791 -2.0269203]
[1.0780679 0.95736885 -2.0267842]
[1.0780586 0.95741993 -2.0267868]
[1.0780703 0.9576045 -2.0268297]
[1.0780418 0.9587291 -2.0269125]
[1.0780432 0.95734423 -2.0267875]
[1.0780306 0.9588599 -2.0269449]
[1.0780306 0.9588599 -2.0269449]
[1.0780172 0.958826 -2.026934]
[1.0779998 0.95856476 -2.0268927]
[1.0779896 0.9586592 -2.0269248]
[1.077987 0.9588339 -2.0269532]
[1.0779543 0.9580719 -2.0268312]
[1.0779641 0.9587197 -2.0269423]
[1.0779641 0.9587197 -2.0269423]
[1.0779724 0.95901924 -2.0269856]
[1.0779581 0.95882016 -2.0269547]
[1.0779581 0.95882016 -2.0269547]
[1.0779282 0.95863205 -2.0269237]
[1.0779185 0.9586806 -2.02694   ]
[1.0778784 0.95831543 -2.0268986]
[1.0778841 0.958681 -2.0269568]
[1.0779086 0.95908266 -2.0270388]
[1.0779086 0.95908266 -2.0270388]
[1.0778804 0.9587218 -2.0269644]
[1.0778813 0.9587611 -2.0269692]
[1.0778673 0.95903933 -2.0270195]
[1.0778414 0.95866746 -2.026953]
[1.0778232 0.95870954 -2.0269582]
[1.0778115 0.95818675 -2.0268912]
[1.0777812 0.95807666 -2.0268419]
[1.0777739 0.95856804 -2.0269623]
[1.077778 0.9591319 -2.0270736]
[1.0777509 0.9589647 -2.0270321]
[1.0777402 0.95874864 -2.0270243]
[1.0777204 0.95827377 -2.0269856]
[1.077634 0.9579806 -2.026889]
[1.0776415 0.95814234 -2.0269325]
[1.0776415 0.95814234 -2.0269325]
[1.077667 0.9588529 -2.0270464]
[1.0776513 0.9585613 -2.0270097]
[1.0776124 0.9587073 -2.0270593]
[1.0776199 0.9588747 -2.0270884]
[1.0775899 0.9588749 -2.0270681]]

B1 and B2

Essential matrix:

[[0.02594348 -0.59135081 -0.27692398]
 [0.64062882 -0.06831098 -0.07001068]
 [0.2957447 0.23831386 0.09334175]]

Rotation:
[[0.94272111 -0.10665497 0.31607219]
 [0.10759025 0.99408894 0.0145439]
 [-0.31575505 0.02029545 0.94862366]]

Translation:
[[0.38195359]
 [-0.40006709]
 [0.83310131]]

Point position:
[[1.0214605   0.7952379 -1.8506521]
 [1.0214579   0.7949538 -1.8506483]
 [1.0214483   0.79491544 -1.8506522]
 [1.0214264   0.7949306 -1.8506433]
 [1.0213745   0.7953943 -1.8506107]
 [1.0213938   0.7949163 -1.8506353]
 [1.0213586   0.79540706 -1.850611]
 [1.0213586   0.79540706 -1.850611]
 [1.0213573   0.7952363 -1.8506285]
 [1.0213256   0.7953129 -1.8506165]
 [1.0213356   0.79493964 -1.8506284]
 [1.0212996   0.795298   -1.8506113]
 [1.0212755   0.79531765 -1.8505903]
 [1.0212762   0.7950118 -1.8506049]
 [1.0212762   0.7950118 -1.8506049]
 [1.0211996   0.7954845 -1.8505164]
 [1.0212384   0.7949007 -1.850622]
 [1.0212384   0.7949007 -1.850622]
 [1.0212092   0.7950802 -1.8505914]
 [1.021185   0.7951202 -1.8505759]
 [1.0211889   0.7949381 -1.850603]
 [1.0211571   0.7951128 -1.8505713]
 [1.0211629   0.7949241 -1.8506055]
 [1.021141   0.79503775 -1.8505801]
 [1.0211588   0.7948932 -1.8506107]
 [1.0211099   0.7948628 -1.8506079]
 [1.021013   0.79522353 -1.8505881]
 [1.0209605   0.79513884 -1.8505862]
 [1.0209605   0.79513884 -1.8505862]
 [1.0209559   0.795111   -1.8505896]
 [1.0209031   0.7954361 -1.8505244]
 [1.0209031   0.7954361 -1.8505244]
 [1.0209423   0.7951557 -1.8505808]
 [1.0209129   0.79533726 -1.850542]
 [1.0209322   0.795099   -1.8505863]
 [1.0209413   0.7949678 -1.8506105]
 [1.0209167   0.7951224 -1.8505758]
 [1.0208596   0.7953913 -1.8505243]
 [1.0208924   0.7951003 -1.8505865]
 [1.0208924   0.7951003 -1.8505865]
 [1.020868   0.7951857 -1.8505554]
 [1.020868   0.7951857 -1.8505554]
 [1.0208876   0.79496753 -1.8505946]
 [1.0208727   0.7950345 -1.8505837]
 [1.0208216   0.79513896 -1.8505657]
 [1.0208089   0.7952612 -1.8505647]
 [1.0208752   0.79400474 -1.8506293]
 [1.0207765   0.7953217 -1.8505355]
 [1.020692   0.79541427 -1.8505098]
 [1.0207087   0.7952646 -1.850546]
 [1.0207133   0.7950358 -1.8505685]
 [1.0206536   0.7954953 -1.8504939]
 [1.0206392   0.7953117 -1.8505212]
 [1.0206263   0.79528147 -1.850528]
 [1.0206263   0.79528147 -1.850528]
 [1.0205728   0.7955358 -1.8504803]
 [1.0205781   0.79547024 -1.8505007]
 [1.0204448   0.7947469 -1.8503999]
 [1.0204265   0.79537886 -1.850507]
 [1.0204406   0.79517365 -1.8505377]
 [1.0204238   0.79520327 -1.8505337]
 [1.0204238   0.79520327 -1.8505337]
 [1.0204269   0.7951304 -1.8505459]
 [1.0204208   0.7950997 -1.8505518]
 [1.0204313   0.7950076 -1.8505661]
 [1.0204026   0.7951762 -1.8505356]
 [1.020357   0.7955313 -1.8504834]
 [1.0203991   0.79515916 -1.8505446]

[1.0203912 0.79501474 -1.8505588]
[1.0203815 0.79511285 -1.8505491]
[1.020371   0.7950586 -1.8505564]
[1.0203178 0.795227   -1.850534]
[1.0203091 0.7950898 -1.8505543]
[1.0203091 0.7950898 -1.8505543]
[1.020274   0.7943815 -1.8505105]
[1.0202502 0.79501724 -1.8505592]
[1.0201728 0.7952502 -1.8505187]
[1.0201826 0.794736   -1.8505208]
[1.0201726 0.79512215 -1.8505371]
[1.0201492 0.7946228 -1.8504745]
[1.0201324 0.7938398 -1.8504299]
[1.0201534 0.7948297 -1.8505293]
[1.020165   0.7949947 -1.8505691]
[1.0201477 0.79475665 -1.8505365]
[1.0201312 0.7948634 -1.8505505]
[1.0201224 0.79497015 -1.8505552]
[1.0201224 0.79497015 -1.8505552]
[1.0200357 0.79457915 -1.8505973]
[1.0197582 0.79505205 -1.8505689]
[1.0197816 0.79434955 -1.8505824]
[1.0197816 0.79434955 -1.8505824]
[1.019754   0.7947447 -1.8505732]
[1.0197495 0.7946711 -1.8505802]
[1.0197443 0.7947253 -1.8505847]
[1.0197381 0.7947597 -1.8505781]
[1.0197649 0.7943218 -1.8506151]
[1.0197259 0.79477745 -1.8505723]
[1.0196866 0.7947688 -1.8506014]
[1.0196751 0.79482895 -1.8505896]
[1.0196664 0.7948576 -1.8505911]
[1.0196909 0.79451084 -1.8506252]
[1.0196925 0.7943972 -1.8506227]
[1.0196729 0.7947107 -1.8506126]
[1.0196635 0.7948592 -1.8506134]
[1.0196569 0.7947753 -1.8506258]
[1.0196335 0.7948708 -1.850598]
[1.0196043 0.7948318 -1.8506422]
[1.0195906 0.7947548 -1.8506459]
[1.0195798 0.79442734 -1.8506877]
[1.019538   0.79479676 -1.850651]
[1.019518   0.79466414 -1.8506374]
[1.0195447 0.79447156 -1.850701]
[1.0195057 0.794801   -1.8506793]
[1.0195404 0.79398495 -1.8507406]
[1.0194651 0.7946753 -1.8506726]
[1.0195004 0.79401535 -1.8507785]
[1.0194361 0.79471636 -1.8507073]
[1.0194361 0.79471636 -1.8507073]
[1.0193648 0.7947902 -1.8507605]
[1.0193963 0.7940232 -1.8508178]
[1.01937   0.79425573 -1.8508266]
[1.019281   0.79488856 -1.8507764]
[1.019281   0.79488856 -1.8507764]
[1.0192984 0.79406095 -1.8508825]]

C1 and C2

Essential matrix:

```
[[ 0.00473978  0.31054486  0.00603249]
 [ 0.39721785 -0.03937412  0.58356854]
 [-0.02344382 -0.6340427  -0.03253751]]
```

Rotation:

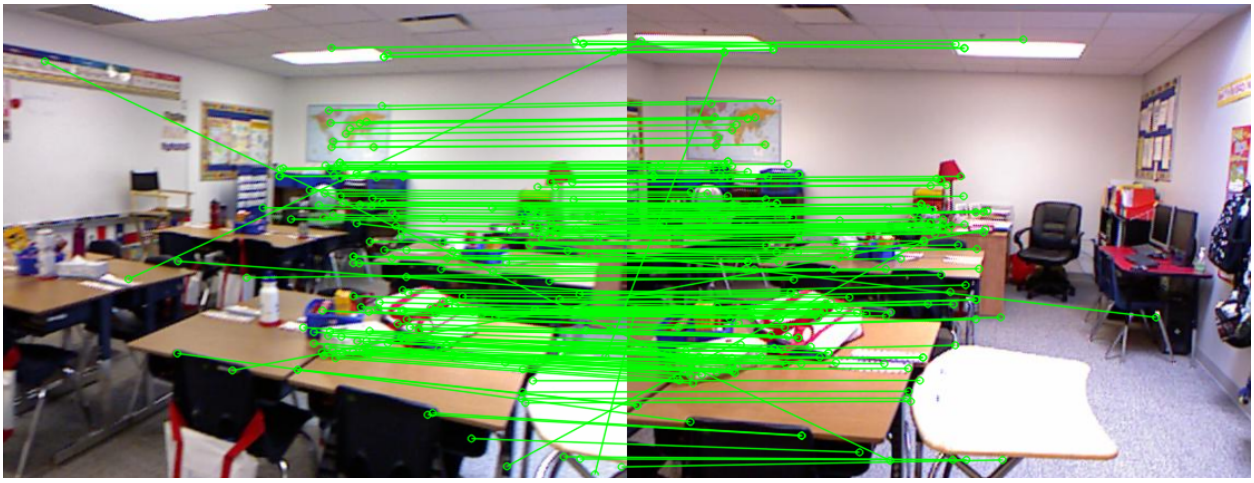
```
[[ 0.98971631 -0.01233748 -0.14251108]
 [-0.02016788 -0.99835704 -0.05363276]
 [-0.14161524  0.05595537 -0.98833907]]
```

Translation:

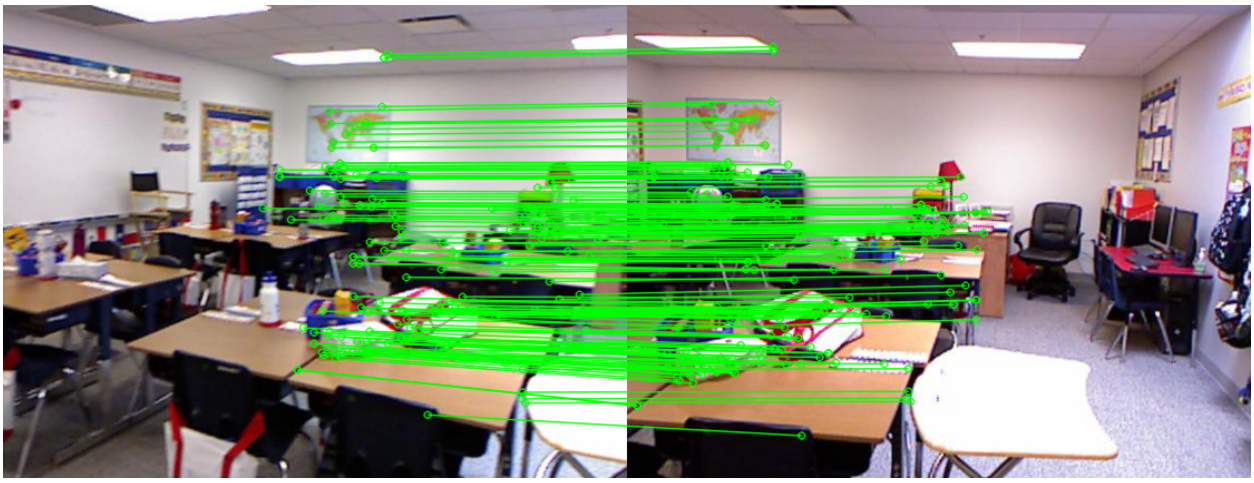
```
[[0.89833516]
 [0.01519329]
 [0.43904795]]
```

Point position:
[[-0.52305204 -0.3118095 0.9719722]
 [-0.5226809 -0.31111246 0.9716656]
 [-0.5225481 -0.31127498 0.97203225]
 [-0.5225415 -0.31115067 0.97205335]
 [-0.52257204 -0.31132177 0.972132]
 [-0.52256423 -0.3113337   0.97215796]
 [-0.5224394 -0.3113255   0.9721747]
 [-0.52242434 -0.31136072 0.9722294]
 [-0.5224644 -0.31140193 0.97230756]
 [-0.52242094 -0.31133613 0.9722297]
 [-0.52242094 -0.31133613 0.9722297]
 [-0.5224733 -0.31125873 0.9723335]
 [-0.52239645 -0.31136516 0.97251767]
 [-0.5223769 -0.31158283 0.9726685]
 [-0.52235866 -0.31141186 0.97264504]
 [-0.5223581 -0.31160498 0.97271675]
 [-0.52225006 -0.31142476 0.97279185]
 [-0.5222408 -0.31147888 0.9727998]]

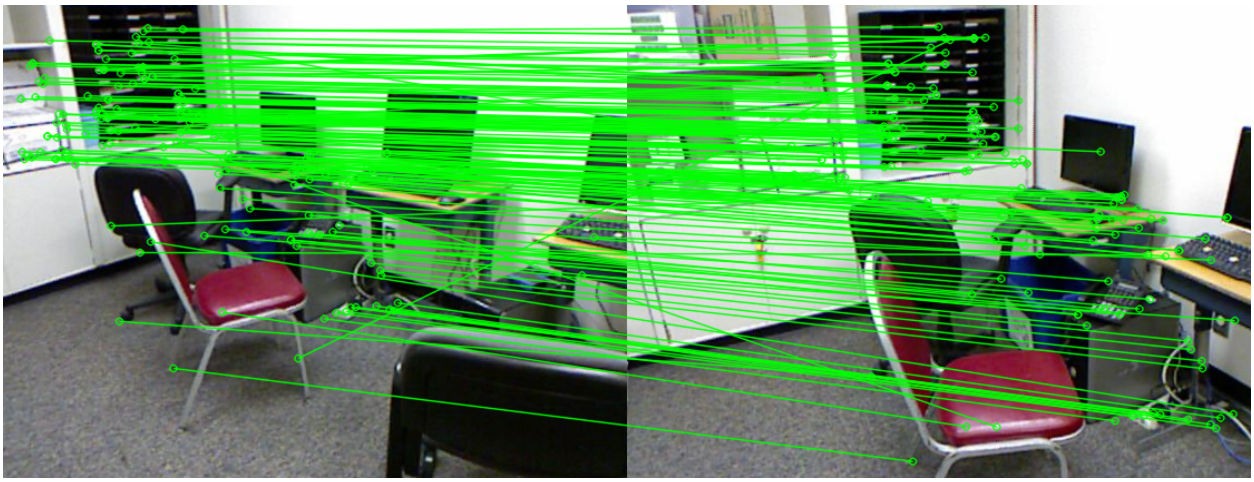
3.2 Display some of the matches graphically, before and after applying RANSAC.
A1 and A2 Before



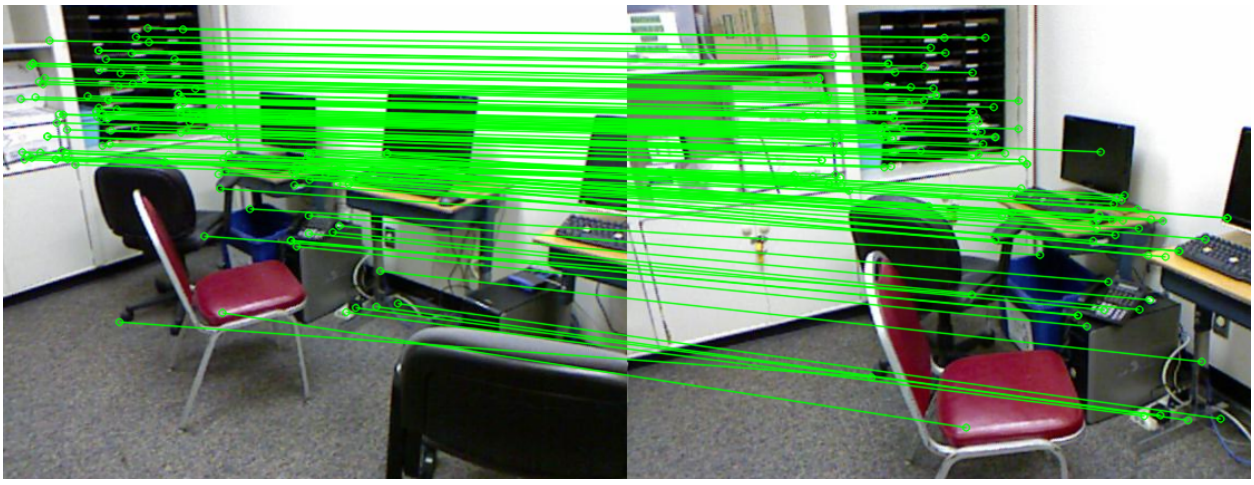
A1 and A2 After



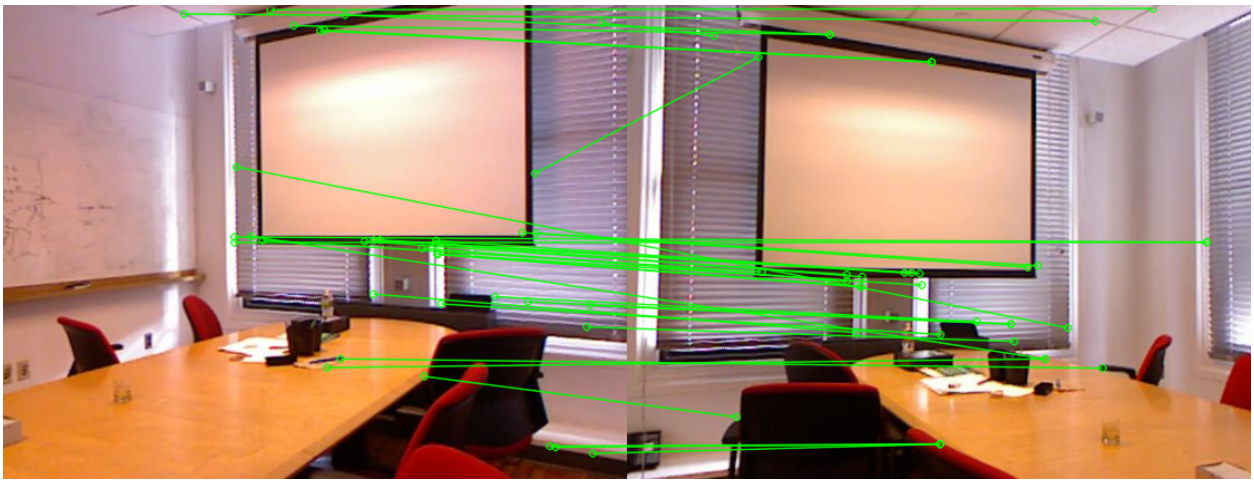
B1 and B2 Before



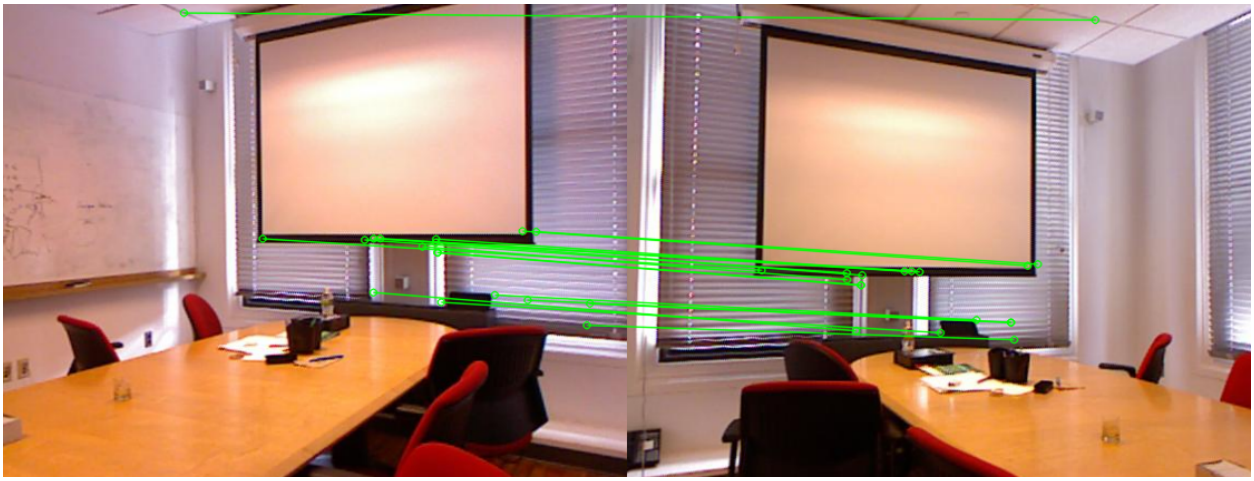
B1 and B2 After



C1 and C2 Before

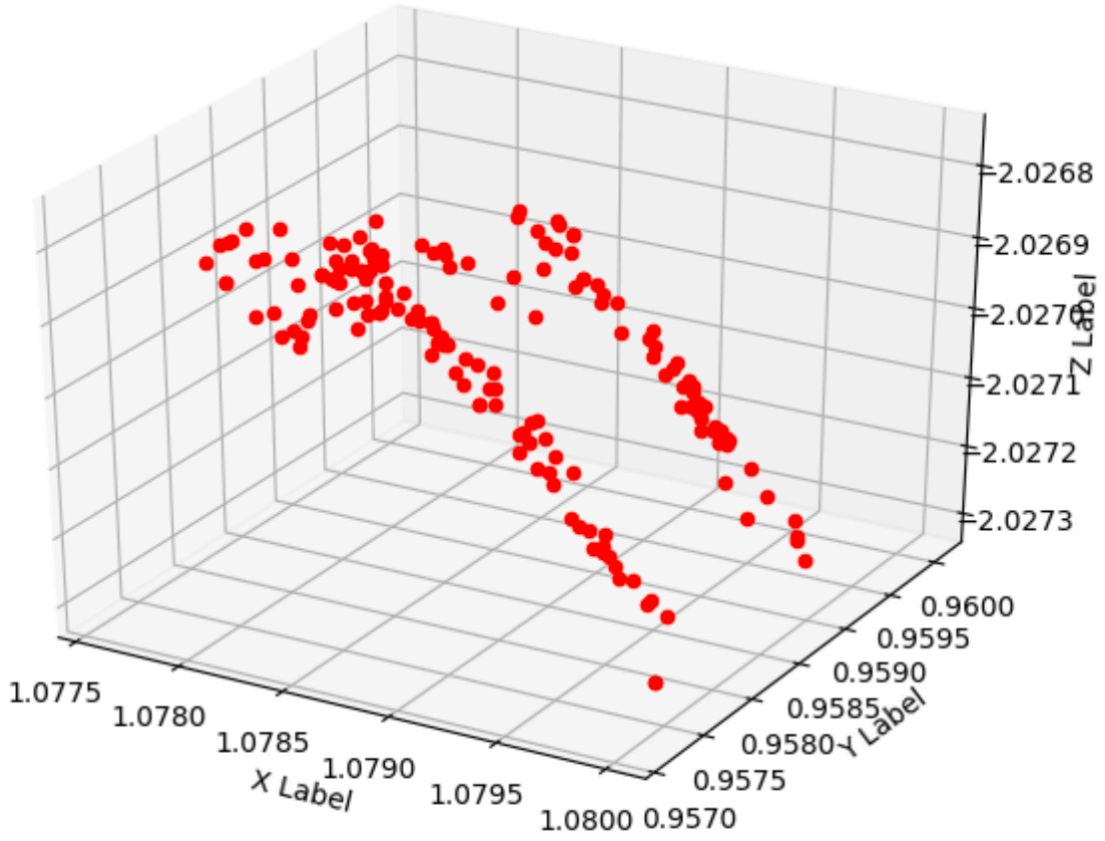


C1 and C2 After

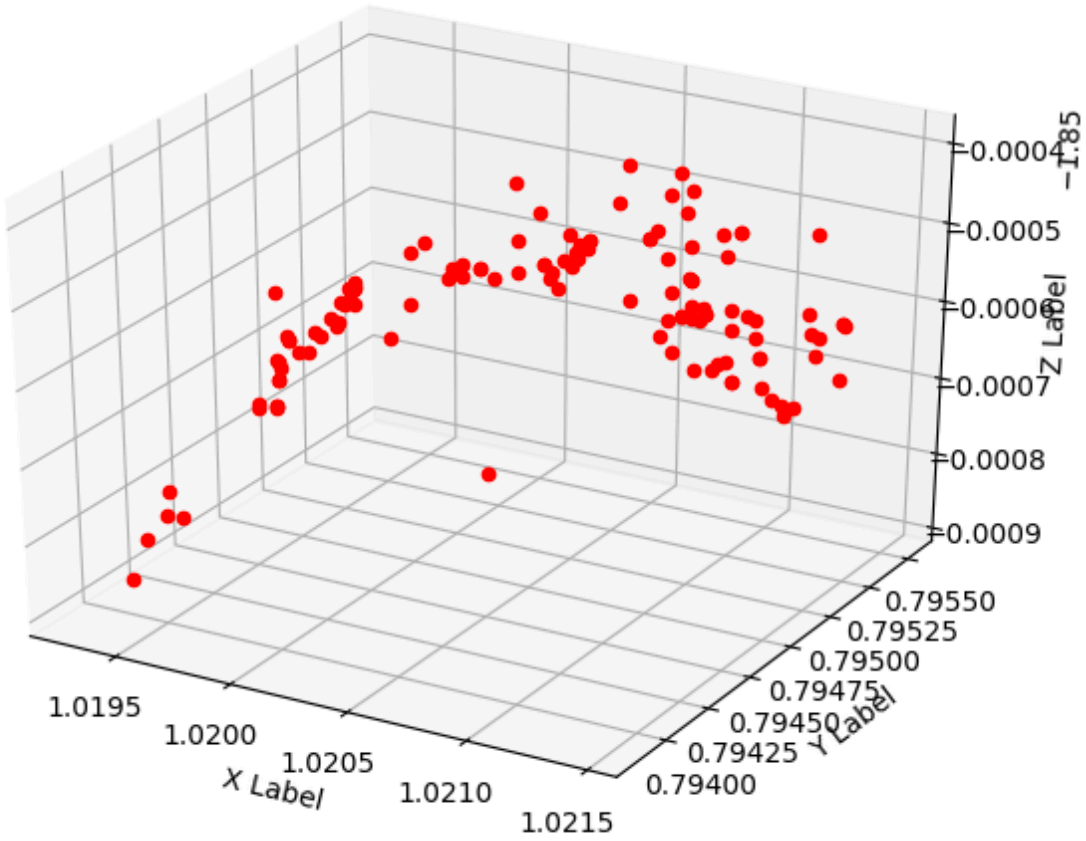


3.3 Display the reconstructed 3-D points from a viewpoint other than that of the original images.

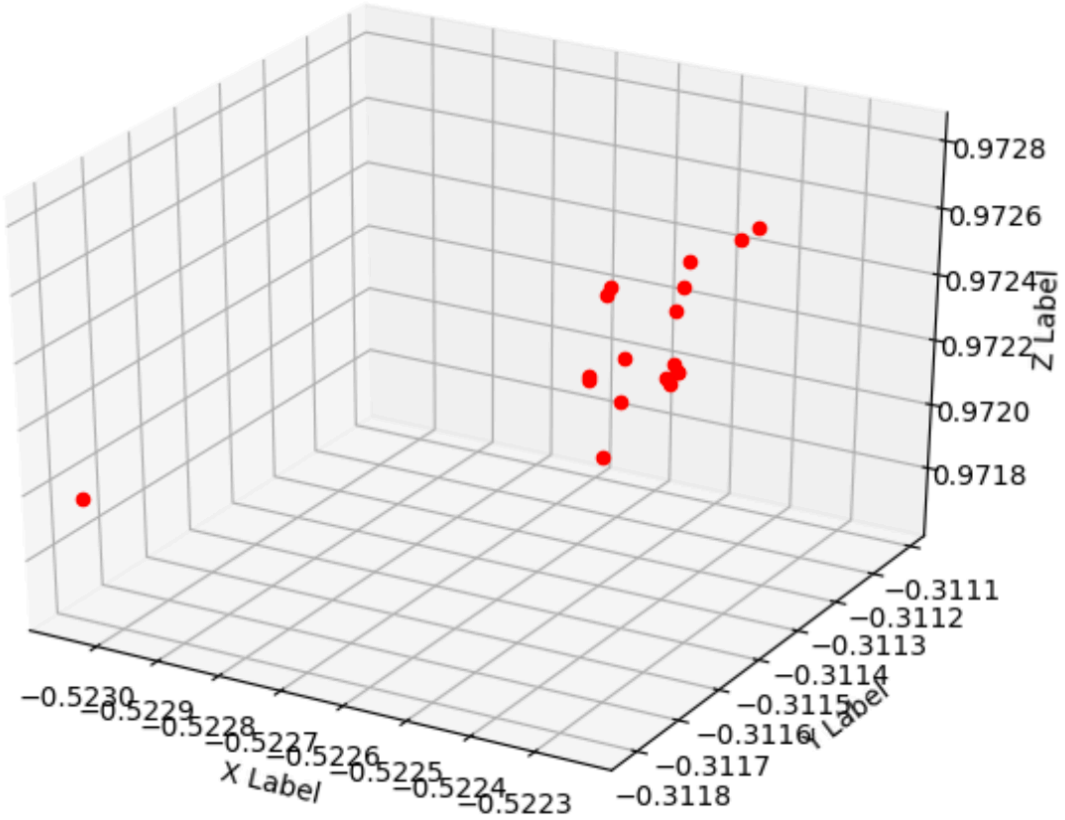
A1 and A2


2.0268
2.0269
2.0270
2.0271
2.0272
2.0273
Z Label
0.9600
0.9595
0.9590
0.9585
0.9580
0.9575
0.9570
Y Label
1.0775
1.0780
1.0785
1.0790
1.0795
1.0800
X Label

B1 and B2


1.0195
1.0200
1.0205
1.0210
1.0215
X Label
0.79550
0.79525
0.79500
0.79475
0.79450
0.79425
0.79400
Y Label
0.0004
0.0005
0.0006
0.0007
0.0008
0.0009
Z Label

C1 and C2



4 Analysis

We would like to know how well does the method work? For A1 and A2, we can see that there are a lot of matched points and inliers matches. It has a good result. For B1 and B2, it also obtains a good result. However, this time, the camera changes a little bit. The camera moves slight down. For C1 and C2, it gets a bad result. The camera gets two pictures from totally different angles. Before RANSIC, it gets some matches, but those matches are not really good. After RANSIC, just a little number of inliers are maintained. Based on those observation, the keys of obtaining a good result are 1) to have more overlaps between two pictures; 2) to move the camera slowly; 3) don't change the bearing angles.