Clab-3 Report

ENGN6528-Master

Tianjun Peng

UID 6761542

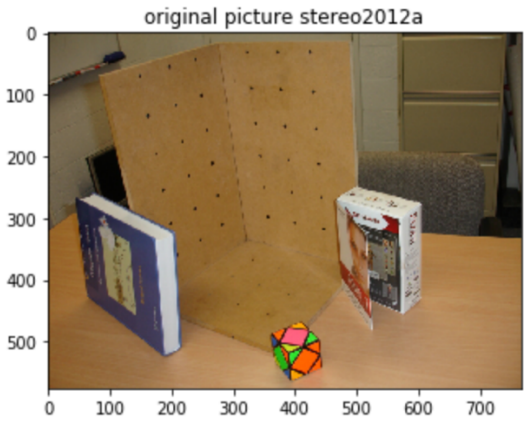
12/10/2020

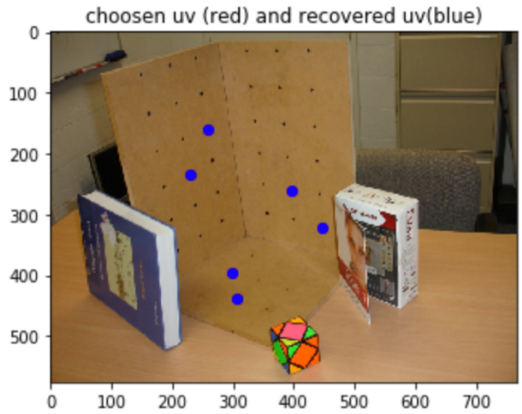
Task 1

1. List the calibrate function in your PDF file. (2 marks)

```
def calibrate(im, XYZ, uv):
    # TBD
    C = None
    n = len(uv)
    b = uv.reshape((2*n, 1))
    X,Y,Z = XYZ[:, 0],XYZ[:, 1],XYZ[:, 2]
    list_A = []
    for i in range(n):
        list_A.append([X[i], Y[i], Z[i], 1, 0, 0, 0, 0, -X[i]*uv[i, 0], -Y[i]*uv[i, 0], -Z[i]*uv[i, 0]])
        list_A.append([0, 0, 0, 0, X[i], Y[i], Z[i], 1, -X[i]*uv[i, 1], -Y[i]*uv[i, 1], -Z[i]*uv[i, 1]])
    A = np.asarray(list_A)

    q = np.linalg.lstsq(A, b, rcond='warn')[0]
    q = np.vstack((q, [1]))

    C = q.reshape((3, 4))

    X = X.reshape((1, len(XYZ)))
    Y = Y.reshape((1, len(XYZ)))
    Z = Z.reshape((1, len(XYZ)))

    u = (q[0]*X + q[1]*Y + q[2]*Z + q[3]) / (q[8]*X + q[9]*Y + q[10]*Z + 1)
    v = (q[4]*X + q[5]*Y + q[6]*Z + q[7])/ (q[8]*X + q[9]*Y + q[10]*Z + 1)
    plt.scatter(uv[:, 0], uv[:, 1], color='r')
    plt.scatter(u, v, color='b')
    plt.title('choosen uv (red) and recovered uv(blue)')
    plt.imshow(im)

    return C
```

2. List which image you have chosen for your experiment and display the annotated image in your PDF file.





We could see the real world projection is almost 100% accuracy projecting on the image, since the blue point cover red point entirely.

3. **List the 3x4 camera calibration matrix C that you have calculated. (1 mark)**

```
[[ 4.51903566e+00 -1.85426723e+00 -5.79524068e+00  3.26881053e+02]
 [ 5.83983814e-01 -7.18750997e+00  2.20688082e+00  3.31391363e+02]
 [-3.34394684e-03 -2.47305024e-03 -3.53024315e-03  1.00000000e+00]]
```

4. **Decompose the C matrix into K, R, t, such that C = K[R|t], by using the following provided code (vgg_KR_from_P.m or vgg_KR_from_P.py). List the results in your PDF file. (0.5 marks)**

```
Intrinsic matrix [[ 1.36889690e+03 -1.05012906e+02  8.16252239e+02]
 [ 0.00000000e+00  1.43660411e+03  6.46545010e+02]
 [ 0.00000000e+00  0.00000000e+00  1.00000000e+00]]
Rotation matrix [[ 0.89612056  0.03747937 -0.44222533]
 [ 0.34727235 -0.67966846  0.6461058 ]
 [-0.27635097 -0.73256132 -0.62208043]]
Transformation matrix [133.86964634 117.16527608 119.02252655]
```

5. Answer the following questions:

a. What is the focal length of the camera? (0.75 marks)

The focal length is the distance between the pinhole and the image plane. In this case, from the intrinsic matrix above we know the focal length fx = 1369 and fy = 1437

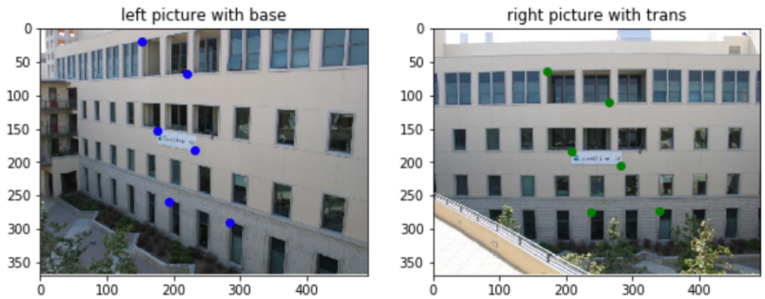
b. What is the pitch angle of the camera with respect to the X-Z plane in the world coordinate system?

From the intrinsic matrix s = -105 = tanθ, so θ = -89.45 degree.

Task3

1. **List your source code for homography estimation and display the two images and the location of six pairs of selected points. (2 marks)**

```
def homography(u2Trans, v2Trans, uBase, vBase):
    H = None
    n = len(uBase)
    list_A = []
    for i in range(n):
        list_A.append([uBase[i],vBase[i],1,0,0,0,-u2Trans[i]*uBase[i],-u2Trans[i]*vBase[i],-u2Trans[i]])
        list_A.append([0,0,0,uBase[i],vBase[i],1,-v2Trans[i]*uBase[i],-v2Trans[i]*vBase[i],-v2Trans[i]])
    A = np.asarray(list_A)
    u, sigma, vt = np.linalg.svd(A)
    H = vt.T[:,-1].reshape((3,3))
    return H
```



2. List the 3x3 camera homography matrix H that you have calculated. (1 mark)

```
[[-1.45931765e-02 -8.74062629e-05  9.99757609e-01]
 [-2.27605268e-03 -6.64935491e-03  1.41905958e-02]
 [-1.73656041e-05  5.22102668e-07 -4.58120373e-03]]
```