Machine Vision – Assignment 2

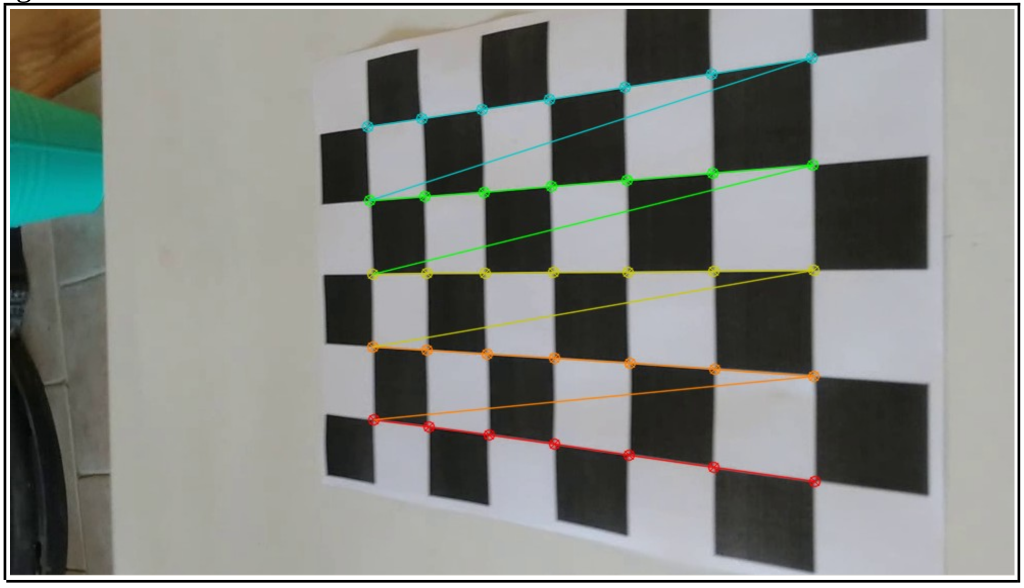
Name : Pavithra Ramasubramaniana Ramachandran
Student ID : R00183771

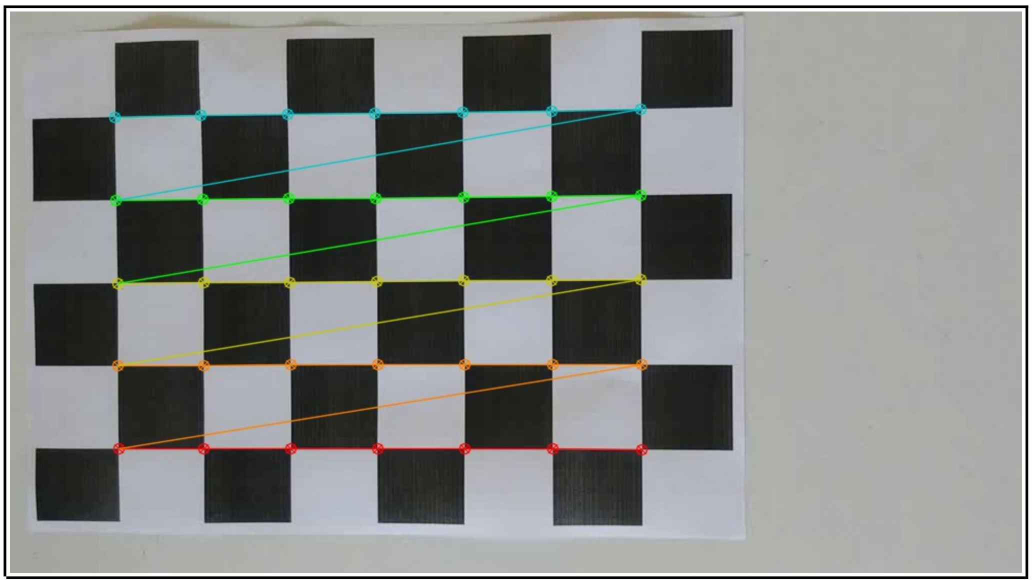
Task 1 (Pre-Processing)

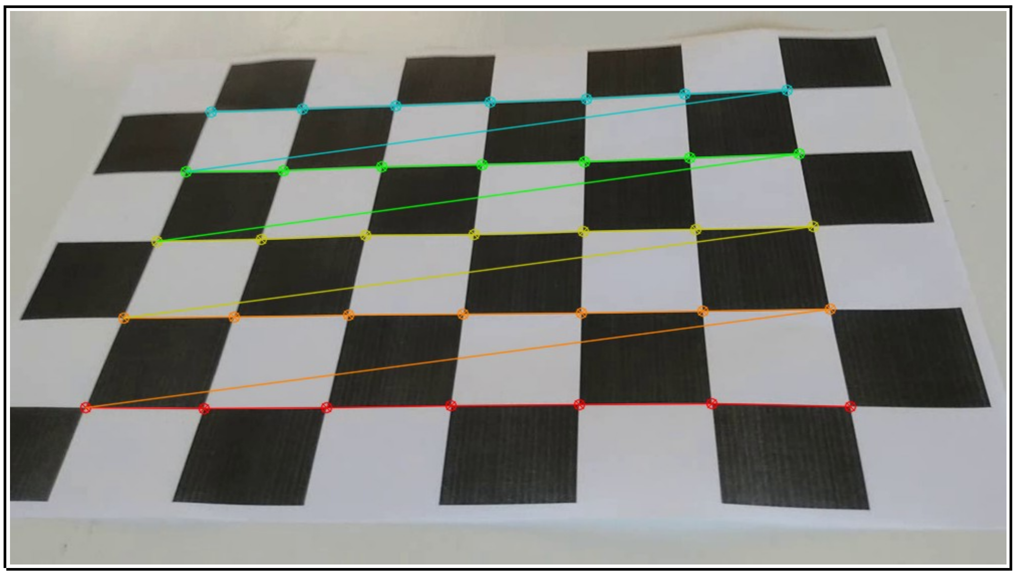
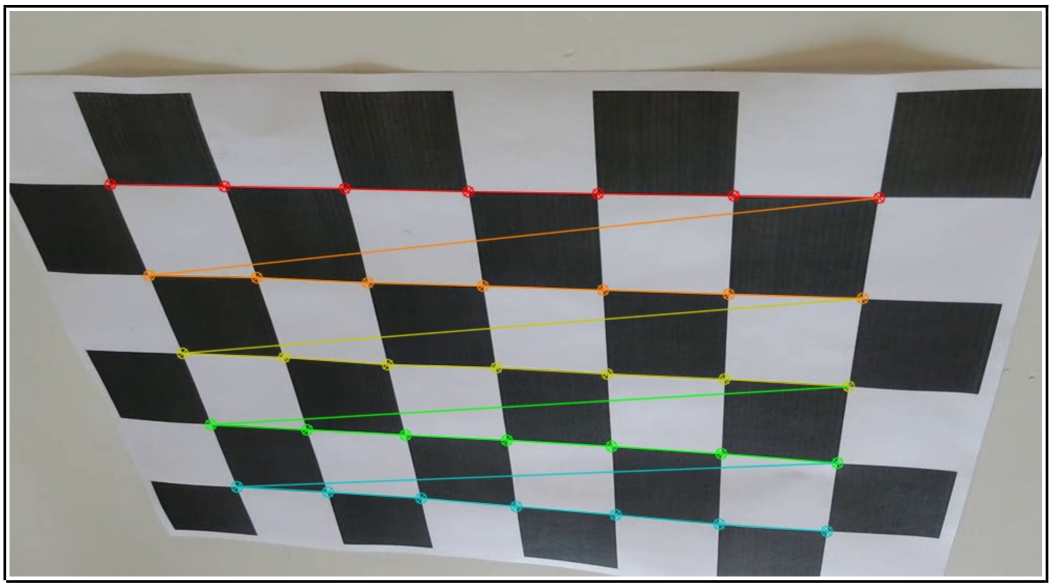
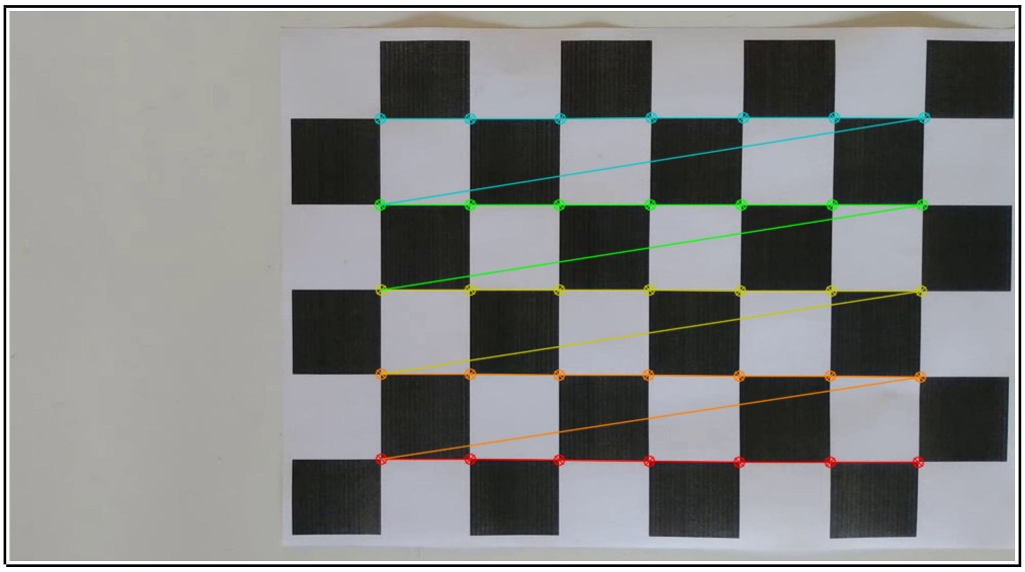
A. Download the file Assignment_MV_02_calibration.zip from Canvas and load all calibration images contained in this archive. Extract and display the checkerboard corners to subpixel accuracy in all images using the OpenCV calibration tools.

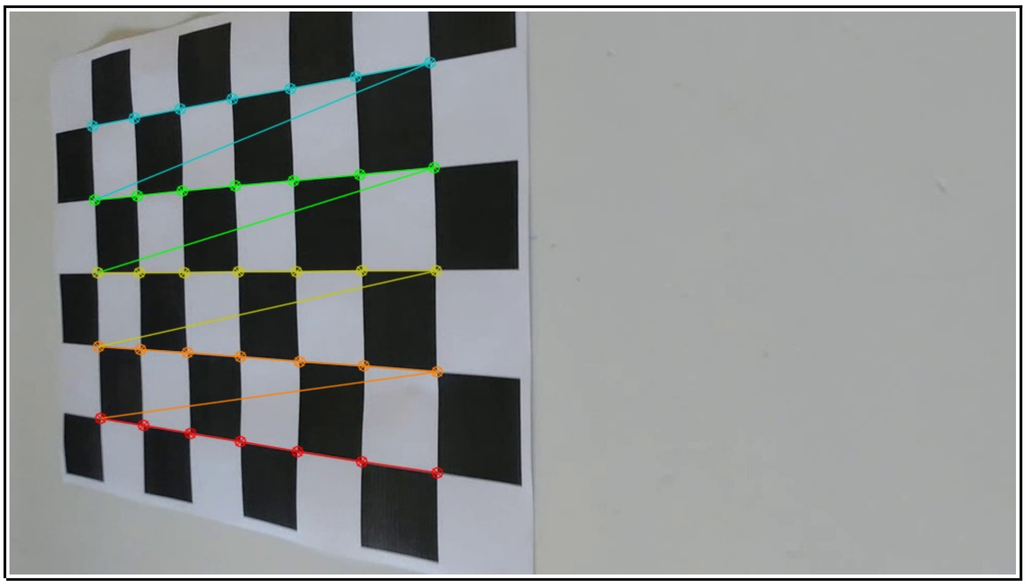
User defined function : camera_caliberation()
Used cv2.findChessboardCorners() to find chessboard corners, cv2.cornerSubPix() to calculate subpixel accuracy and cv2.drawChessboardCorners() to draw chessboard corners to get the output images.

Output Images:







B. Determine and output the camera calibration matrix K using the OpenCV calibration tools.
User defined function : camera_caliberation()
Used cv2.calibrateCamera() to get K.

```
K =
 [[994.9374175    0.         485.13219066]
 [  0.         953.82995226 286.16774858]
 [  0.           0.           1.        ]]
```

C. Download the file Assignment_MV_02_video.mp4 from Canvas and open it for processing. Identify good features to track in the first frame using the OpenCV feature extraction and tracking functions. Refine the feature point coordinates to sub-pixel accuracy.

User defined funtion : get_tracks()
Used cv2.goodFeaturesToTrack() to track good features and cv2.cornerSubPix() to compute subpixel accuracy of first frame.

D. Use the OpenCV implementation of the KLT algorithm to track these features across the whole image sequence. Make sure to refine the feature pointcoordinates to sub-pixel accuracy in each step.

User define function : get_tracks()
Used cv2.goodFeaturesToTrack() to track good features if the features get lost in subsequent frames, cv2.cornerSubPix() to compute subpixel accuracy and cv2.calcOpticalFlowPyrLK() to calculate optical flow between two frames.

<u>Task 2 : Fundamental Matrix</u>

A. Extract and visualise the feature tracks calculated in task 1 which are visible in both the first and the last frame to establish correspondences x i ↔ x ′ i between the two images [2 points]. Use Euclidean normalised homogeneous vectors.

User defined function : calculate_homography()
Calculated correspondences x1 and x2 from the first and last frames.

B. Calculate the mean feature coordinates μ = N ∑ i x i and μ ′ = N ∑ i x i ′ in the first and the last frame. Also calculate the corresponding standard deviations. Normalise all feature coordinates and work with y i = Tx i and y ′ i = T ′ x ′ i which are translated and scaled using the homographies T1 nad T2.

The mean and standard deviations of the correspondences are calculatedin order to compute T1 and T2. After this, y1 nad y2 are calculated using the formula given in the PDF.

C. Select eight feature correspondences at random and build a matrix comprising the eight corresponding rows a T i = y T i ⊗ y i ′ to calculate the fundamental matrix using the 8-point DLT algorithm.

Eight feature correspondences are selected at random using np.ramdom.random_interger() function. The kronecker product is the computed to obtain the matrix A.

D. Use the 8-point DLT algorithm to calculate the fundamental matrix F for the eight selected normalised correspondences y i ↔ y ′ i. Make sure that F is the singular. Apply the normalisation homographies to F to obtain the fundatmental matrix F.

The obtained A matrix is singularized using np.linalg.svd() and then np.matmuloperations are performed to obtain the fundamental matrix F.

E. For the remaining feature correspondences x i ↔ x ′ i not used in the 8-point algorithm calculate the value of the model equation.Also calculate the variance of the model equation using the Cxx matrix.

For the remaining correspondences, the model equation and variance is computed using np.matmul functions.

F. Determine for each of these correspondences if they are an outlier with respect to the selection of the eight points or not by calculating the test statistic.Use an outlier threshold of T i > 6.635 . Sum up the test statistics over all inliers.

Used 6.635 as the threshold to identify the outliers, and inliers.

G. Repeat the above procedure 10000 times for different random selections of correspondences. Select the fundamental matrix and remove all outliers for the selection of eight points which yielded the least number of outliers. Break ties by looking at the sum of the test statistic over the inliers.

Identified the iteration with the least number of outliers, and stored the count of outliers, the sum of inliers, the best fundamental matrix and the inlier correspondences.

H. Adapt the display of feature tracks implemented in subtask A to indicate which of these tracks are inliers and which tracks are outliers. Also calculate and output the coordinates of the two epipoles?

Co-ordinates of two epipoles :
e1 : [-0.61975005 -0.78479695 -0.00190501]
e2 : [-0.6114062 -0.79131463 -0.00189971]

Task 3 : Essential Matrix and 3d points

A. Use the fundamental matrix F determined in task 2 and the calibration matrix K determined in task 1 to calculate the essential matrix E . Make sure that the non-zero singular values of E are identical. Also make sure that the rotation matrices of the singular value decomposition have positive determinants.

```
Essential matrix :
[[-26.59518751  16.96895151  -2.52063906]
 [-17.7699044  -25.6752839    5.89295856]
 [ -1.9358972    6.29447168  -1.21918795]]

Rotation matrix determinant :  0.9999999999999994
```

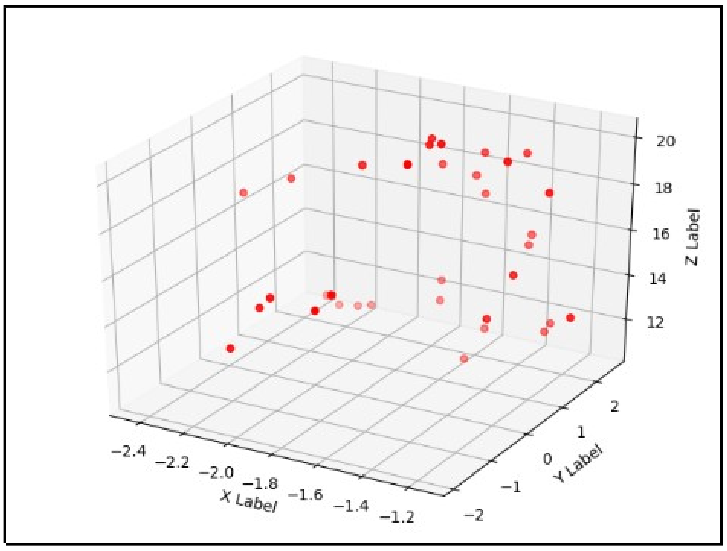
B. Determine the four potential combinations of rotation matrices R and translation vector t between the first and the last frame. Assume the camera was moving at 50km/h and that the video was taken at 30fps to determine the scale of the baseline t in meters.

```
Rotation Matrix 1 :
 [[ 0.53954126  0.84150423  0.02767415]
 [-0.82553121  0.52226689  0.21385864]
 [ 0.16550966 -0.13823144  0.97647254]]
Translation Vector:  [  0.48735252   2.78377194 -13.59832107]

Rotation Matrix 1 :
 [[ 0.53954126  0.84150423  0.02767415]
 [-0.82553121  0.52226689  0.21385864]
 [ 0.16550966 -0.13823144  0.97647254]]
Translation Vector:  [-0.48735252 -2.78377194 13.59832107]

Rotation Matrix 2 :
 [[-0.5282775  -0.77716529 -0.34196052]
 [ 0.8161502  -0.57585152  0.0478945 ]
 [-0.23414043 -0.25378956  0.938493  ]]
Translation Vector:  [  0.48735252   2.78377194 -13.59832107]

Rotation Matrix 2 :
 [[-0.5282775  -0.77716529 -0.34196052]
 [ 0.8161502  -0.57585152  0.0478945 ]
 [-0.23414043 -0.25378956  0.938493  ]]
Translation Vector:  [-0.48735252 -2.78377194 13.59832107]
```

C. Calculate for each inlier feature correspondence determined in task 2 and each potential solution calculated in the previous subtask the directions m and m′ of the 3d lines originating from the centre of projection towards the 3d points. Then calculate the unknown distances λ and μ by solving the linear equation system to obtain the 3d coordinates of the scene points. Determine which of the four solutions calculated in the previous subtask is correct by selecting the one where most of the scene points are in front of both frames, i.e. where both distances λ > 0 and μ > 0. Discard all points, which are behind either of the frames for this solution as outliers .

The lambda and mu values of m1 and m2 were calculated and the solution from R1T1, R1T2, R2T1 and R2T2 with highest number of scene points based on lambda >0 and mu>0 was chosen to the best inliers.

D. Create a 3d plot to show the two camera centres and all 3d points.



E. Project the 3d points into the first and the last frame and display their position in relation to the corresponding features to visualise the reprojection error.

Movement of inliers from for m2 is depicted using blue lines. For m1 there are no lines as these lines are depicted on the first frame.