CMSC 426 Final Project Report

Ronak Agarwal

Part 1

3.1
Used the ReadCameraModel function to import the model, using fx, fy, cx, and cy to create the intrinsic matrix as a numpy array.

3.2
Loaded the images in from the Oxford dataset, and sorted them in order. Created a loop that iterates over each of the 376 images reading in the current and the next image on each iteration. Used cv2.imread to get the bayer encoded images, and converted them to color images using cv2.COLOR_BayerGR2BGR. Then used UndistordImage and the LUT from RealCameraModel to undistort the images.

3.3
Used cv2.SIFT_create() SIFT keypoint algorithm to get keypoints and descriptions for each image. Used cv2.BFMatcher brute force matcher to find matches between the successive frames. Then only kept the best 250 matches sorted by distance.

3.4
Used the keypoints for both images and cv2.findFundamentalMat to find the fundamental matrices for each image. Did this by reshaping the keypoints to work with the function, and using cv2.FM_RANSAC.

3.5
Calculated the essential matrix by multiplying the intrinsic matrix with the fundamental matrix.

3.6
Used recover pose with the essential matrix, keypoints, and intrinsic matrix to get the rotation and matrices translation vectors. Then appended these to a list for further use.

Part 2

Created a matrix to cumulatively store the movement between all of the images. Initializes three empty lists,to store the camera positions along the X, Y, and Z axes, respectively.

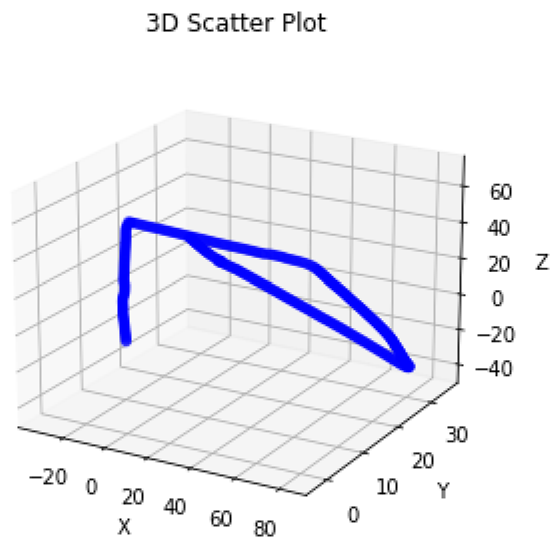
Looped over the sequence of images and computed the transformation matrix T between camera i and camera i+1. This transformation matrix combines the rotation matrix and the translation vector using np.hstack() and np.vstack().

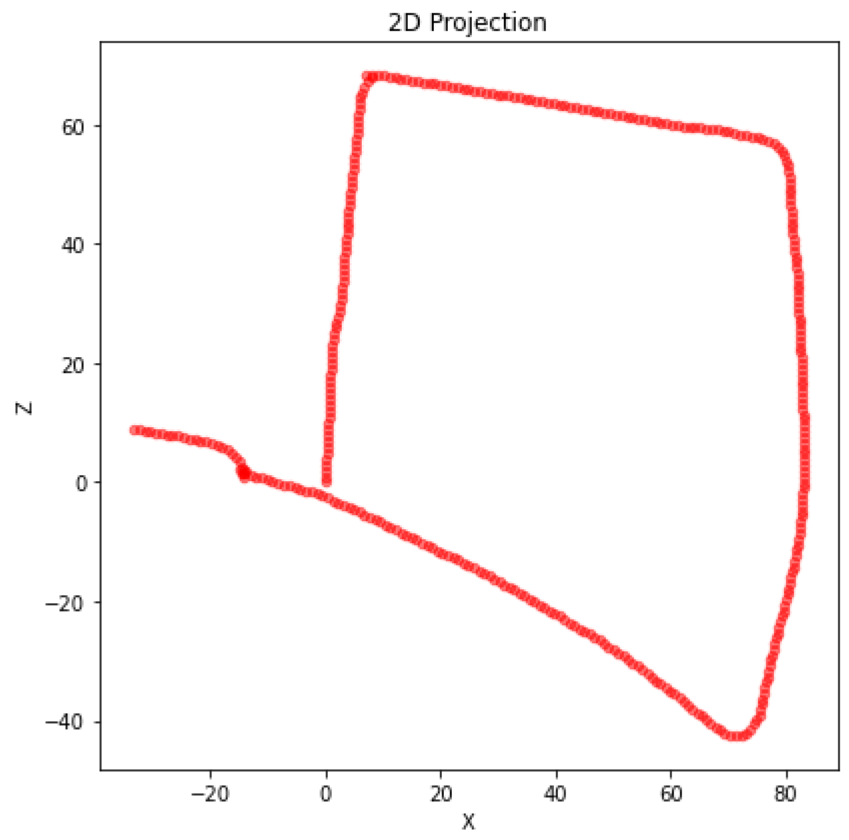
Multiplied the transformation matrix and the cumulative matrix, storing it back in the cumulative matrix and updating the camera position for each iteration.

Computed the inverse of the transformation matrix and mulitplied the inverse transformation matrix with the origin array, [0, 0, 0, 1]. The resulting transformed vector represents the camera center position relative to the current image.

Appended the X, Y, and Z coordinates of the camera center position to the respective lists. And graphed.

2D and 3D Projection Results





Extra Credit

I have implemented this code as functions external to the rest of the code because I find it to significantly slow down performance. However I have left the reference to this function in the main code as a comment.

- The random_matching_points function selects random points from two sets of matching points.
- The normalized_f_matrix function calculates the fundamental matrix using normalized eight-point algorithm.
- The compute_fundamental_matrix function iteratively computes the best fundamental matrix by randomly selecting points, normalizing them, and calculating the inliers using a threshold.

Finally, the function returns the best fundamental matrix.