CMSC733 Project 1 Report

One-day Extension Granted

Pyone Win
Department of Computer Science
University of Maryland, College Park
College Park, Maryland
Email: pwin17@umd.edu

Nitesh Jha
MEng Robotics
University of Maryland, College Park
College Park, Maryland
Email: niteshj@umd.edu

I. Phase 1

In this Phase, the detailed explanation and the results of the traditional approach to stitch multiple images for a panoramic effect is presented. The approach consists of six steps in total: corner detection, finding best corners, extracting feature descriptors, matching features, outlier rejection for matching features and calculating the homography between two images, and image warping and stitching. In this section, we refer to the image we want to stitch to as the first image and the one we want to warp to match the first image as the second image.

A. Corner Detection

Before corner detection, we convert our images to grey scale. Then, Harris Corner Detection algorithm is used to obtain corners of the images. Below are the corner detection results from input images as well as stitched images for stitching with another image. We are only including the first two sets as the results are similar and this part is quite straightforward.

Fig. 1. Corner Detection result of image 1 from Set 1

B. Adaptive Non-Maximal Suppression

Harris Corner Detection gives us a huge number of corners so we apply adaptive non-maximal suppression algorithm to filter those corners to get N-best corners. In our case, we obtain 300 best corners per image. Although ANMS works well for the input images, when it is applied on the stitched images, it will consider the border between stitched image and the black background which is a result of warping as strong corners. We handled that by checking whether a half of a patch from the corner point is all black. We then remove those corners from our best corners. Below are the ANMS results.We are only including the first two sets as the results are similar and this part is quite straightforward.

Fig. 2. Corner Detection result of image 2 from Set 1

Fig. 3. Corner Detection result of image 2 from Set 1

C. Feature Descriptors

In this step, we take the best corner points, take 41 by 41 patches centering at the corner. The patch is then applied

Fig. 4. Corner Detection result of stitched image from Set 1

Fig. 5. Corner Detection result of image 1 from Set 2

Gaussian Blurring. In this step, we used (7,7) kernel size for the most samples, but we also used (5,5) and (9,9) depending on the results we have seen to optimized the results. Once we obtained the blurred feature, we reshaped it into 8 by 8 patch and finally flatten it to get the feature vector to be used in the next step. I have included random feature descriptors from set 1 here.

Fig. 6. Corner Detection result of image 2 from Set 2

Fig. 7. Corner Detection result of image 2 from Set 2

D. Features Matching

Using the features obtained from the last step, we try to match the features between two images. We compared each feature in the first image with all features in the second image. The comparison is done by calculating the sum of squared differences between two features. If the ratio of the best two features are small enough to be under the threshold, we will consider this as a matching features pair. Below are the feature matching results of the ones that worked.

E. RANSAC for outlier rejection and Homography Calculation

The feature pairs we obtained from the last step has possible outliers. To overcome this, we perform RANSAC algorithm. For a number of user-input iterations, we randomly select 4 pairs of points and calculate the estimated homography. Then, it is applied to the main image points and the sum of squared difference (SSD) is calculated between it and the points of the second image. For all SSD less than the input threshold (50.0 for us), the points are considered inliers. Using our inliers, we calculated the homography matrix which will be used in the next step.

Fig. 8. ANMS result of image 1 from Set 1

Fig. 9. ANMS result of image 2 from Set 1

F. Image Warping and Stitching

Using the inverse of the homography, we applied it to the the second image. That image is stitched to the first image by creating a blank template and putting the images in their respective locations. The respective locations are calculated by checking the distance between the matching points from RANSAC. If the distances are less along the x-axis, the images are stitched horizontally. If the distance is negative, the second image is on top of the first image, vice versa. The same logic applies to the horizontal stitching. We used floor(N/2) as our anchor image where N is the total number of images. The rest of the images were stitched based on this. The reason being is that this reduces the problem of extreme distortions as the number of images to be stitched increased. We did not use any blending as we judged that it can make the stitched image more blurry based on some trials we tested. Here, we made an assumption that input images will be in-order for us to stitch. So, our program fails when the two successive images are matching enough to stitch. One big challenge of using the traditional approach is that sometimes, homography calculation is not good enough. The warped image looks really bad as a result and it effects the images that follows to be failed as well. We tried to crop out the extra black regions in the final image but was not successful due to the time constraint.

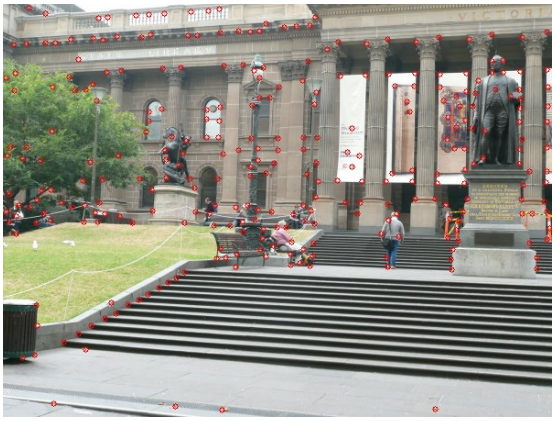


Fig. 10. ANMS result of image 2 from Set 1

Fig. 11. ANMS result of stitched image from Set 1

G. Testing Image Sets

Fig. 12. Random set of 41 by 41 feature descriptors from Set 1

Fig. 13. Random set of 8 by 8 feature descriptors from Set 1

II. Phase 2

In this section, we use deep learning approach to estimate homography between a pair of image patches that will be used to stitch the images. We will implement a supervised and an unsupervised network for this task.

A. Data Generation

The input to both networks are a pair of image patches of a fixed size that are related by some homography between them. We use a method of generating infinitely many samples from a fixed number of training images. For this, we first take a training image and resize it to (324x240) dimensions for uniformity across all training images (all of different sizes). Then, we convert it to grayscale, and randomly select a 128x128 patch on this image. This patch is called 'patch A' henceforth. Then, we apply some perturbation ($\rho \in$ [-32,32]) to each corner of patch A, and obtain 'patch B'. Using corners of both patches, we find the homography matrix between them. To generalize the model, patch generation is performed while training , instead of saving a fixed number of patches. This way, the network gets a different pair of patches every time and inputs are never repeated in the entire training. An example of patch generation is shown in fig 52

Fig. 14. Random set of 41 by 41 feature descriptors from Set 3

Fig. 15. Random set of 8 by 8 feature descriptors from Set 3

B. Supervised network

1) Network architecture: The network consists of 8 convolution layers with maxpooling after every two layers, and each convolution is batch-normalized and activated with ReLU function. The convolution layers are followed by two fully connected layers of 1024 and 8 units respectively. A dropout layer is used in between them which drops 50% of the activations and helps to generalize the model. The architecture is shown in fig 53

2) Results: The network is trained with inputs as the image patches and the output as H_{4pt} which is given by H_{4pt} = C_a - C_b. The network is trained using SGD optimizer for 200 epochs, with a learning rate of 0.0001 and momentum 0.9. The resulting curves are shown in fig 54. The mean absolute error with the predicted corners of patch B and the ground truth is plotted against epochs in fig 55. To test this model, we visualize the model predictions using the test images provided, and the result shows that the model is reasonably accurate in predicting the homography between patches. This is shown in fig 56,57,58,59. From the table I, we see that the model has

Fig. 16. Random set of 41 by 41 feature descriptors from Custom Set 1

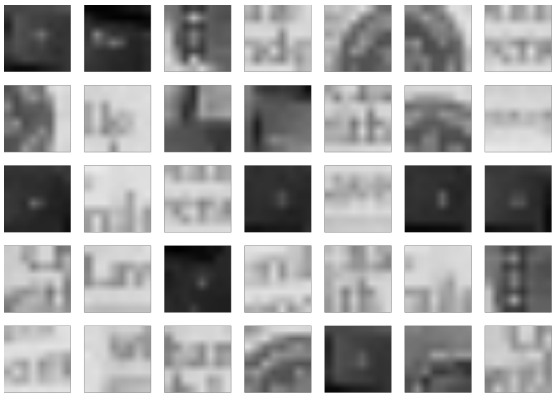


Fig. 17. Random set of 8 by 8 feature descriptors from Custom Set 1

generalized well for predicting homography.

TABLE I
MEAN CORNER ERROR AND L2 LOSS

Metrix	Train	Val	Test
MCE	7.26	7.21	7.32
L2	12.89	12.65	13.07

C. Unsupervised network

1) Network architecture: The network is the same as the supervised network, with the addition of a Tensor Direct Linear Transform to the output of regression model, followed by a Spatial Transform Network, as shown in fig 60.
TensorDLT: The H_{4pt} from the regression network is then converted to the homography matrix such that the transformation is differentiable, so that backpropogation of errors is possible.
Spatial Transform Network: This 'layer' warps the input images using the homography matrix generated by TensorDLT, keeping this transformation differentiable for backpropogation.

The network uses photometric loss for training. We use Adam optimizer with a learning rate of 1e-6, batch size of 64,

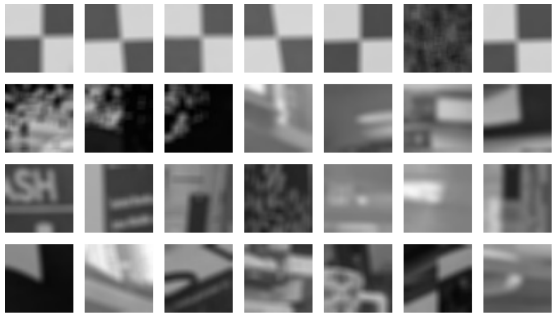


Fig. 18. Random set of 41 by 41 feature descriptors from all test sets

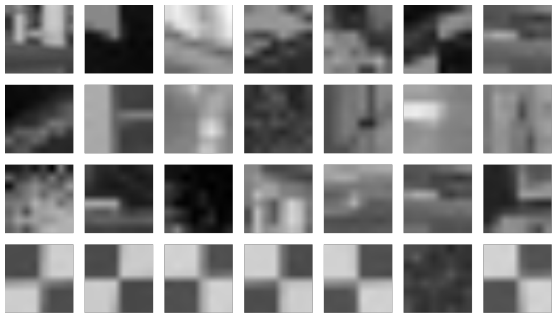


Fig. 19. Random set of 8 by 8 feature descriptors from all test sets

and the total number of samples as 5000. The input images are scaled to put them in range [0,1].

2) Results: A plot of the loss is shown in fig 61. As we see, the network does not improve over 100 epochs (5000 images, 64 per batch) with the current parameters used. Due to the computation time of training further, owing to the time constraints, we could not retrain the model with different parameters, and thus do not have further results.

Fig. 20. Matching Features from the first two inputs from Set 1

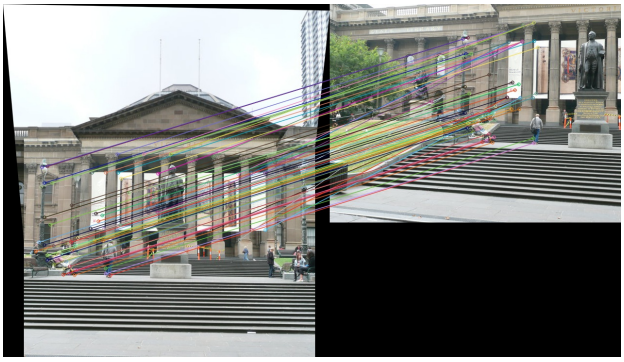


Fig. 21. Matching Features from the stitched image and next inputs from Set 1

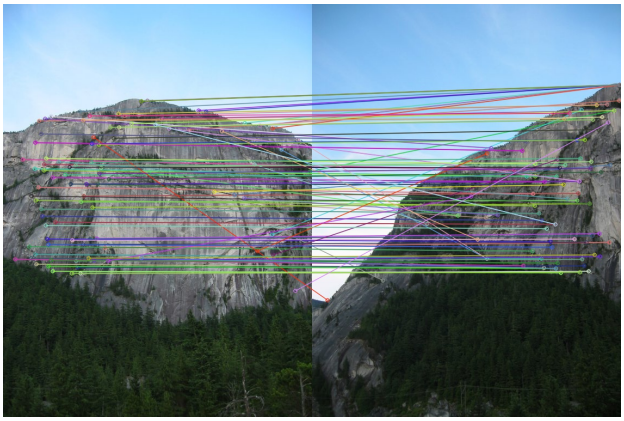


Fig. 22. Matching Features from the first two inputs from Set 2

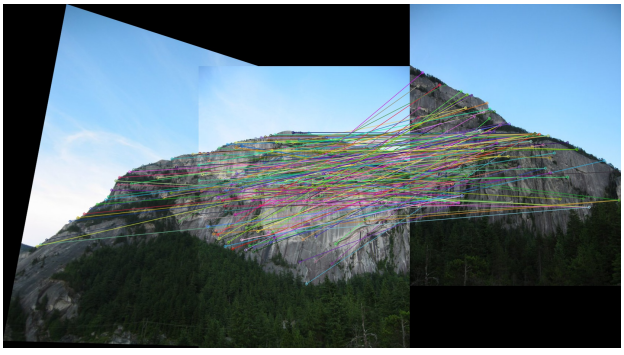


Fig. 23. Matching Features from the stitched image and next inputs from Set 2

Fig. 24. Matching Features from the first two inputs from Test Set 3

Fig. 25. Matching Features from the stitched image and next inputs from Set 3

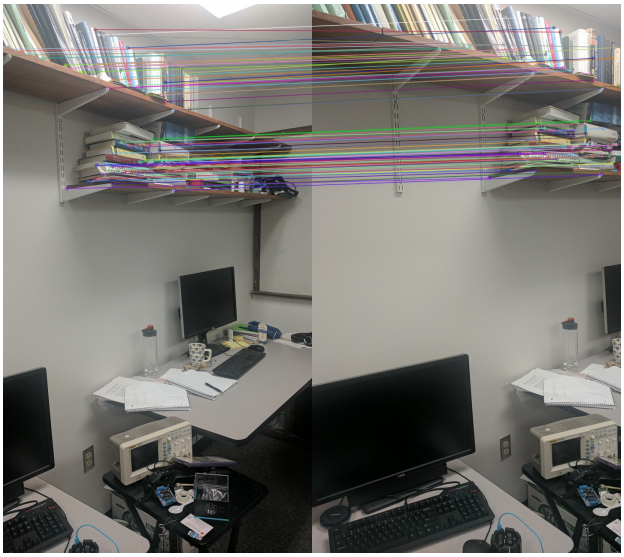


Fig. 26. Matching Features from the first two inputs from Test Set 4

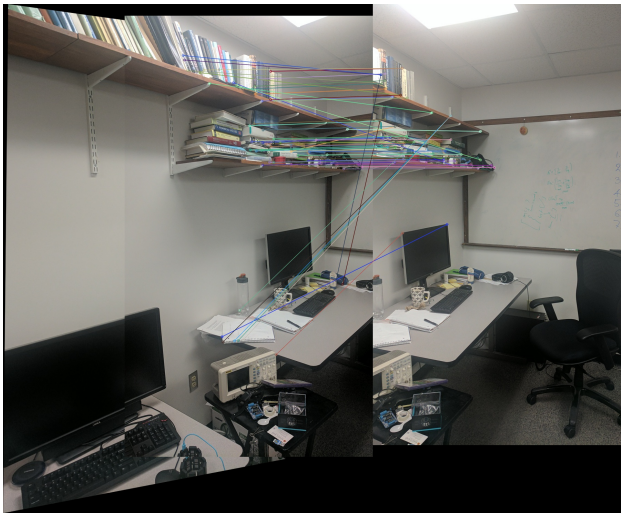


Fig. 27. Matching Features from the stitched image and next inputs from Set 4

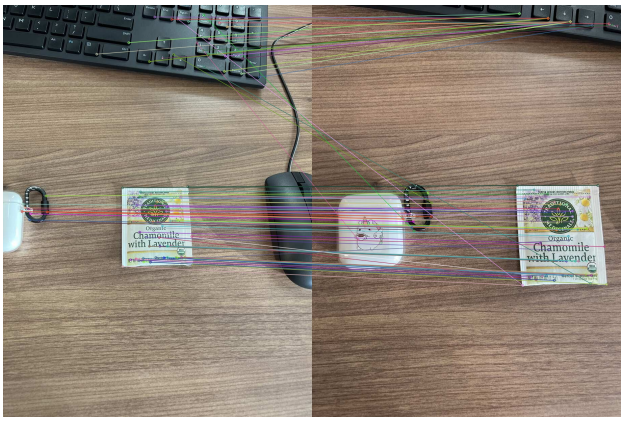


Fig. 28. Matching Features from the first two inputs from Custom Set 1

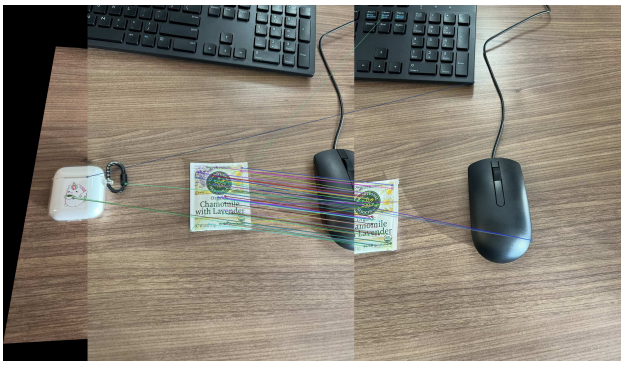


Fig. 29. Matching Features from the stitched image and next inputs from Custom Set 1

Fig. 30. Matching Features from the first two inputs from Custom Set 2

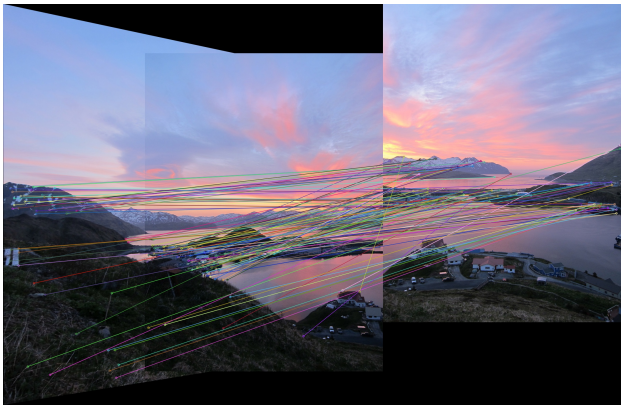


Fig. 31. Matching Features from the stitched image and next inputs from Custom Set 2

Fig. 32. RANSAC result of the first two inputs from Set 1

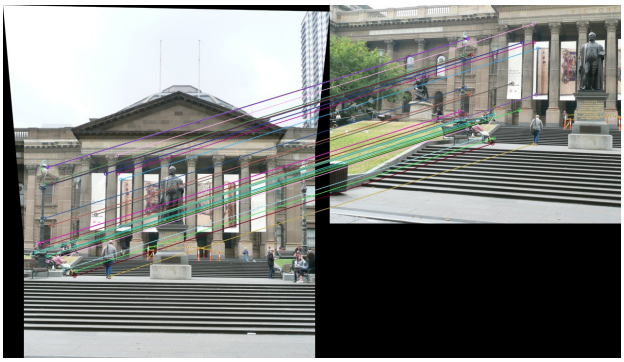


Fig. 33. RANSAC result of the stitched image and next inputs from Set 1

Fig. 34. RANSAC result of the first two inputs from Custom Set 1

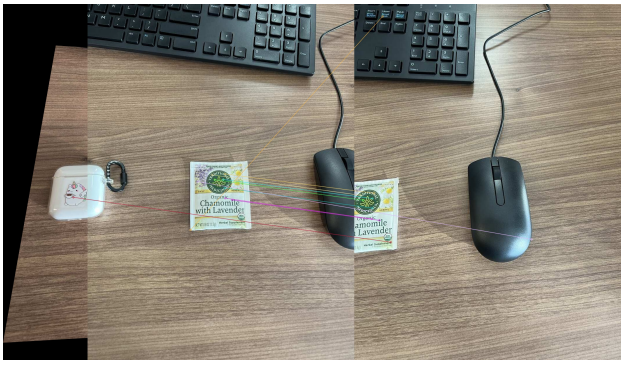


Fig. 35. RANSAC result of the stitched image and next inputs from Custom Set 1

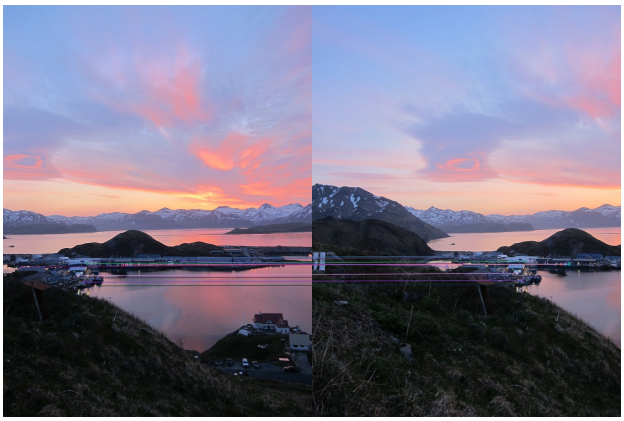


Fig. 36. RANSAC result of the first two inputs from Custom Set 2

Fig. 37. RANSAC result of the stitched image and next inputs from Custom Set 1

Fig. 38. RANSAC result of the first two inputs from Test Set 3

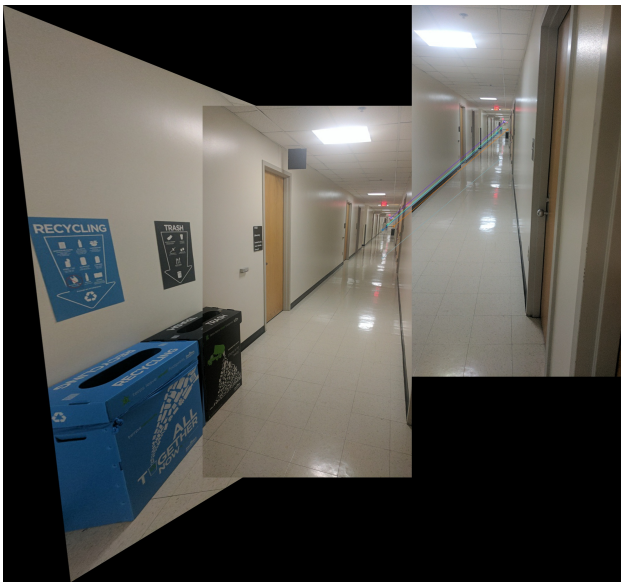


Fig. 39. RANSAC result of the stitched image and next inputs from Test Set 3

Fig. 40. RANSAC result of the first two inputs from Test Set 4

Fig. 41. RANSAC result of the stitched image and next inputs from Test Set 4

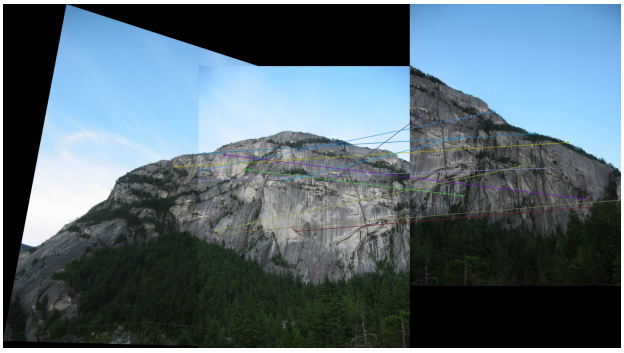


Fig. 42. RANSAC result of the stitched image and next inputs from Set 2

Fig. 43. Final result of Set 1

Fig. 44. Failed result of Set 2

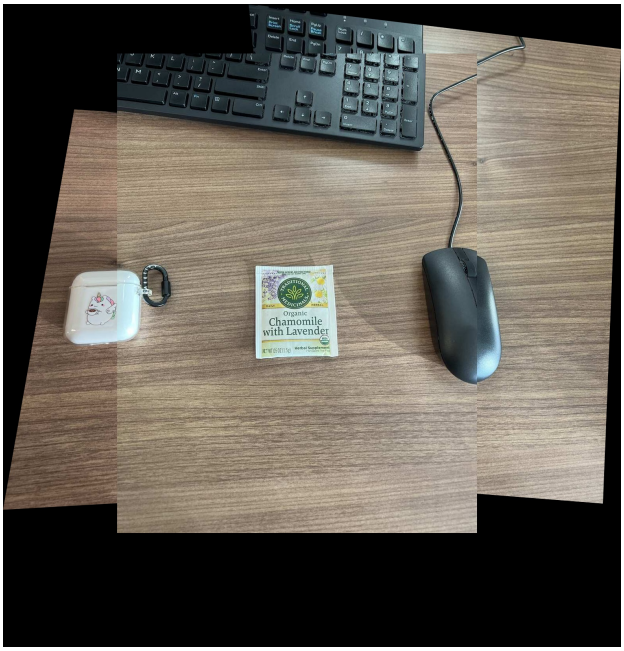


Fig. 45. Final result of Custom Set 1

Fig. 46. Failed result of Custom Set 2

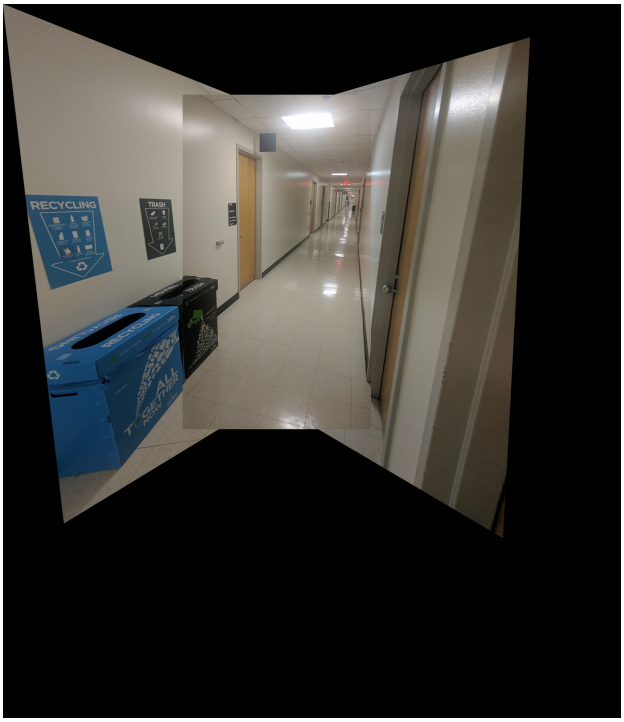
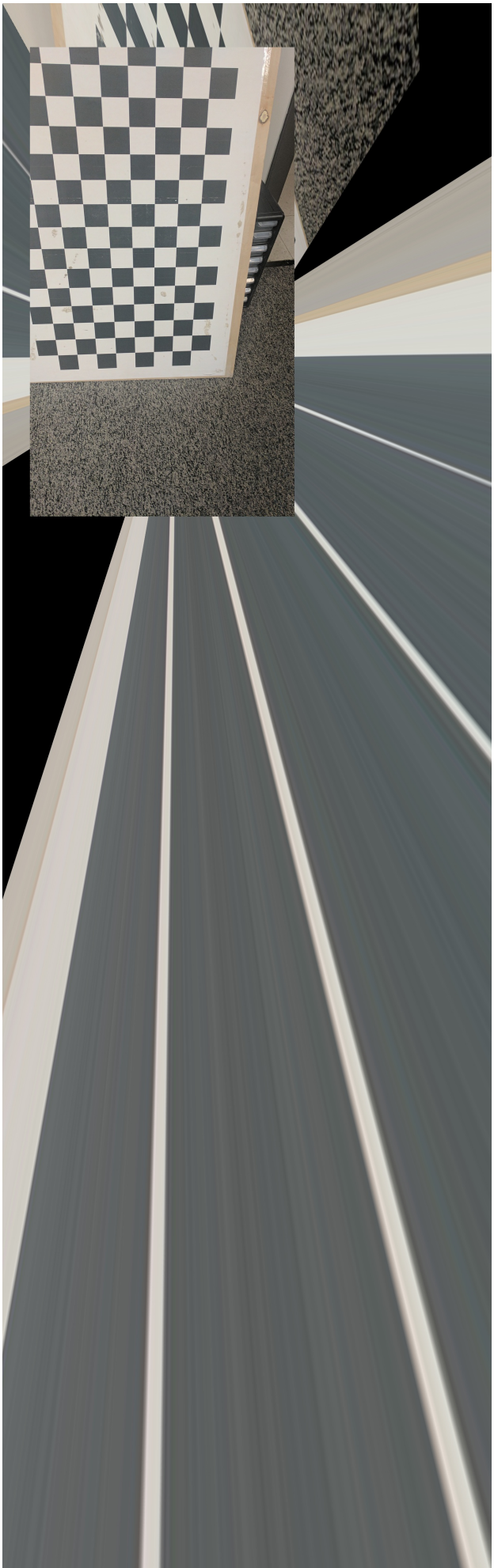


Fig. 47. Final result of Test Set 3

Fig. 48. Final result of Test Set 4

Fig. 49. Failed Result of Set 3



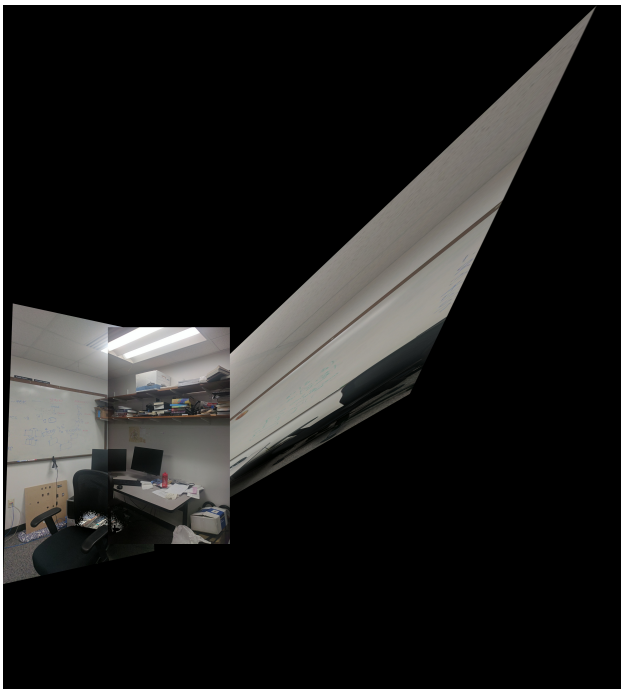


Fig. 51. Failed Result of Test Set 2

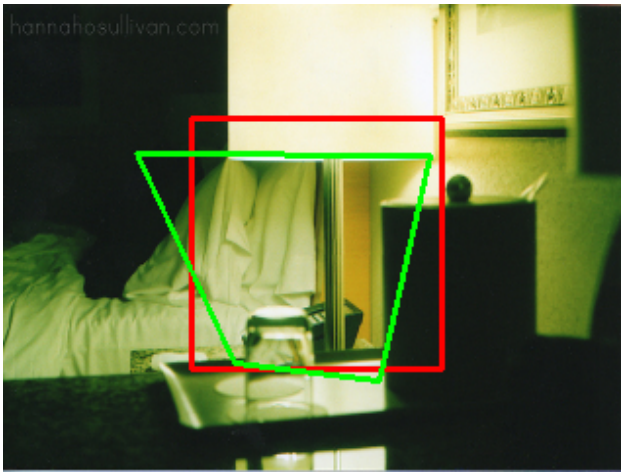


Fig. 52. Patches selected: Patch A (red) and Patch B(green)

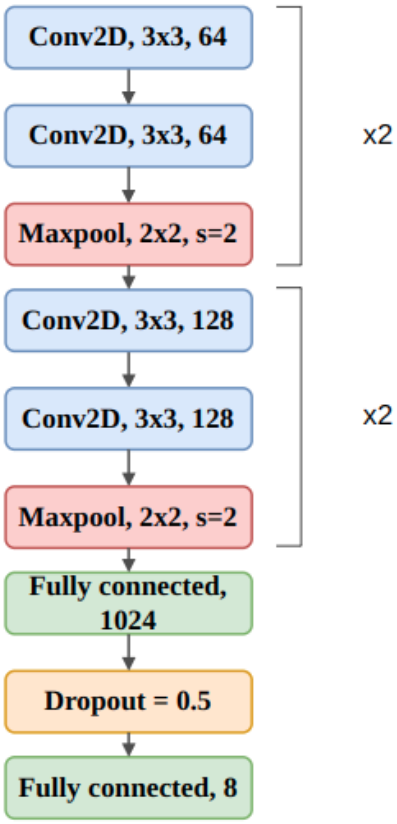


Fig. 53. Supervised network

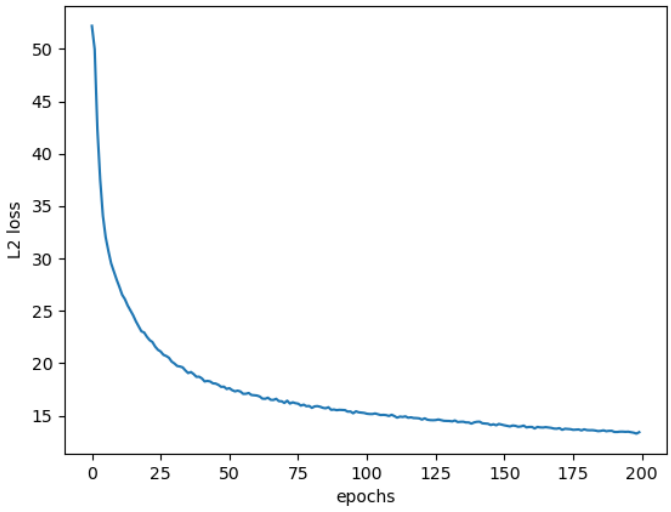


Fig. 54. Supervised training L2 loss

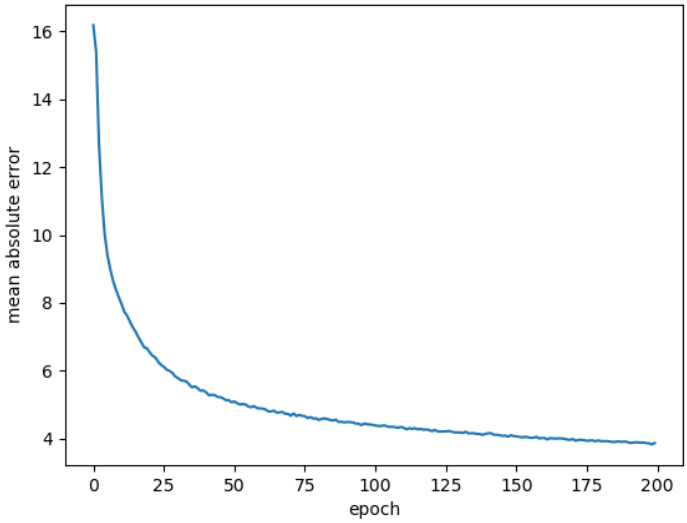


Fig. 55. Supervised training mean absolute error

Fig. 56. Original patch A(red), ground truth patch B(blue), and predicted patch B(green)

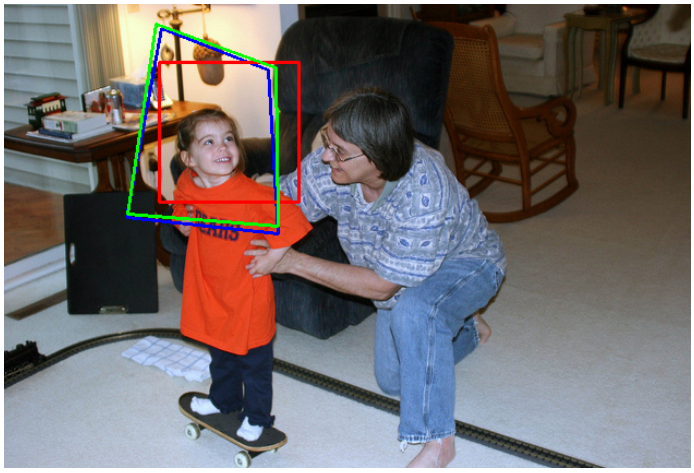


Fig. 57. Original patch A(red), ground truth patch B(blue), and predicted patch B(green)

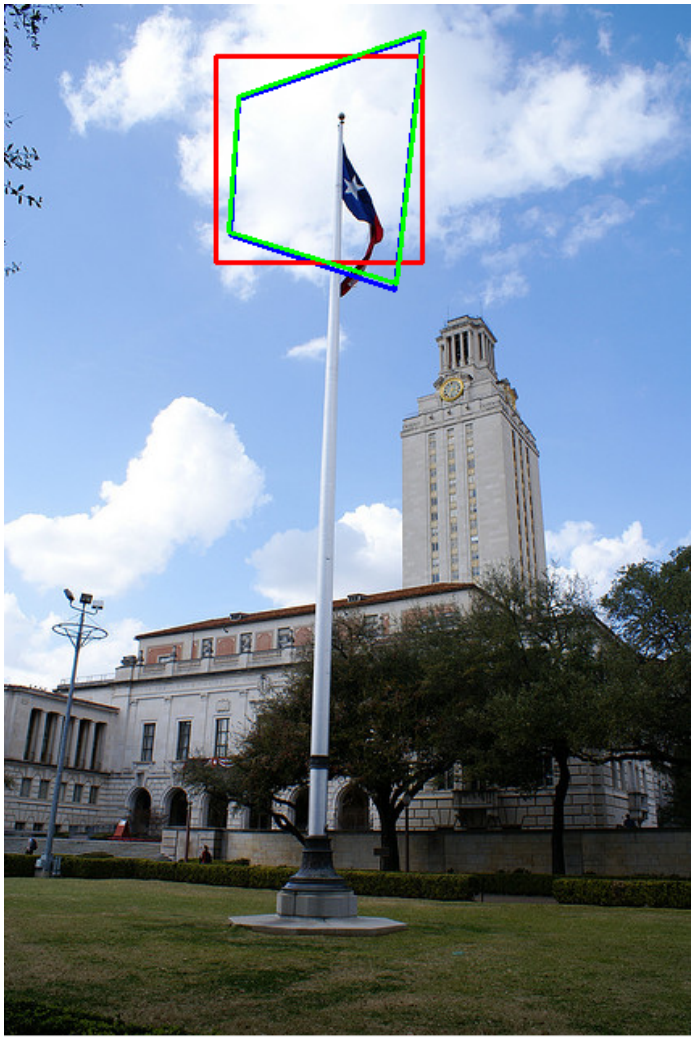


Fig. 58. Original patch A(red), ground truth patch B(blue), and predicted patch B(green)

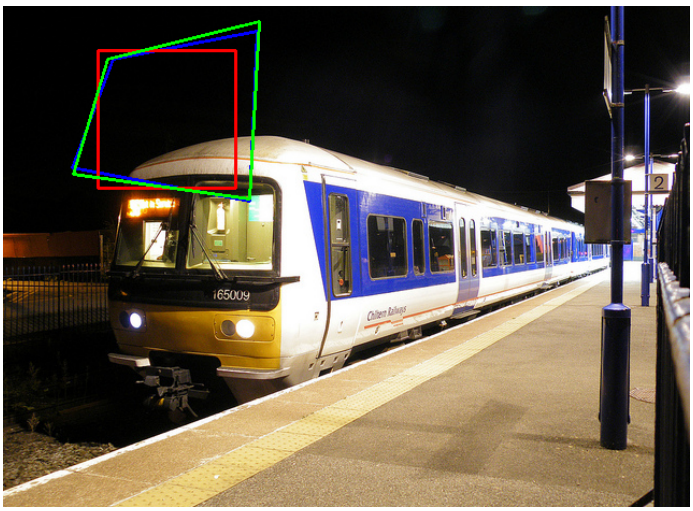


Fig. 59. Original patch A(red), ground truth patch B(blue), and predicted patch B(green)

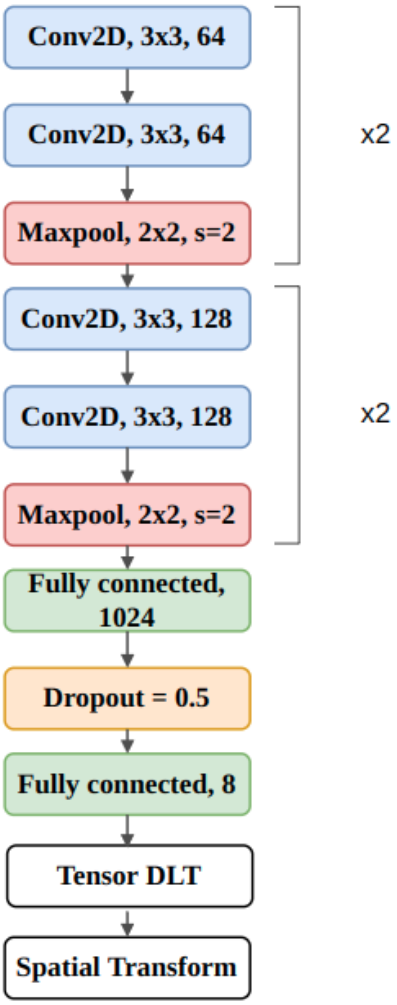


Fig. 60. Unsupervised network

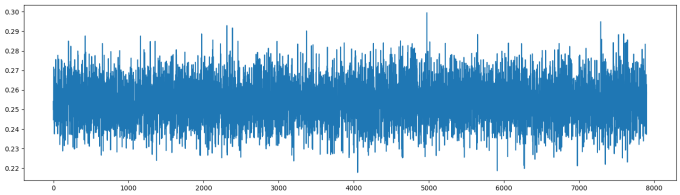


Fig. 61. Unsupervised network: training loss