Project1: AutoPano
RBE549
(Using 2 late days)

Karter Krueger
Department of Robotics Engineering
Worcester Polytechnic Institute
Worcester, MA 01609
Email: kkrueger2@wpi.edu

Tript Sharma
Department of Robotics Engineering
Worcester Polytechnic Institute
Worcester, MA, 01609
Email: tsharma@wpi.edu

Fig. 1. Corner Harris

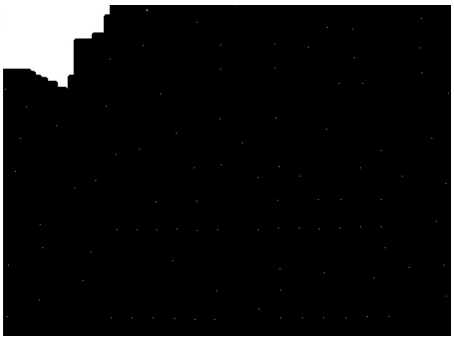

Fig. 2. ANMS Result without convolution pruning

I. Phase 1: Traditional Approach

The first task of the assignment was to create a panorama using traditional computer vision. The approach has six stages:

1) Corner Generation
2) Adaptive Non-Maximal Suppression (ANMS)
3) Feature Generation
4) Feature Matching
5) RANSAC
6) Merging and Blending

A. Corner Detection

We detected corners in the image using the `cv2.cornerHarris()` function which outputs the corner scores on each pixel. These corners are treated as the lowest level features we'll use to match images. The parameters for the function i.e neighbourhood size, aperture size and K are set to 4, 5 and 0.04 respectively. The output is shown in Figure 1.

B. Adaptive Non-Maximal Suppression

The output of the Harris corners in Figure (1) contains many points that are clustered to certain regions, which is problematic when it comes to stitching as we want features that are more evenly distributed across the image. Points close to the optimal corners in the image have a high score as well which leads to the accretion of corers into a small blob. Ideally a corner should be just one pixel and the pixels should be spread across the image uniformly (to prevent abrupt warping in some regions of the image). To achieve this we use ANMS which calculates the local maximas in the image. We used the `scipy.ndimage.maximum_filter()` function and converted it into a mask. However, we were receiving a white blob in top left corner that affected the computaition and feature descriptor calculation giving false positives. We convoluted the image with a kernel of ones of size (3,3) and removed the points with value greater than 1. The outputs of ANMS before and after convolution are presented in Figures 2 and 3 respectively.

C. Feature Descriptors

The corner points returned from Section **??** are used to generate features encoding the local information about the unique points in each image which can be used to match image pairs. A patch of size (40,40) was converted into a descriptor of (64,1) using a sub-sampling step of 5 pixels after performing Gaussian blur with $\sigma = 5$. One of the resultant feature descriptors for Figure 1 is shown in Figure 4

Fig. 3. ANMS Final Result

Fig. 4. Feature Descriptor

D. Feature Matching

The features generated in Section I-C are used to compare two images using a $O(N^2)$ SSD calculation between each feature descriptor pair of any two given images. We kept the best matches where the ratio between the best and the second best pair was less than a set threshold of 0.75.

E. RANSAC and Homography Estimation

Feature matches are often noisy (as seen in Figure 5), so it is necessary to remove bad matches before estimating the homography to align the images. RANSAC [**?**] is a method of removing outliers while estimating the best homography. RANSAC works by repeatedly selecting 4 random pairs of points and computing the homography and number of inliers between the two images which is evident on comparing Figures 5 and 6

F. Matching and Blending

After obtaining the homography matrix using all the inliers from RANSAC, we match images based on the overlap between the image pairs. However, to calculate which images

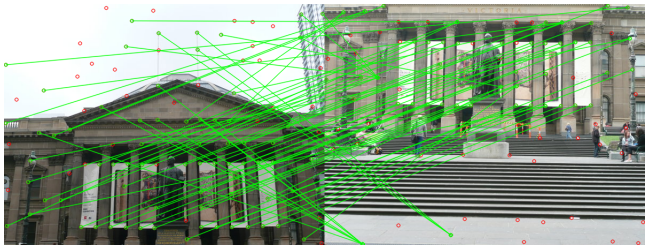

Fig. 5. Feature Matching

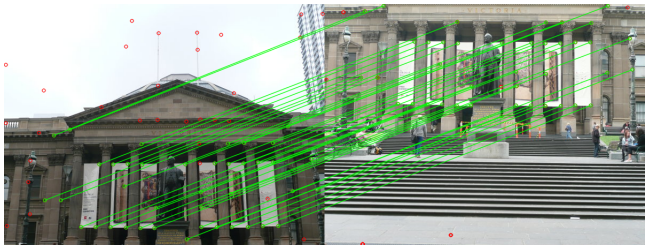

Fig. 6. RANSAC

stitch together well, we created a connectivity matrix where $C_(i,j) = Inliers_(i,j) \forall Inlier_(i,j) / Matches_(i,j) > 0.25$. Here C is the connectivity matrix, i and j correspond to the image pair of Images *i* and *j* in the Image set.

The connectivity matrix is used to identify how each image is linked to other images. We performed a DFS search on each image set to identify the longest chain of images. The resultant graph was used to warp the images around the center image in the chain with their homographies chained together in the direction of the image from the center to obtain the actual homography for an image in the set. The images are blended together by taking the union of each image pair across the three channels.

G. Test Results

The following are our approach's results on Train and Test sets:

Fig. 7. Corner Harris on Train Set 1

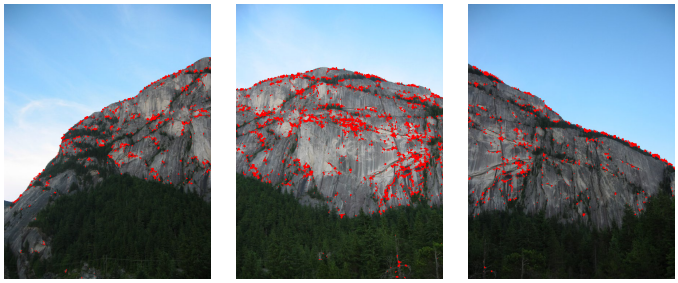

Fig. 8. Corner Harris on Train Set 2

H. Deep Learning-based Stitching Results

An alternative to the classic method is to use a deep learning approach to compute a homography between each pair of images and stitch them together. While the training loss from our tests appeared low on the training and validation data, we found that stitching results did not turn out as good. The results

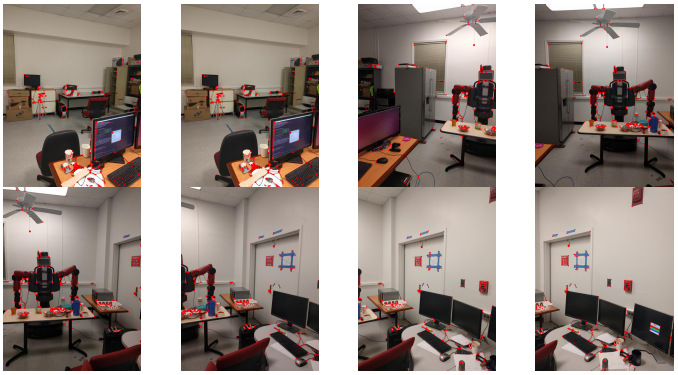

Fig. 9. Corner Harris on Train Set 1

Fig. 10. ANMS on Train Set 1

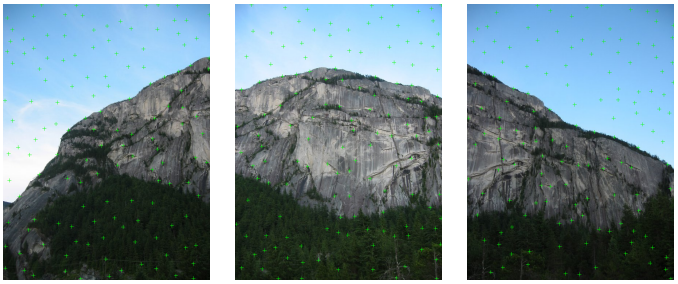

Fig. 11. ANMS on Train Set 2

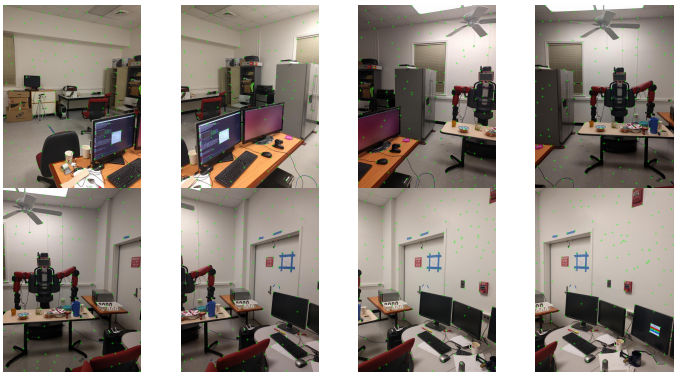

Fig. 12. ANMS on Train Set 3

of stitching using the supervised and unsupervised methods are shown in Figs. 30 and 31, respectively.

II. Phase 2: Deep Learning Homography Estimation

At times, it is challenging for the classic methods to match similar features, such as those seen in the checkerboard dataset. This is one way that deep learning-based methods

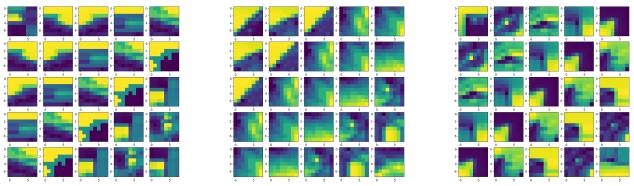

Fig. 13. 25 Random Feature Descriptors for Train Set Images 1,2,3 respectively

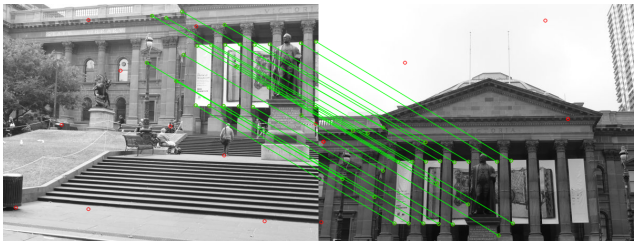

Fig. 14. RANSAC Train Set 1

can out-perform the classic methods, in addition to running much faster with one-shot homography generation rather than the tedious steps of finding points, matching, and filtering. In phase 2 we implement two approaches to deep networks with both supervised and unsupervised approaches.

A. Dataset Generation

First, we must generate a dataset that can be used for training since it would be very challenging to otherwise collect enough accurate ground truth data to train a model. We use a subset of the MS-COCO dataset of 5000 images. For each image, we select a random coordinate in the view and get 4 points that create a 128x128 box to be cropped from the image as a patch. We then apply random perturbations to the 4 corner points and calculate a homography matrix for the simulated perspective change. This homography can then be applied to the original image and the matching patch from the warped

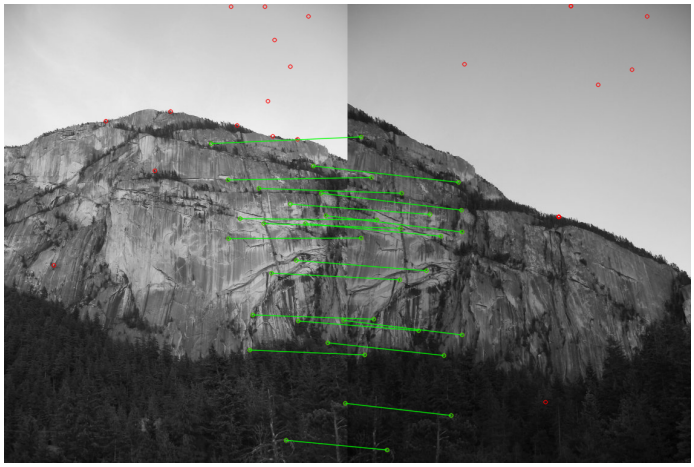

Fig. 15. RANSAC Train Set 2

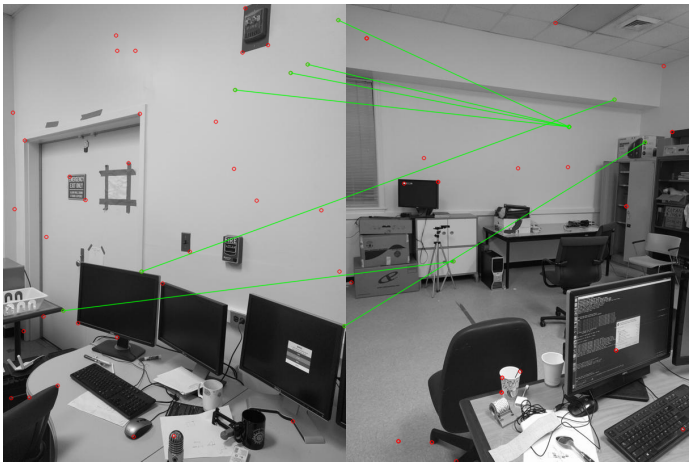

Fig. 16. RANSAC Train Set 3

Fig. 19. Set 3 Images Stitch

Fig. 17. Set 1 Images Stitch

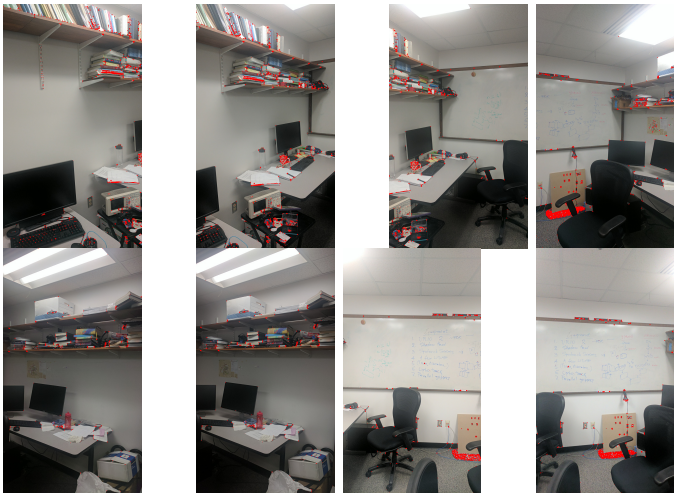

Fig. 20. Corner Harris on Test Set 2

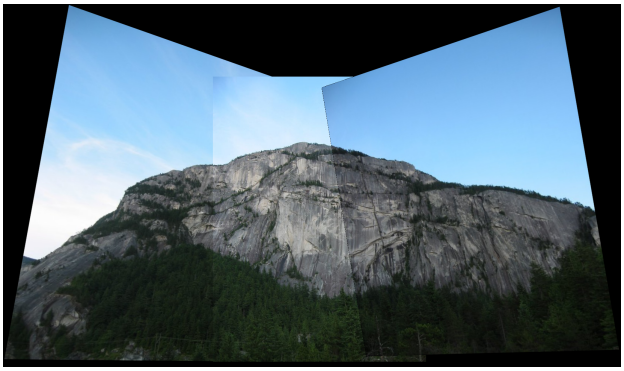

Fig. 18. Set 2 Images Stitch

Fig. 21. Corner Harris on Test Set 3

image is then cropped from the same region. This causes a simulated shift in perspective as if the camera was at a new angle when taking the photo.

We warped photos by adding random deltas to the corners using a normal distribution with mean = 0 and $\sigma = 10$. We

Fig. 22. ANMS on Train Set 2

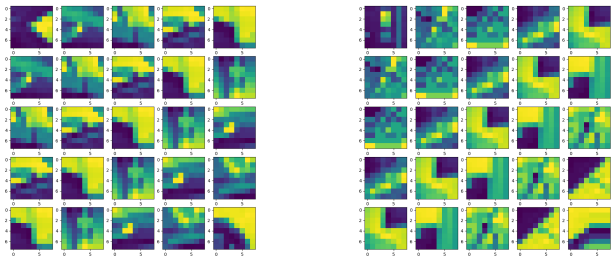

Fig. 23. Feature Descriptors for Test Set Images 1,2,3 respectively

Fig. 24. ANMS on Train Set 3

also trained a second network with a $\sigma = 25$ for more severe warps to see if the network would be more robust for the datasets.

B. Supervised Approach

The first network implemented is a supervised approach. This network takes in a stack of the two RGB images for an input size of 128x128x6 and outputs a vector of the 4 corner point perturbations as an 8x1 vector that can be reshaped to 4x2 for 4 points with x,y shifts. The shifts can then be added

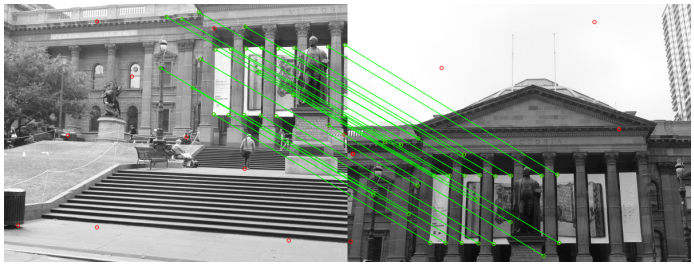

Fig. 25. RANSAC Test Set 1

Fig. 26. RANSAC Test Set 2

to the the original corner values and a homography matrix is calculated. We implemented the original architecture seen in the paper. The network architecture used for HomographyNet used in both supervised and unsupervised training is also shown in Figure

We trained using SGD, with a learning rate of 0.0005, for 6hours to reach 200 epochs. The training and validation loss plots are seen below in Figs. 40, 41. The result of the 1st supervised network on the test set is a loss value of 14.68 MSE average across the 1000 test images. The result on the 2nd supervised test (with $\sigma = 25$) was 37.28 MSE.

C. Unsupervised Approach

The second network uses an unsupervised approach with an identical architecture to the supervised method, but a very different calculation of loss. Loss is calculated through several steps. First, the corner delta offset values come out of the network and are used with the 4 corners (that we used to crop) to find the homography matrix. Next, the homography is applied to the original image A and cropped to get the new version of image B patch. We then calculate the L1 (absolute mean error) loss between the original patch B and the re-produced version based on the network homography. We call

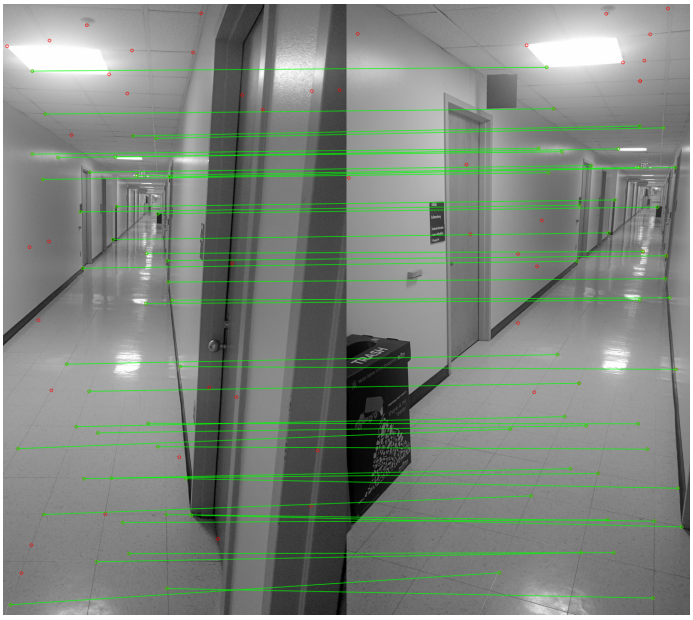

Fig. 27. RANSAC Test Set 3

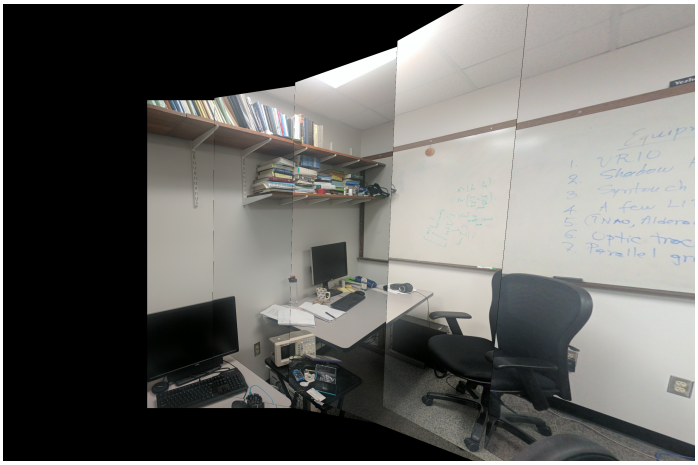

Fig. 28. Test Set 2 Images Stitch

this the photometric loss as it is the difference between the image pixel intensities. We trained the unsupervised network for 33 epochs with a SGD learning rate of .005, similar to our supervised network. The training and validation losses from each epoch are shown in Figs. 42 and 43, respectively. The network was then evaluated on the provided test set (1000 images) and resulted in an average MSE loss of 144 using the same evaluation metric as the supervised approach for easier comparison.

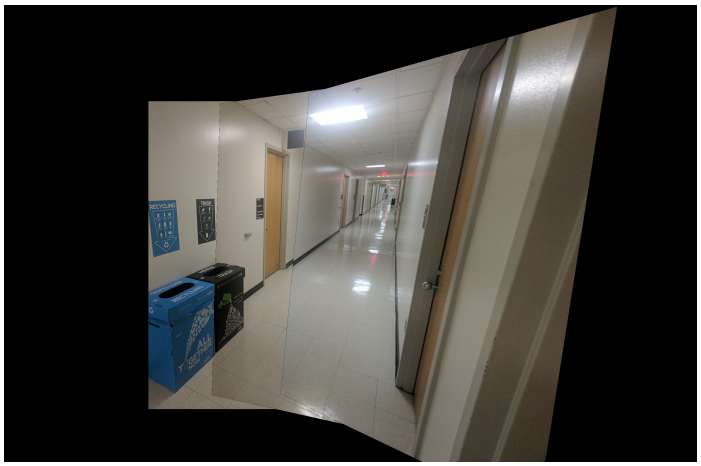

Fig. 29. Test Set 3 Images Stitch

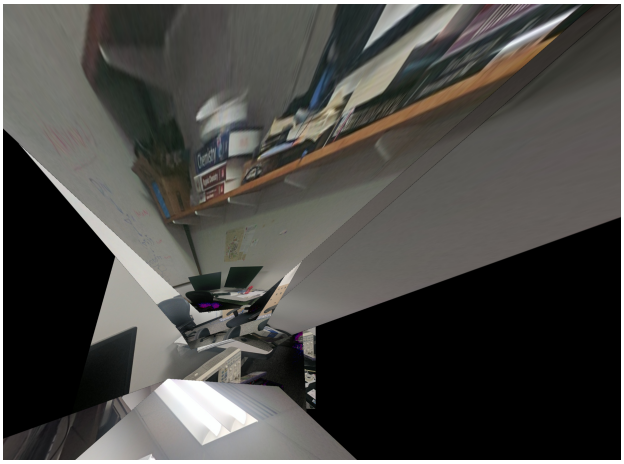

Fig. 30. Stitching result on the dataset using the supervised homography network

Fig. 31. Stitching result on the train set using the unsupervised homography network

Fig. 32. Original image (top) and 5 warp pairs (bottom), with $\sigma = 10$

Fig. 33. Original image (top) and 5 warp pairs (bottom), with $\sigma = 10$

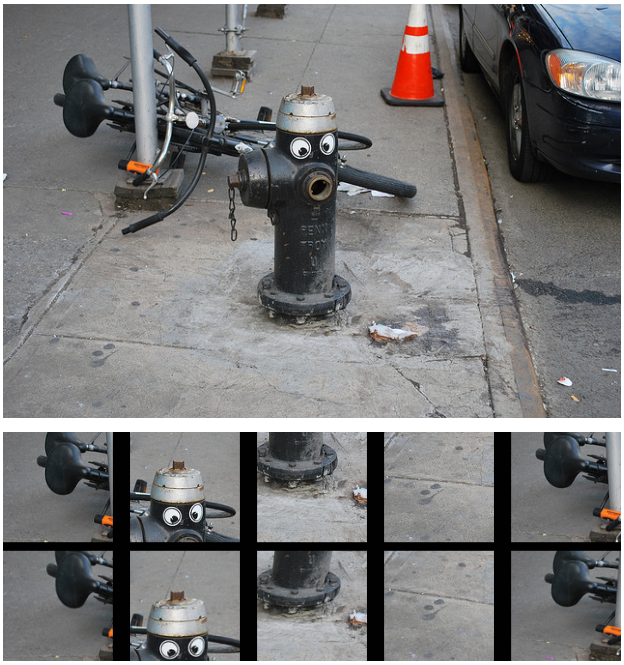

Fig. 34. Original image (top) and 5 warp pairs (bottom), with $\sigma = 10$

Fig. 35. Original image (top) and 5 warp pairs (bottom), with $\sigma = 10$

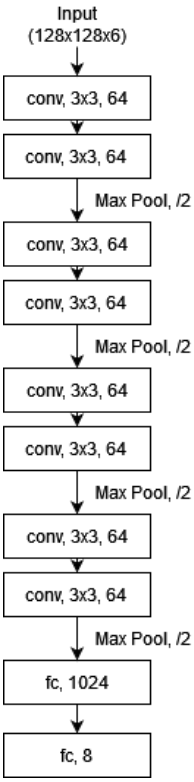

Fig. 36. HomographyNet Architecture

Fig. 37. Original image (top) and 5 warp pairs (bottom), with $\sigma = 25$

Fig. 38. Original image (top) and 5 warp pairs (bottom), with $\sigma = 10$

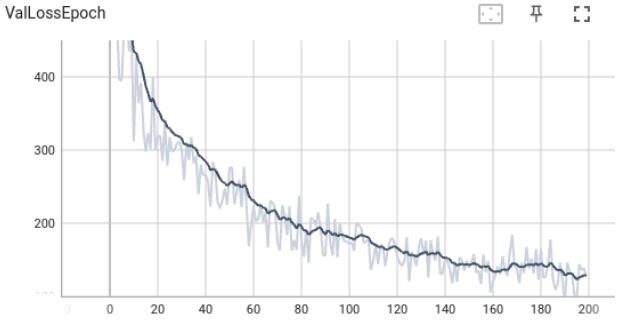

Fig. 39. Validation Loss for Supervised Model ($\sigma = 10$)

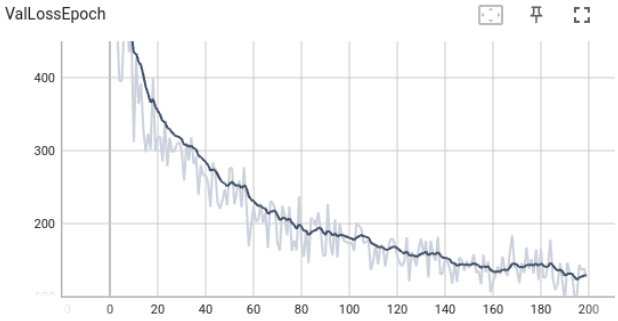

Fig. 40. Validation Loss for Supervised Model ($\sigma = 25$)

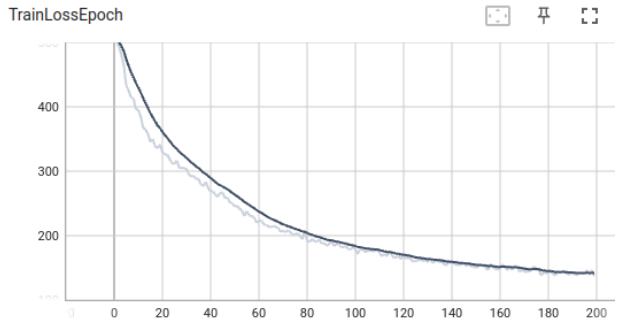

Fig. 41. Training Loss for Supervised Model ($\sigma = 25$)

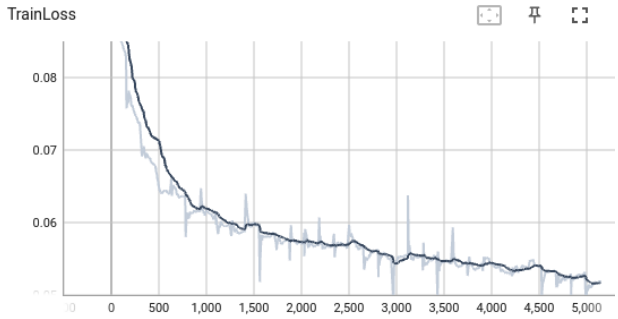

Fig. 42. Training Loss for Unsupervised Model

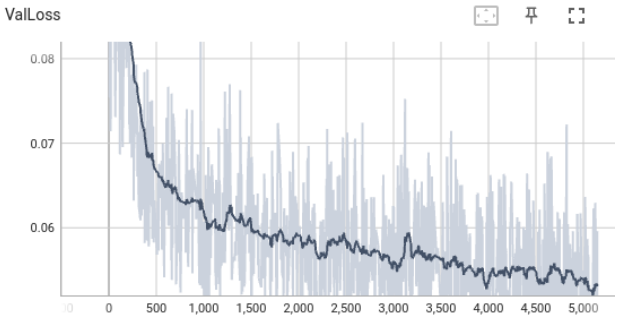

Fig. 43. Training Loss for Unsupervised Model