Project 1: My AutoPano (USING 2 LATE DAYS)

Nitin Suresh
School of Electrical and
Computer Engineering
University of Maryland - College Park
UID: 113638855

Jo Shoemaker
Department Of Computer Science
University of Maryland - College Park
UID: 115506787

I. Phase 1

Our classical panorama stitching approach involved four phases: corner selection, corner matching, homography estimation, and image warping and stitching.

A. *Corner Selection*

The OpenCV `cornerHarris` function with a block size of 9, aperture of 3, and $k = 0.04$ was used to score corners. Adaptive Non-Maximal Selection was used to select 250 evenly distributed prominent features in each image, including images that were already "absorbed" into a larger panorama.

B. *Corner Matching*

The surroundings of corners were represented by an $8{\times}8{\times}3$ square downsampled from a $40 \times 40 \times 3$ median blur of the corner's immediate vicinity. The degree of match of all corners was compared and corners that were both each other's top match and had no rival matches with an equally good matching score were considered valid pairs. An image was rejected for addition to the panorama if fewer than ten valid matches were found.

C. *Homography Estimation*

The RANSAC algorithm was used to sample the most robust homography. If after 500 iterations no homography could be found that included at least four inliers (within a tolerance of about 40 pixels' distance), the image was rejected for addition to the panorama.

D. *Image Warping and Stitching*

The openCV function `seamlessClone` was used to blend the new image into the current panorama. In order to make the blending easier, the new image was first added "beneath" the panorama in the correct location, and then cloned in on top of itself using `seamlessClone`.

II. Phase 2

In this section of the project, we implemented both supervised and unsupervised approaches towards estimation of the homography between two images. These models learn all the feature extraction and combination steps to give an the 8-pt homography output.

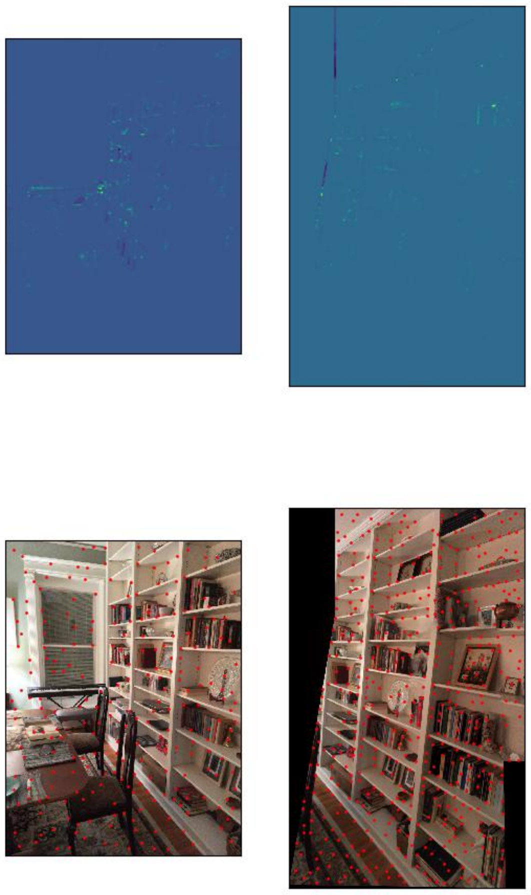
Fig. 1. Top: initial corner ratings for CustomSet1 using `cornerHarris`. Bottom: Corners selected by ANMS. Note the right image has twice as many dots as the left because it is made of two images.

A. *Supervised Approach*

1) Data Generation: Data generation for this step was carried out by extracting random patches of size 128x128 from the first image and identifying a random perturbation matrix with maximum values of [-32,32] for each point. By

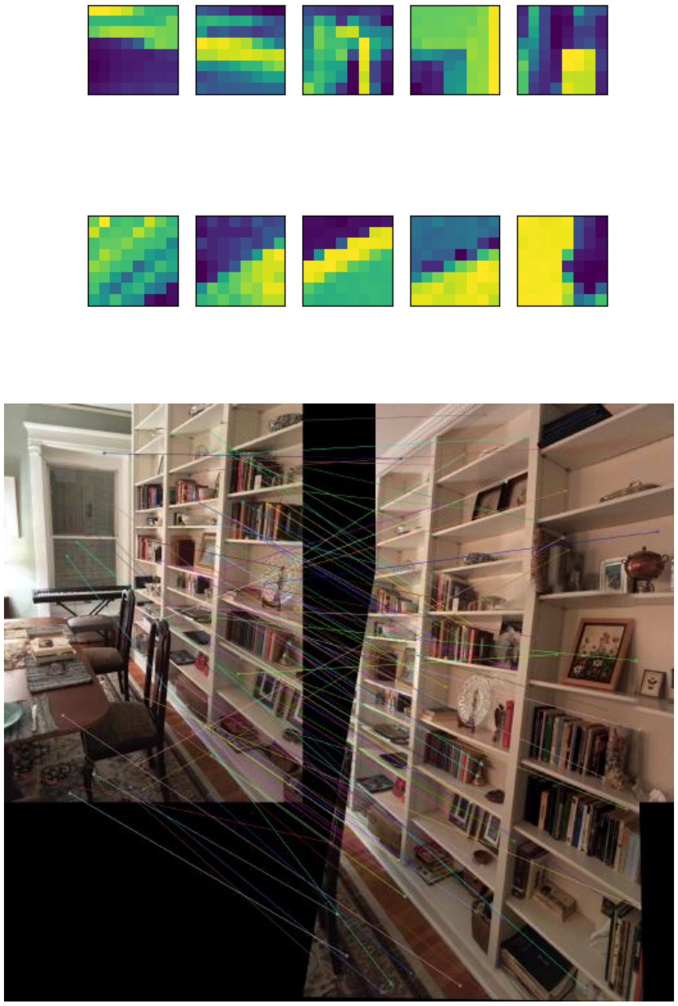
Fig. 2. Top: grayscale portions of a random five corner patches in two images in CustomSet1. Bottom: Matched corners in CustomSet1.

calculating the inverse homography and using that to transform the original image, the second patch was generated. This (128,128,2) stacked image was the input to the net, and the 8-pt homography was the label.

2) Network Architecture: The network architecture used was similar to the one used in [1], with a sequence of 4 sets of 2 convolution filter sets each, with max-pooling in between, and ReLU activations. Additionally, we included batch normalization layers after every max-pooling layer. The network architecture is displayed in Fig. 5.

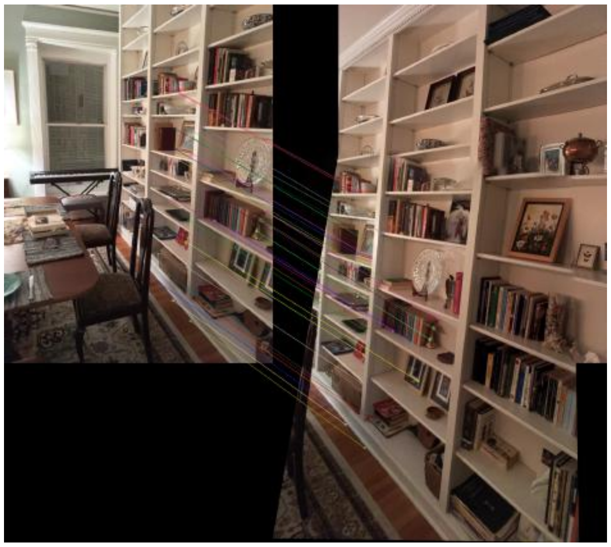
Fig. 3. Inlier corners in CustomSet1 after RANSAC.

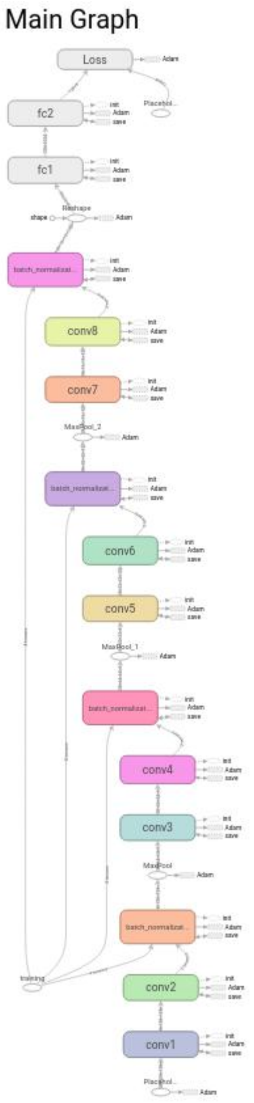

Fig. 5. Network Architecture

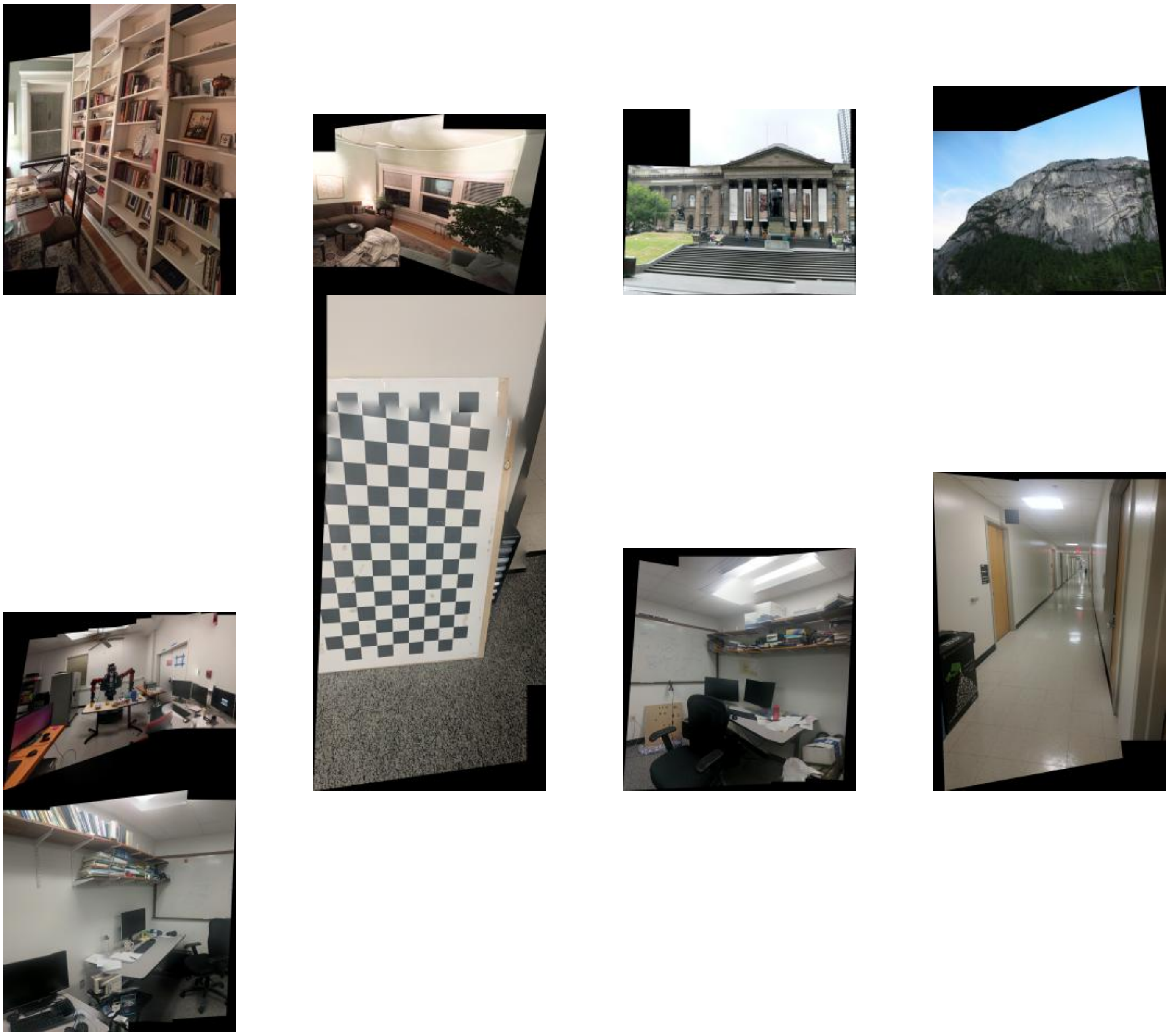

Fig. 4. Largest panoramas produced from each image set.

3) Training and Results: Training was carried out over 50 epochs (of size 5000 images), with a mini-batch size of 128 images. The loss metric used was the 2-norm of the difference tensor between the ground truth H4Pt label and the calculated H4Pt from the network. The Adam optimizer was used with a learning rate of 1e-4. The epoch loss, iteration loss and validation loss are displayed in Fig. 6, Fig. 7, and Fig. 8 respectively. The validation loss was calculated after each epoch on the provided validation dataset. Two metrics were calculated on the test set - the EPE (average L2 error) for the supervised approach was 59.81, and the average pixel error (L1 loss) was 17.83 pixels. The forward pass run-time was 0.069s.

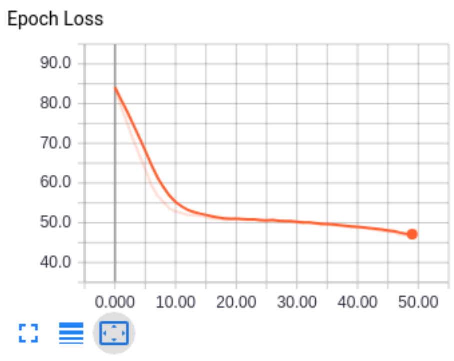

Fig. 6. Epoch Loss (supervised)

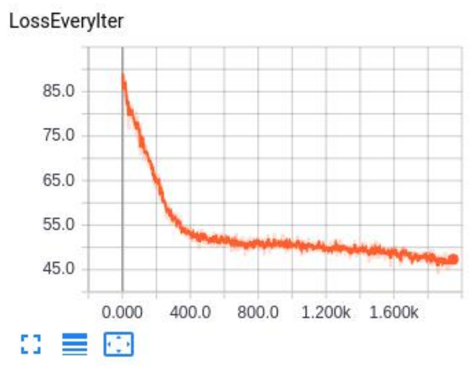

Fig. 7. Iteration Loss (supervised)

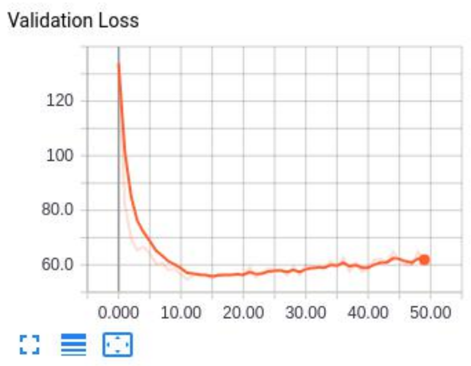

Fig. 8. Validation Loss (supervised)

B. *Unsupervised Approach*

1) Data Generation: The unsupervised approach used the same stack of images but also included usage of the original image, which was warped by the spatial transformer.

2) Network Architecture: The network architecture used was the same as in the supervised approach. The two additional components here were the tensor direct linear transform implementation, and the spatial transformer network, which were adapted from [2]. These two modules perform the functions of homography estimation and image warping, in a differentiable manner, which provides a pathway for Tensorflow to estimate gradients backwards.

3) Training and Results: Training was carried out over 63 epochs (of size 5000 images), with a mini-batch size of 128 images. The loss metric used was the photometric loss calculated between the warped image and the second image. This warping was carried out by the spatial transformer network, with the homography estimated using the tensor direct linear transform. The Adam optimizer was used with a learning rate of 1e-4. The epoch loss, iteration loss and validation loss are displayed in Fig. 9, Fig. 10, and Fig. 11 respectively. The validation loss was calculated after each epoch on the provided validation dataset. Training was much slower for the unsupervised approach, and did not reach the levels of accuracy seen in the case of the supervised approach. On the test set, the EPE (average L2 error) for the unsupervised approach was 146.62, and the average pixel error (L1 loss) was 42.88 pixels. The forward pass run-time was 0.072s.

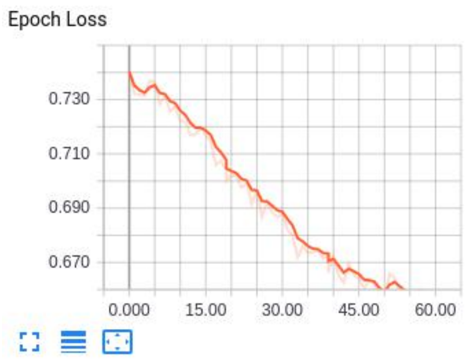

Fig. 9. Epoch Loss (unsupervised)

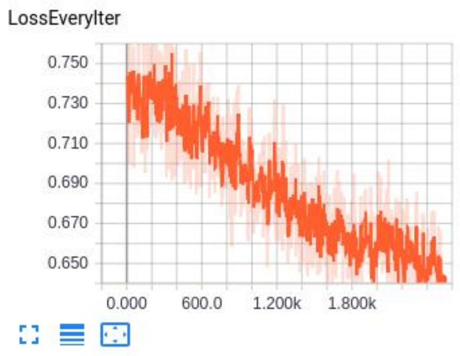

Fig. 10. Iteration Loss (unsupervised)

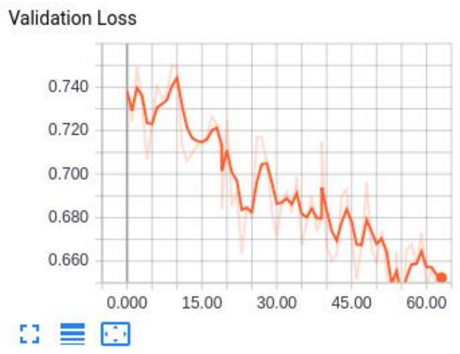

Fig. 11. Validation Loss (unsupervised)

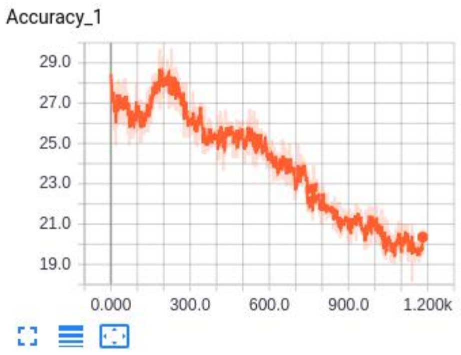

Fig. 12. Pixel accuracy (L1-loss) (unsupervised)

C. Panorama Stitching

Best results are shown for panorama stitching of the test sets using the two networks. Both networks were succesful in stitching panoramas for the 1st and 3rd sets, but did not give meaningful image results for the 2nd and 4th testsets.

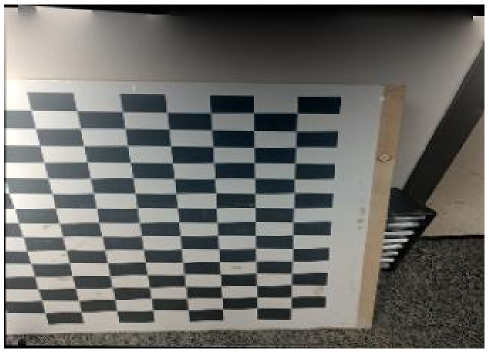

Fig. 13. TestSet1 panorama (supervised)

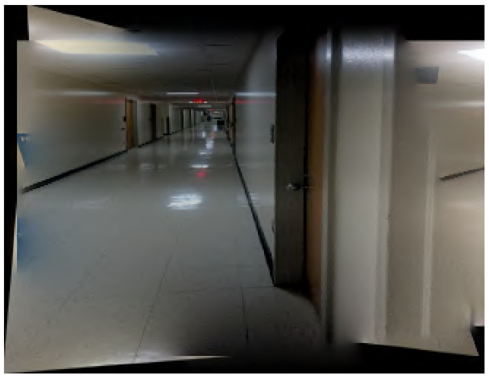

Fig. 14. TestSet3 panorama (supervised)

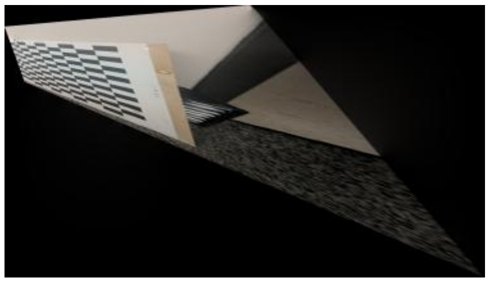

Fig. 15. TestSet1 panorama (unsupervised)

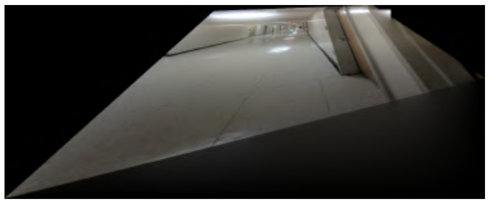

Fig. 16. TestSet3 panorama (unsupervised)

III. References

[1] DeTone, Daniel, Tomasz Malisiewicz, and Andrew Rabinovich. ”Deep image homography estimation.” arXiv preprint arXiv:1606.03798 (2016).

[2] github.com/tynguyen/unsupervisedDeepHomographyRAL2018