MyAutoPan

Computer Vision (RBE549) Project 1

Hrishikesh Pawar
MS Robotics Engineering
Worcester Polytechnic Institute
Email: hpawar@wpi.edu

Tejas Rane
MS Robotics Engineering
Worcester Polytechnic Institute
Email: turane@wpi.edu

I. Phase 1: Traditional Approach

The main goal of this phase is to generate a continuous panorama by merging multiple overlapping images using traditional techniques. This conventional method underscores the significance of repeated local features in each image for effective stitching. The process in the classical approach involves the following stages:

1) Corner Detection
2) Adaptive Non-Maximal Suppression (ANMS)
3) Feature Descriptor
4) Feature Matching
5) RANSAC for outlier rejection and to estimate Robust Homography
6) Stitching Images

Each section provides a detailed explanation of the methodology employed and the resulting output of the images.

A. Corner Detection

Corner detection lays the groundwork for all subsequent stages of the image stitching process. Corner detection is a fundamental and crucial step in image stitching, since they are robust to various transformations like rotation and scaling, making them ideal for matching and aligning multiple images.

Our implementation focuses on identifying corners in the images using the OpenCV Harris corner detection method. We used a blockSize of 7 as the neighbourhood size for corner detection. Additionally, the Sobel kernel aperture size is 11 with a Harris detector free parameter as 0.04.

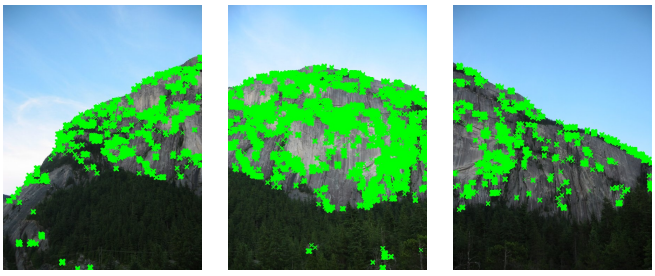

Fig. 1: Harris Corner Detection

B. Adaptive Non-Maximal Suppression (ANMS)

As seen in Figure 1 Harris Corner Detection is effective in identifying a large number of potential corner points in an image. However these set of points include many closely clustered corners and hence is quite dense. Therefore, we employed ANMS to address this issue by selecting a subset of the corners based on their strength and spatial distribution. ANMS is divided in two main steps:

1) Finding the local maxima from the Harris detector response.
2) Selecting a subset of corners from those obtained from [1] such that selected corners are evenly distributed across the image.

For each corner detected in Step 1, the ANMS algorithm calculates a 'robustness' measure. This measure is based on the corner's strength and its spatial relation to other corners. Robustness of a corner is determined by the Euclidean distance to the nearest corner that has a stronger Harris response. A larger distance implies a higher robustness, indicating that the corner is not only strong but also spatially isolated from other strong corners. In areas where corners are densely clustered, many corners will have a smaller robustness measure since they are close to other strong corners. The corners are then sorted based on their robustness measures. Those with the highest measures are considered the most significant. From this sorted list, a predetermined number of top corners are selected. Refer Figure 2 and Figure 3.

C. Feature Descriptor

Following the selection of feature points through ANMS, the next step is to encode the information at each feature point into a vector, commonly known as a feature descriptor. Feature descriptors uniquely represent a feature point, facilitating the matching process in later stages. For each feature point, we considered a 41×41 patch centered around it and applied Gaussian blur (kernel size = 5) to it. For considering the corners at the edges we considered padding of 20 around the image. We sub-sampled the blurred patch to reduce the size of the blurred image from 41×41 to 8×8. This significantly reduces the dimensionality of the feature descriptor resulting into computational efficiency while retaining essential information. The resulting 8×8 matrix is vectorized into a 64×1 normalized vector.

D. Feature Matching

Next step is establishing correspondences between key-points in two images to be stitched, based on their respective

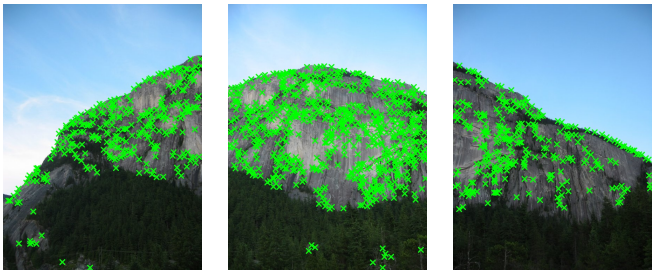

Fig. 2: Harris Corner Detection after Local Maxima

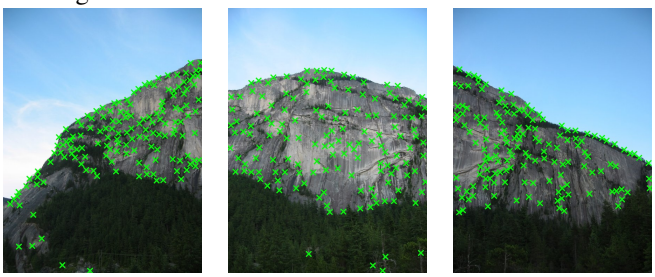

Fig. 3: ANMS Output

feature descriptors. For each feature descriptor of the first image we calculate the Euclidean distance to every descriptor in the second image. The closest and second-closest matches are identified for each descriptor. To ensure the distinctiveness of these matches, Lowe's ratio test is applied. This test compares the distances of the closest and second-closest matches; if the ratio of these distances is below a threshold (for us it is 0.75), the match is deemed valid. This ratio test is critical in reducing false positives – it ensures that the selected match is not only the closest but also significantly closer than the next best match, indicating a high likelihood of it being a correct match. Matches passing this test are considered as valid matches.

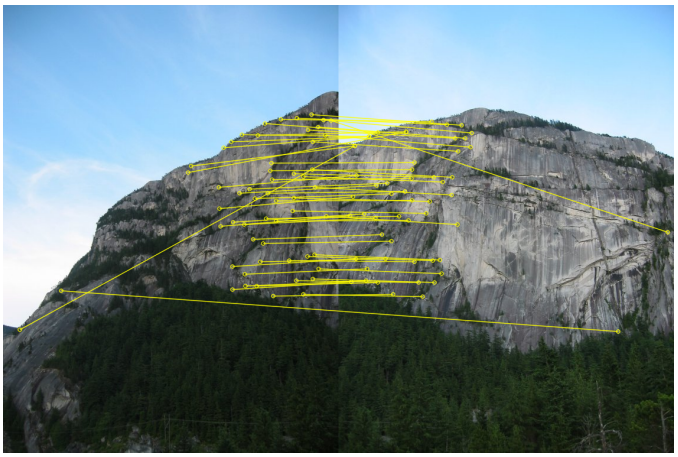

Fig. 4: Feature matching

E. Random Sampling and Consensus (RANSAC)

The feature correspondences obtained are not entirely correct and consist of a lot of outliers. It is essential to identify and remove the outliers to ensure accurate homography estimation and ultimately image alignment. For this purpose we implemented RANSAC with the following steps:

1) **Random Selection and Homography Computation:**
   - Select four pairs of matched features randomly (`pi` from image1, `p'i` from image2).
   - Compute the homography `H` using these points.
2) **Inlier Computation:**
   - For each feature pair, calculate if it is an inlier by checking if the Sum of Squared Differences (SSD) between the transformed point `Hpi` and the corresponding point in the second image `p'i` is less than a threshold τ.
3) **Iterations:**
   - Repeat the above two steps for a maximum of `nmax` iterations (we are iterating for 1000 iterations), or until a certain percentage of inliers (we have set a threshold of 90%) is achieved.
4) **Finilazing Inliers:**
   - Retain the largest set of inliers obtained from above iterations.
5) **Compute the final Homography:**
   - Considering all set of inliers obtained from Step 4 re-compute the final homography matrix.
6) **Check if the pair of images are valid for stitching:**
   - Considering all set of inliers obtained from Step 4 we additionally check if the number of inliers are above a threshold of 4 to deem the stitch as valid otherwise invalid.

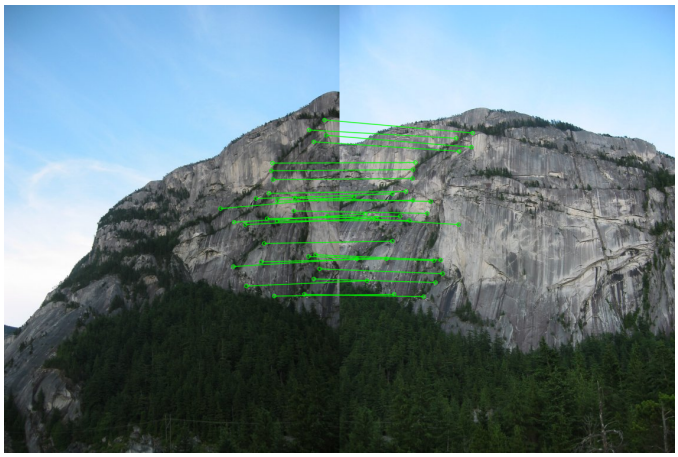

Fig. 5: Feature Matches after RANSAC

F. Stitching and Blending

Once the relative homography between the pair of images is determined, the next step is to stitch the images. For further discussion lets consider that the stitching occurs from left to right. Given two images image1 and image2, image1 is warped to be stitched with image2. With this context, steps below outline out approach:

1) **Warping image1 with the Homography matrix:**
   - The homography matrix obtained from RANSAC is used to transform the corner points of image1.
2) **Calculating the canvas dimensions:**
   - The transformed points of image1 and the original points of image2 are considered to determine the total space that the stitched image will occupy.
   - This combination of points helps in understanding the extent to which the images overlap and the additional space required for the panorama.
3) **Computing the Translation Matrix:**
   - From the set of combined points we determine the minimum x and y coordinates. These coordinates indicate how much the stitched image extends in the negative x and y coordinates.
   - A translation matrix is calculated considering these values which effectively shifts the transformed image (in this case image1) rightwards and/or downwards such that entire image falls within positive quadrant.
4) **Applying translation and stitching the image:**
   - This step involves a warp perspective transformation, which adjusts the image based on combined homography and translation matrices. The result is an image where image1 is transformed and aligned with image2.

G. Results- Success and Failure Case Analysis

Train Set Cases: Our implemented algorithm successfully executed the image stitching process for Train Sets 1 and 2. The stitched output of Train Set2 is illustrated in Figure 7, while that of Train Set1 is depicted in Figure 8. However, we encountered challenges with Train Set3, with multiple images. Initially, we adopted a linear stitching approach, progressing from left to right. This method proved ineffective beyond the fourth image in the sequence, leading to significant distortion and warping.

To address this issue, we revised our approach by initiating the stitching process from a centrally located image within the sequence. Train Set3 consists of eight images, with Image4 designated as the central reference point. The revised stitching methodology was executed in the following steps:

- Starting from Image 4, the algorithm proceeded to stitch subsequent images in the rightward direction of the sequence.
- Starting again from Image 4, the stitching process was conducted in the leftward direction.
- The resulting left and right stitched panoramas were then combined to form a panaroma.

This strategy successfully stitched Images 2 through 7. However, Images 1 and 8 presented challenges, again resulting in excessive warping. The final stitching outcome for Train Set 3 is displayed in Figure 9.

Test Set Cases:

Fig. 6: Train Set 2, first 2 images stitched.

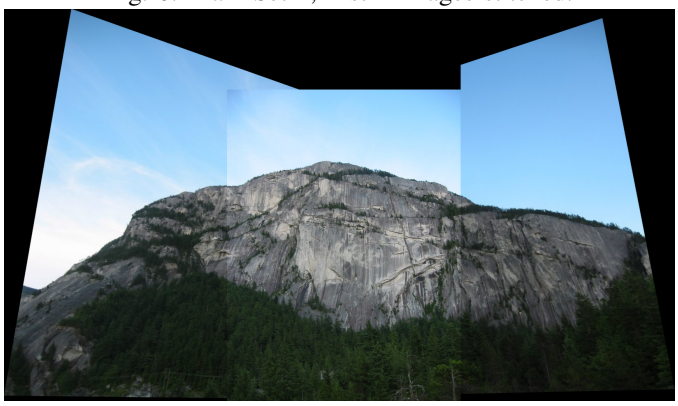

Fig. 7: Train Set 2 full panorama

Fig. 8: Train Set 1 full panorama

TestSet1: For this set we were able to stitch the first two images. However for the images further in the sequence there are non-unique correspondences in the image and hence the images are not suitable for stitching. Figure 10(a) shows the stitch of Image1 and Image2 in the TestSet. Figure 10(b) shows the non-unique correspondences in the further images.

TestSet2: Our algorithm is developed for sequential image stitching (left to right or vice-versa) or from a central image

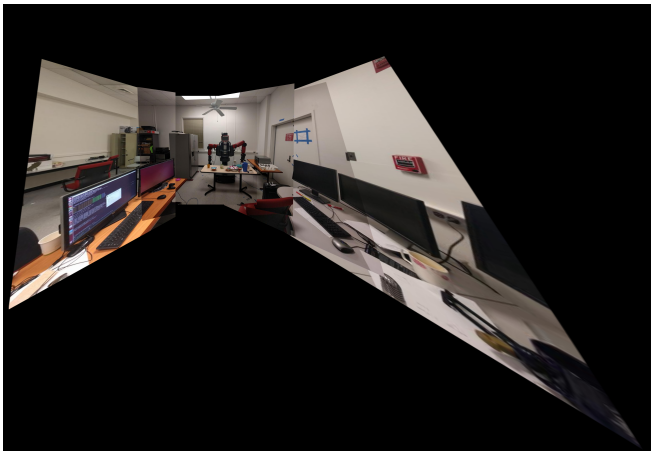

Fig. 9: Train Set 3 images 2 to 7 stitched

(a) (b)

Fig. 10: TestSet1 Images

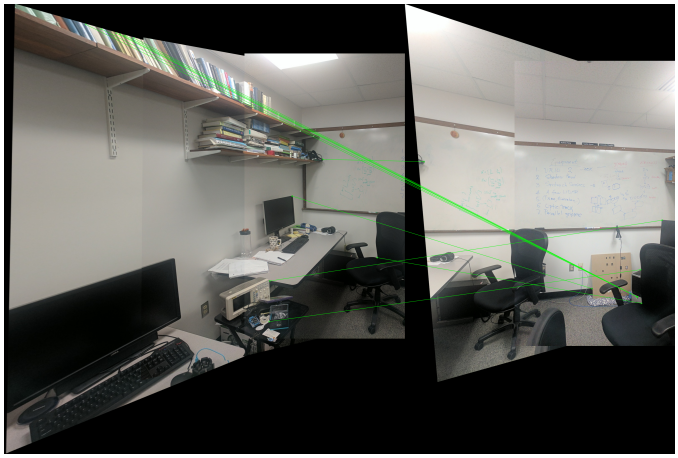

Fig. 11: TestSet2 Stitch Failure

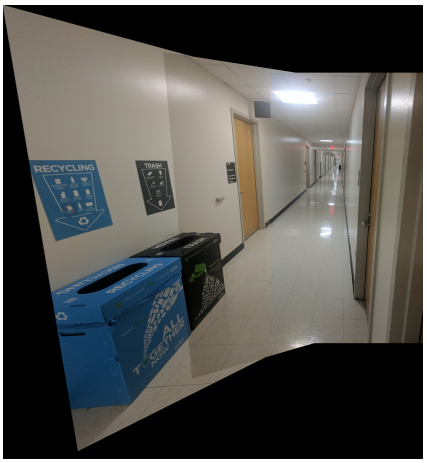

Fig. 12: TestSet3 Panaroma

to each of the left or right directions. For every image we have incorporated a feature-matching check. If the number of feature matches after RANSAC is below a threshold of 4 (since we need four points for calculating homography), the image is deemed unsuitable for stitching, and the algorithm proceeds to the next image in the sequence. So our algorithm is able to stitch the images in groups. We obtained a stitch of image groups 1-2-3, 4-5-6 and 7-8 and later stitched these groups together. However, we feel that the group stitch (1-2-3 with 7-8) fails due to two reasons; first due to very less overlap between the images and second due to less features (corners) due to a white background as seen in Figure 11.

TestSet3: Our algorithm was able to stitch TestSet3 successfully. Figure 12 shows the final stitched output.

TestSet4: This set consists images seen in Figure 13. We stitch images 1-3 successfully with the strategy of stitching sequentially from left to right. Then as mentioned in TestSet2 we are checking the inlier count after RANSAC before final stitching. If the number of feature matches after RANSAC is below a threshold of 4 (since we need four points for calculating homography), the image is deemed unsuitable for stitching, and the algorithm proceeds to the next image in the sequence. So our algorithm stitches images 1-3 in a full panaroma as seen in Fig. 15 and skips Image4 and Image5. Figure 14 shows the feature matches of the 1-3 stitch with Image5 after RANSAC.

Custom Set Cases:

Custom Set 1: This set contains four images seen in Figure 16:

The stitched panorama for these images is seen in Figure 17:

Custom Set 2: This set contains four images seen in Figure 18:

The stitched panaroma for these images is seen in Figure 19:

II. Phase 2: Deep Learning Approach

In this section, we explore two different deep learning techniques to find the homography between two images to perform panorama stitching. Specifically, we use Supervised and Unsupervised training regimes to estimate the 4-point homography matrix (H_{4PT}) between the two images.

(a) Image 1 (b) Image 2 (c) Image 3

(d) Image 4 (e) Image 5

Fig. 13: TestSet4 Images

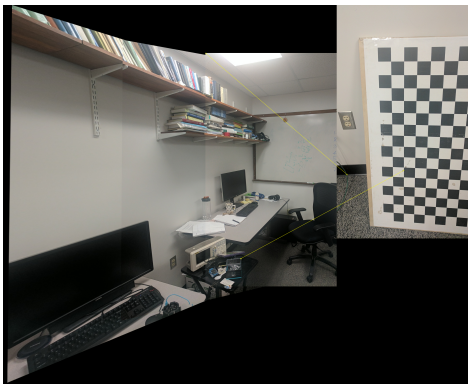

Fig. 14: Feature Matches for TestSet4

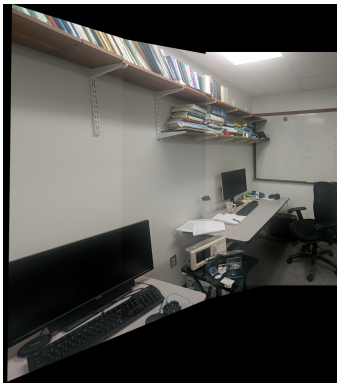

Fig. 15: TestSet4 Panorama

(a) Image 1 (b) Image 2 (c) Image 3

(d) Image 4

Fig. 16: CustomSet1 Images

A. Data Generation

The training data used for these models is generated from a small subset of the MS-COCO dataset. The images were resized to 320×240 pixels. We created a buffer of 30 pixels on all sides of the image and then generated 10 patches per image from the center region, to ensure that we are not extracting the patch from outside the image after warping. Let each patch be denoted by P_A, and the corners of the patch be denoted by C_A. The corners of these patches were then perturbed randomly in the range of $[-32, 32]$ pixels, and a random translation less than 32 pixels was added to generate the corresponding patches P_B, with corners C_B. We then calculate the 4-point homography matrix for these patches, given by $H_{4PT} = C_A - C_B$. We create a dataset containing $50,000$ images in the training set and $10,000$ images in the validation set. The labels are saved as a dictionary of H_{4PT} and C_A for each image. An example pair of patches P_A and P_B is shown in Figure 20.

B. Supervised Approach

The network used for the supervised approach has a VGG-like architecture with 8 convolutional layers and 2 fully connected layers. The network architecture is shown in Figure 21. The input to the network are 3 channel RGB patches P_A and P_B stacked in the depth dimension, with the input size of $128 \times 128 \times 6$. The output of the network is an eight dimensional vector which is defined as the estimated 4-point homography matrix between the corners of patches P_A and P_B. We use the RMSE loss between the ground-truth and estimated 4-point homography matrices to train the network. The overview of the supervised approach is shown in Figure 22.

The network is trained for 100 epochs with a batch-size of 512. We use AdamW optimizer with a learning rate of 0.001 and regularization strength (weight decay) of 0.0001. A

Fig. 17: CustomSet1 Panorama

(a) Image 1 (b) Image 2 (c) Image 3

(d) Image 4

Fig. 18: CustomSet2 Images

learning rate scheduler is also use which decreases the learning rate exponentially by a factor of 0.99 after each epoch. The training and validation loss curves are shown in Figure 23. The network achieves the best train loss of 7.495 and a validation loss of 8.65. This network took about 4 hours 15 minutes to train.

C. Unsupervised Approach

The overview of the supervised approach is shown in Figure 28. The first part of the network is similar to the supervised network. Here, the input to the network are grayscale patches P_A and P_B stacked in the depth dimension, with the input size of $128 \times 128 \times 2$, and the output is the estimated 4-point Homography matrix. This is then added with the patch corners C_A from the training data, and the result is passed through a TensorDLT function, which converts them to the Homography matrix H. Instead of using the Spatial Transformer Network (STN), we use the *kornia.geometry.transform.warp_perspective* function, which takes the patch P_A and the Homography matrix H as input, and produces the estimated warped patch P_B as output. The

Fig. 19: CustomSet2 Panaroma

P_A P_B

Fig. 20: An example of patch P_A and randomly perturbed patch P_B.

estimated and the ground-truth patches P_B are then compared to calculate the photometric (L1) loss.

The network is trained for 50 epochs with a batch-size of 256. We use AdamW optimizer with a learning rate of 0.0001 and regularization strength (weight decay) of 0.0001. A learning rate scheduler is also use which decreases the learning rate exponentially by a factor of 0.99 after each epoch. The training and validation loss curves are shown in Figure 24. As it is evident from the loss curve, the network struggles to converge. This network took about 3 hours to train.

After debugging, the main reason for the improper convergence of the network was found to be the improper warping of patches to create the labels. In our implementation, we pass the patches P_A and P_B to the initial regression model. The resulting 4-point Homography matrix H_{4PT} is then used to warp the patches P_B to find the photometric loss between the warped patches and the ground-truth patches. But, instead of warping the patches, the correct approach should be to warp the original image and re-crop the image with the generated 4-point Homography matrix H_{4PT}.

D. Results and Discussion

To evaluate the performance of the two networks, we performed comparison studies on the estimated homographies from both the networks on 4 images. The results of this study are shown in Figure 25. As it can be seen, the Supervised network performs better at estimating the homographies as compared to the Unsupervised network.

Table I shows a quantitative comparison between the two networks. Here we compare the average training, validation and testing loss for the two networks. The loss for the supervised network is calculated as the RMSE loss between the ground-truth and estimated 4-point homography matrices. For the unsupervised network, the loss is calculated as the

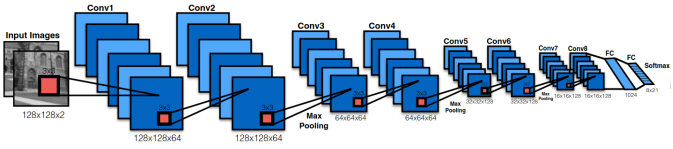

Fig. 21: Network architecture of the supervised network.

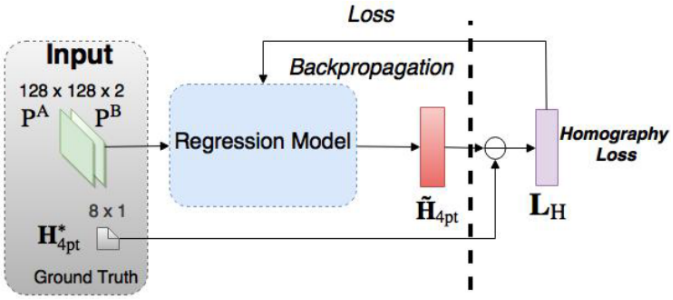

Fig. 22: Training flow chart for supervised network.

photometric loss (L1 loss) between the patches P_B warped with ground-truth homography and the estimated homography.

	Train Loss	Val Loss	Test Loss	Inference Time
Supervised	7.495	8.644	7.480	2.2 ms
Unsupervised	46.339	46.462	48.023	6.4ms

TABLE I: Table comparing Supervised network and the unsupervised network performance.

We also attempt to stitch images using the homographies estimated from the supervised and the unsupervised neural networks, and compare it with the results of the classical stitching method described in Phase 1. The results of the classical stitching method are shown in Figure 26. As it can be seen, the panorama generated is quite good. But when we try to generate the homographies from the deep neural networks, we see that the homographies estimated are incorrect, even if the supervised network produced a low testing loss. The homography matrix has a stronger rotation effect than a translation effect. Results of stitching 2 images with the Supervised network are shown in the Figure 27. To generate this result, the images were first cropped from their high-definition resolution of 1920×1080 pixels to 1000×1000 pixels. Then, the images were resized to 128×128 pixels as an input to the neural network.

We believe that the result is incorrect because the training data used has more examples with rotational perturbations than pure translation perturbations. Another reason could be that the training data is more feature rich as compared to the testing images. This performance can also be improved by training the network more thoroughly with a wider range of data, both in term of features as well as the rotational and translational perturbations.

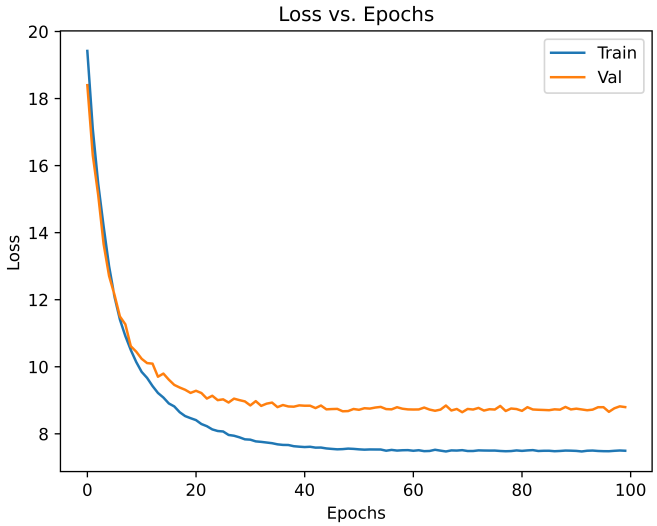

Fig. 23: Training and validation loss for the supervised network.

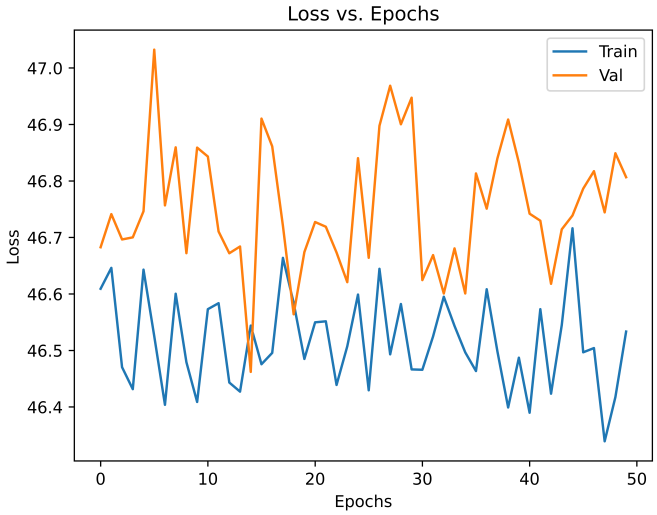

Fig. 24: Training and validation loss for the unsupervised network.

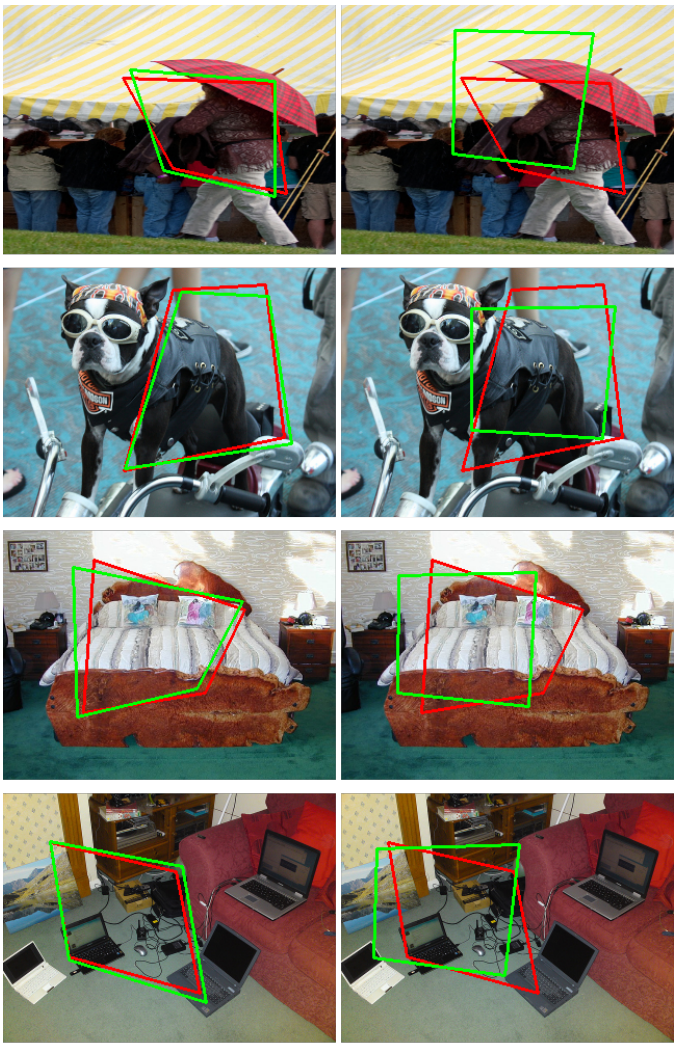

Fig. 25: Comparison of the estimated homography from the Supervised and Unsupervised networks, on 4 images. (Red is ground-truth, green is estimate)

Fig. 26: Images from ”trees” dataset stitched using classical panorama stitching.

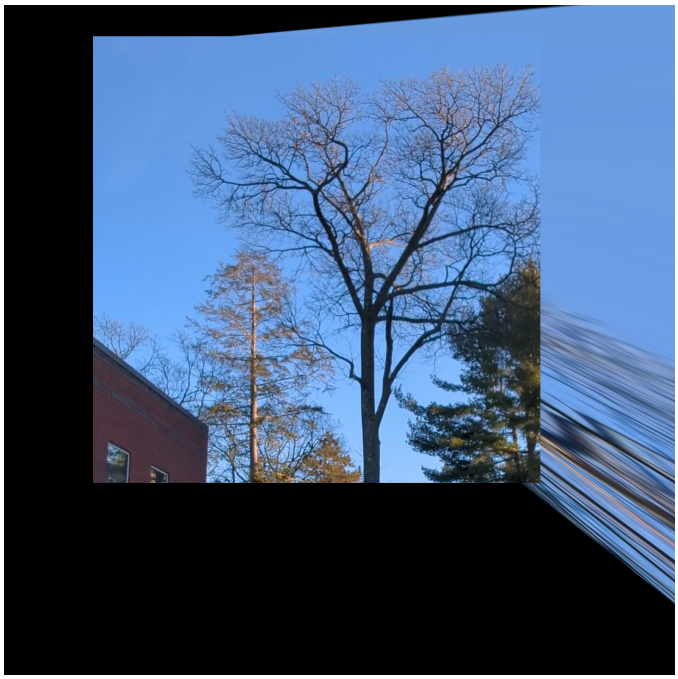

Fig. 27: Images from ”trees” dataset stitched using Homography estimation from the Supervised model.

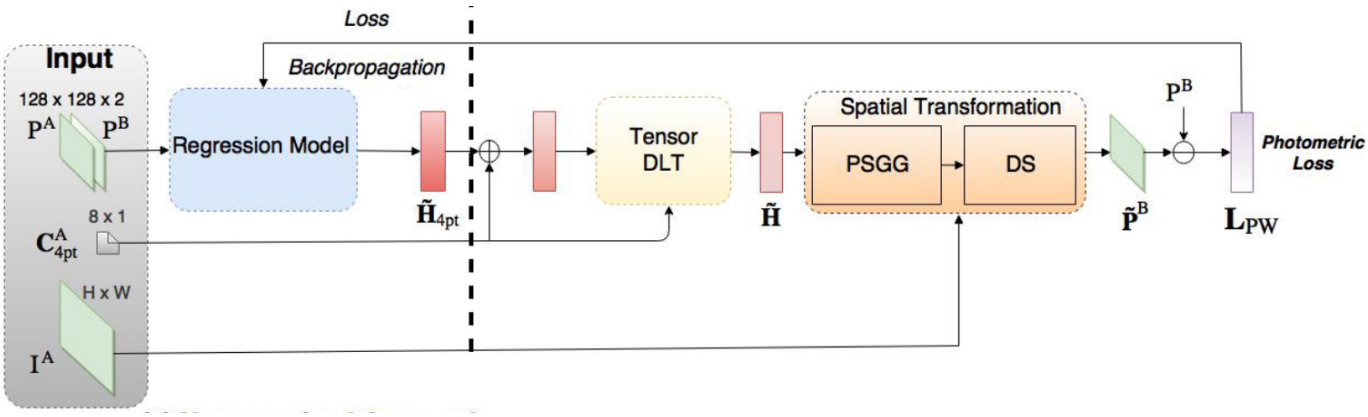

Fig. 28: Training flow chart for unsupervised network.