RBE/CS 549 Computer Vision Project 1: My-Auto-Pano

Puneet Shetty
MS in Robotics
Worcester Polytechnic Institute
Email: ppshetty@wpi.edu

Edwin Clement
MS in Robotics
Worcester Polytechnic Institute
Email: eclement@wpi.edu

***Abstract*—A number of researchers have found that the discipline of computer vision makes for an exciting area of study.The problem of stitching photos together to make a panorama is well-known in this discipline, and it can be tackled using a variety of image processing algorithms.The paper discusses the technique of stitching photos together to make a panorama using both classical and deep learning approaches.The traditional approach focuses on extracting and matching common features in photos, as well as blending them together based on these elements.The report briefly highlights our solutions for Project 1. The report is organized into two sections. The first section looks at the usual method for finding a homography matrix between two sets of photos. The second section details the implementation of both supervised and unsupervised deep learning approaches for estimating homography between synthetically created data.**

Index Terms*— *MyAutoPano, Adaptive Non Maximal Suppression, Corner Detection, Random Sampling and Consensus, Homography, Panorama stitching, Image Stitching.

I. PHASE 1: TRADITIONAL APPROACH

In this section of the research, we investigate the traditional ways for creating a panoroma by stitching image sequences. Each subsection describes in detail the approach used and the results of the photos. Feature-Based Methods are used to extract features that appear in common sections of photos. We match these feature keypoints and estimate the Homography between the two images. We may utilize this Homography to warp the second image relative to the first, which will then be used to stitch the images together with the reference image. To address issues with illumination or unexpected artifact creation during picture stitching, we use blending algorithms to flawlessly mix the images. To summarize, this project will include the following activities:

1) Detect corners using Harris corner detection as feature keypoints.
2) Adaptive Non-Maximal Suppression employing Sqaure Distance to reduce artifacts during warping and optimize Harris corner identification.
3) The Feature Descriptor standardizes keypoints and neighboring pixels and applies Gaussian blur to produce illlumination invariance at a specific level.
4) Feature matching identifies feature connection between two images by calculating the distance between the first and second best matches (lowest).

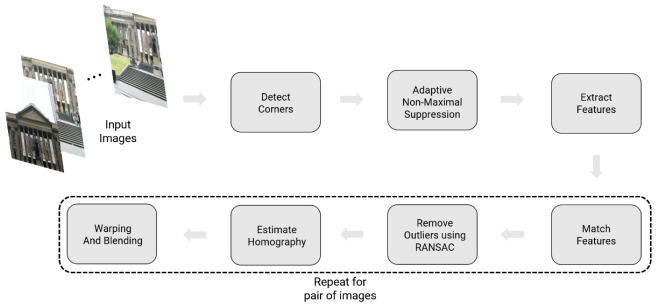


Fig. 1. Proposed Method

Fig. 2. Sample Inputs

5) RANSAC sets a threshold to ignore outliers and computes a homography matrix for each images.
6) Homography allows us to translate one image to the perspective of another, resulting in a stitched image that overlaps common sections.
7) Finally, we mix overlapping photos to prevent illumination inconsistency after stitching.

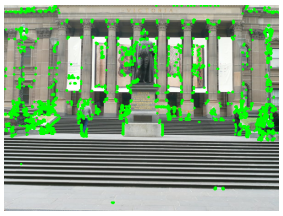
We'll be working our way through each of the previously listed steps.

A. Corner Detection

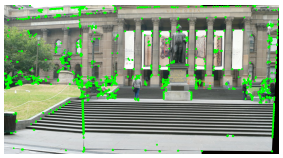
The goal is to build a relationship between photos using a set of features. Corners are the ideal method to do this because they are visible from multiple angles. We can detect as many corners as feasible from the given image and compare the attributes between them. This comparison would indicate how the photographs are geometrically related to one another. For this, we can use two methods:

i Harris Corner Detection: The OpenCV library's cv2.cornerHarris function employs the Harris method to detect corners. This method returns the corner strength

(a) Corners of Image 1



(b) Corners of Image 2



(c) Corners of Image 3

Fig. 3. Set 1

for each pixel, which is needed for adaptive non-maximal suppression, as mentioned in the later section.

ii *Shi-Tomashi Corners Detection:* The OpenCV library's cv2.goodFeaturesToTrack function employs the Shi-Tomashi method with a non-maximal suppression strategy to detect uniform corner points. Thus, the function returns corner co-ordinates rather than corner strengths. As a result, we were unable to apply the output from cv2.goodFeaturesToTrack to the adaptive non-maximal suppression outlined in the later section.

B. Adaptive Non Maximal Suppression

Now that we've recognized corners in each image, we need to determine the "best" corners. The best corners are ones that stand out from their local peer corners. Furthermore, we want these corners to be evenly distributed around the image so that we can achieve superior homographies. To do this, we use Adaptive Non Maximal Suppression (ANMS), which has two elements.

i Consider local maxima over corners.

ii Only examine corners with a greater distance from stronger corners.

The second point is the key component of the ANMS, which provides uniformly distributed corners from a collection of "clusters". The problem with Harris corners and other corner detectors is that they detect a cluster of corners rather than a single corner. This makes sense because a corner is a collection of pixels, and depending on the image's quality, many pixels can be appropriately classified as corners. Even if we create local maxima in the image, the cluster may still exist. To avoid this issue, ANMS selects a point that is maximally away from the other stronger corners, and when we sort the Nbest corners, we will be able to obtain a point from the cluster.

C. Feature Descriptor

To match features and identify common characteristics in photos, we must first generate feature vectors that encode all of the feature's information.To build the feature descriptor, we cut

Input : Corner score Image (C_{img} obtained using `cornermetric`), N_{best} (Number of best corners needed)
Output: (x_i, y_i) for $i = 1 : N_{best}$

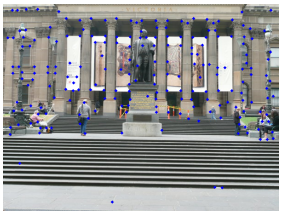
Find all local maxima using `imregionalmax` on C_{img};
Find (x, y) co-ordinates of all local maxima;
((x, y) for a local maxima are inverted row and column indices i.e., If we have local maxima at $[i, j]$ then $x = j$ and $y = i$ for that local maxima);
Initialize $r_i = \infty$ for $i = [1 : N_{strong}]$

```
for i = [1 : N_strong] do
    for j = [1 : N_strong] do
        if (C_img(y_j, x_j) > C_img(y_i, x_i)) then
            ED = (x_j − x_i)^2 + (y_j − y_i)^2
        end
        if ED < r_i then
            r_i = ED
        end
    end
end
```

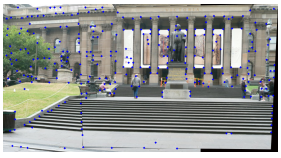
Sort r_i in descending order and pick top N_{best} points

Fig. 4. ANMS Algorithm

(a) ANMS of Image 1



(b) ANMS of Image 2



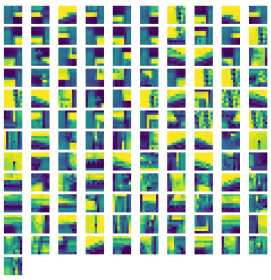
(c) ANMS of Image 3

Fig. 5. Set 1

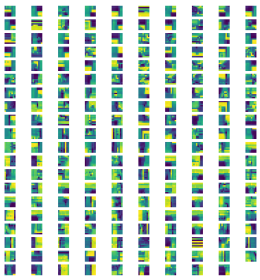
out a 40x40 patch surrounding each key point.We then resized the patch to 8x8 and used Gaussian blur on it.The resulting 8x8 patch was transformed to a 64x1 vector by removing bias and illumination, which are feature vectors that store information about the key point.The graphic depicts the feature vectors for the main points. The key conclusion from the feature descriptor process is the need of standardisation. If we had eliminated the standardization and performed the feature matching mentioned in the next phase, the unnormalized patch would have not fit well with the feature descriptors from the other photos. The explanation is because there are disparities in contrast and intensity. The standardization makes the patch insensitive to these intensity discrepancies, making it more "comparable".

D. Feature Matching

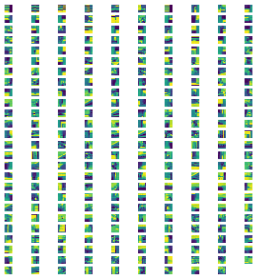
Feature matching uses the sum of square distance to determine the feature correlation between two photos. To find the best matches, we calculate the ratio of the lowest first to lowest second and compare it to the threshold. The threshold can be adjusted depending on the feature mapping's acceptance level.



(a) Features of Image 1



(b) Features of Image 2



(c) Features of Image 3

Fig. 6. Set 1

Now that we have a feature descriptor for each corner point, we can determine which points match between two photos. For each corner point in image 1, we calculated the sum of the square differences between all points in image 2. The best match was identified as the point with the shortest distance, followed by the second best match with the second shortest distance. If the ratio of the lowest distance to the second lowest distance is less than a certain number, we accept the matched pair; otherwise, we reject it.

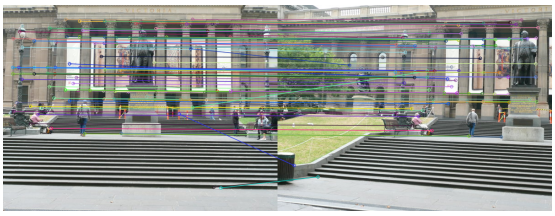
E. Random Sampling and Consensus

We are aware that some of the matches we obtain—referred to as outliers—are not significant. Stitching issues will arise if these outliers aren't eliminated. The Homography matrix is also computed when outliers are removed. The process of transforming two images is called homography. In two dimensions, homography comprises translational and rotational components. Consequently, the homography matrix of the inliers for both match pairs is eventually found. Until we have high-quality inliers from the match pairs, we continue iterating.

i *Outliers Rejection:* To calculate the predicted match with the goal match and threshold the outliers, we first compute the Homography matrix and multiply its inverse with the picture to obtain the second image reference to the first image.

ii *Compute Homography:* After initially rejecting the inliners based on a threshold, we compute Homography on the inliners once again. Using image 1, we pass this homography matrix to obtain the warp perspective.

(a) Matches between image 1 and 2



(b) Matches between image 2 and 3

(c) Matches between image 1 and 3

Fig. 7. Set 1

(a) RANSAC Matches between image 1 and 2

(b) RANSAC Matches between image 2 and 3

(c) RANSAC Matches between image 1 and 3

Fig. 8. Set 1

Fig. 9. Blended Image of Set1

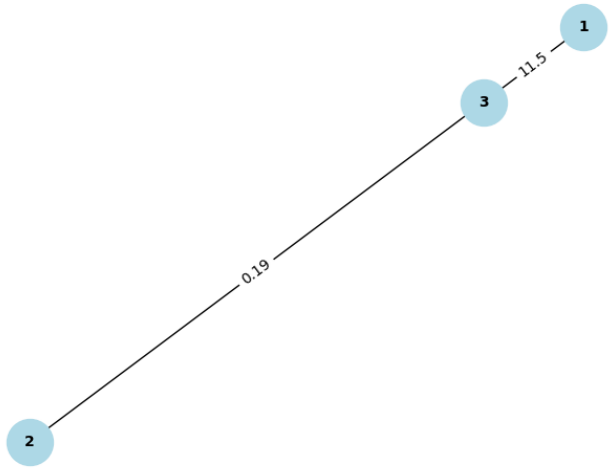


Fig. 10. Graph of blend

F. Warping & Blending

In warping, we simply multiply the inverse of the homography matrix by image 2 to compute the warp perspective with respect to image 1. By determining the xmin, xmax, and ymin, ymax of the freshly created image, we can create the container. The freshly created image will be kept in the container. Here, we did not do any blending as using alpha blending interfered with the corner detection. This caused issues where we needed to stitch multiple photos as corners where not detected correctly.

G. Future Scope

Currently, we are finding and applying warping successively which needs multiple passes over the warped images. After every warp, the image features are distorted more and more. A solution to this is calculating the homographies across every pair like now, but also store the relative homographies. We can

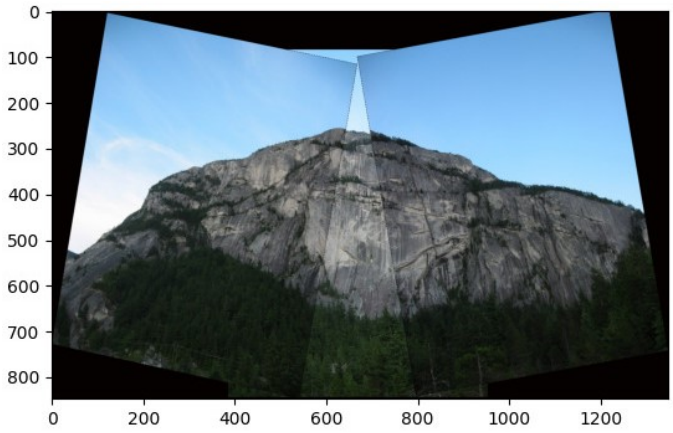


Fig. 11. Blended Image of Set2

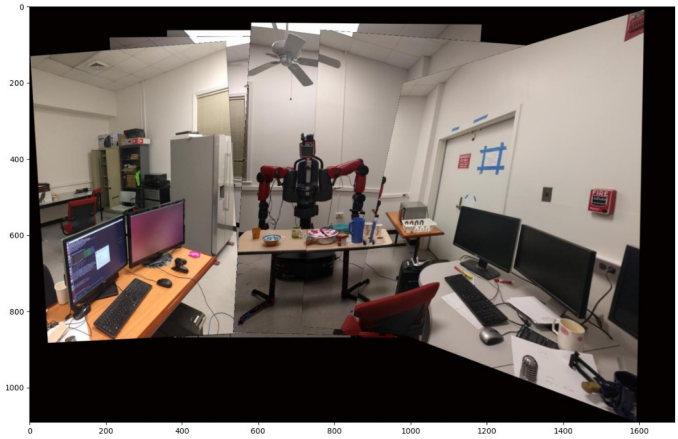


Fig. 12. Blended Image of Set3

then use Matrix multiplication to calculate the transformation from the tree centroid of graph to every other image. This would mean that each image will only undergo one warp which will result in minimal warping, hypothetically. Another improvement would be the use of Laplacian Pyramids to blend the images together as well as matching luminosity across images.

H. Final Analysis

This section's panorama was created using a conventional method that matched the photos' shared features. The first and most crucial step in any panoramic stitching scene is feature detection. Better features like SIFT, FAST, and ORB that offset problems with illumination, rotation, and scale may have been utilized. When combined with ANMS, our Harris Corner detector produced good results. Second, we think RANSAC is more effective because it requires manual adjustment of a large number of parameters. Certain parameter combinations function with one but not with the others. Additionally, the blending was not as flawless as we had hoped. There are numerous excellent third-party blending features with superior outcomes, such as particle blending.

Fig. 13. Blended Test Set1

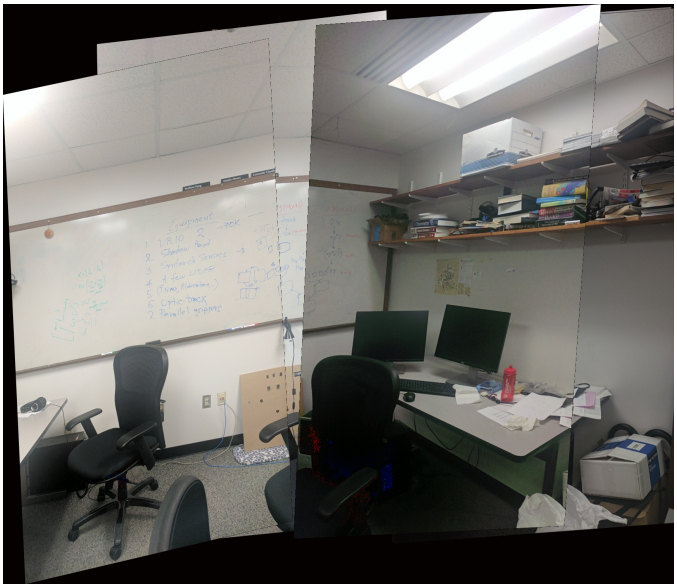


Fig. 15. Blended Test Set2

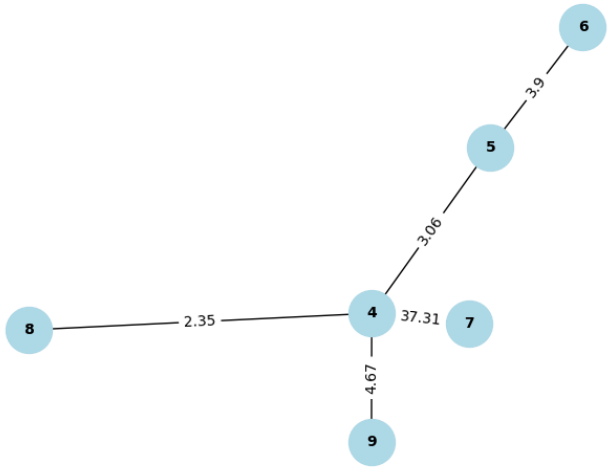


Fig. 16. Graph of blend links for Set2

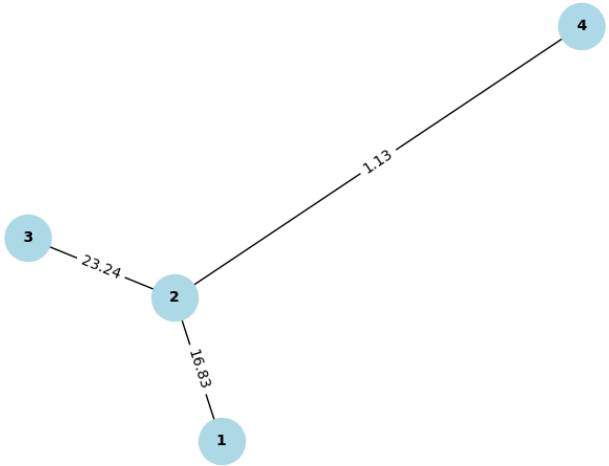


Fig. 14. Graph of blend links for Set1

If there are sufficient common traits among the photographs, this method works well; if not, we must remove the image from the panorama.We had to include translation in addition to homography when the homography matrix turned out to be negative in certain instances in order to preserve the pixels.We attempted splitting the photographs into two halves and applying blending and stitching to them because the stitching breaks down in a cyclic fashion when there are more than four images. However, we found that if there are a lot of photos to stitch together, it also becomes problematic because the resulting stitched image gets quite enormous, consuming a lot of processing power and time to create the panoramic. Additionally, we discovered that in order to eliminate the outliers, we had to adjust the RANSAC settings, such as the threshold value and number of iterations. However,

Fig. 17. Blended Test Set3

Fig. 19. Blended Test Set4

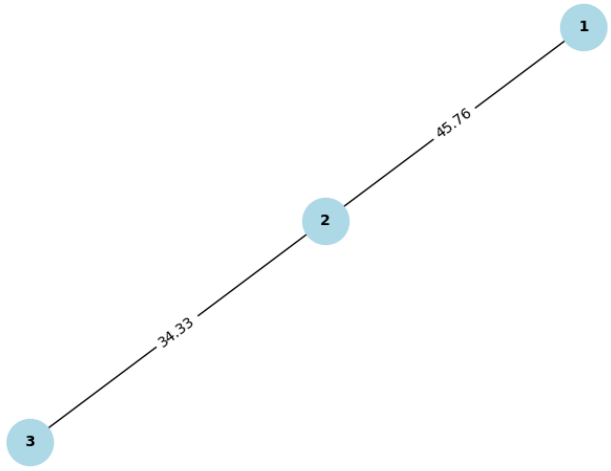


Fig. 18. Graph of blend links for Set3

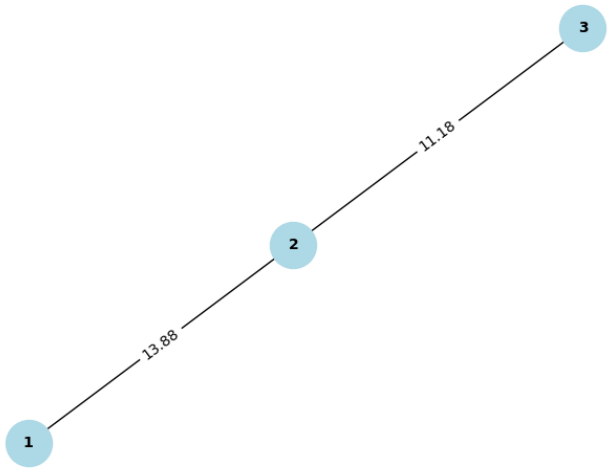


Fig. 20. Graph of blend links for Set4

occasionally, this process resulted in very few inliers; we need to resolve this problem so that the algorithm would function consistently across all photos without requiring modification.

II. PHASE 2: DEEP LEARNING APPROACH

In this section, we go over how the homography and stitching of the photos are affected by the latest deep learning algorithms. A reconstructed version of homography is made by the deep learning model. The primary benefit of this strategy is that, provided the basic network can be generably generated, the algorithm can be strengthened. CNN is used for both supervised and unsupervised training in order to estimate the homography between picture pairings. After learning the patches' corners, this model attempts to predict the warp of the same patch that is perturbed as the label set. This method is known as four-point parameterization.

A. Data Generation

The synthetic data utilized in the unsupervised and unsupervised techniques was created using the 5000 photos in the MSCOCO dataset. Each image has an active region defined, as per the problem statement. This is the area that we have selected to be the center of the image, with the exception of 100 pixels on each of the four sides. Next, a 128x128 patch was selected inside the active zone, we call it P_A. We selected perturbations for each of the patch's four corners

within the interval [-32, 32]. The homography was computed using the original corners and the perturbed corners. After warping the original image using inverse homography, a patch was cropped from the distorted image at the same spot as the original image's corners, we call it P_B. In this manner, we collect several pairs of data from a single image based on the original image size. In this approach, we have approximately 50000 image pairs from the initial 5000 (10 patches from each image). With the corner correspondences, we also produce the 4 point homography H_{4pt} between P_A and P_B, which is given by the equation:

$$H_{4pt} = C_A - C_B \tag{1}$$

where C_A represents the corners of patch A and C_B the corners of patch B.

Our deep networks' input is made up of the patch-pairs P_A and P_B, which have been transformed to grayscale, single-channel images and stacked together. The labels for the supervised approach are assumed to be H_{4pt}. For the complete set of image data, this procedure is repeated.

B. Supervised Model

Since this model has patches A and B, we may compute the ground truth, or H4pt matrix. We calculate the L2 loss by comparing this to the anticipated H4pt matrix. This network design is an adaptation of the VGGNet framework.

I. **Architecture:** This network is architecturally comparable to Oxford's VGG Net, using 3x3 convolutional blocks with Batch-Normalization and ReLU activations. The network consists of eight convolutional layers, with a max pooling layer that occurs after every two convolutions, with pool size = 2 × 2 and stride = 2. Each of the next four convolutional layers uses 128 filters, while the first four use 64 filters each. Two fully-connected Dense layers come after these convolutional layers. To prevent overfitting, dropout with a probability of 0.5 is performed after the first fully-connected layer and the final convolutional layer. There are 1024 units in the first Dense layer and 8 units in the final Dense layer.

II. **Training Parameters:** We trained our model for 16 epochs, with a batch size of 64, using the Adam optimizer with a learning rate of $1e^{-3}$. As recommended by the official implementation, we compute the L2 Loss for our training performance metric.

$$L_2 loss = \|\tilde{H_{4Pt}} - H_{4Pt}\|_2 \tag{2}$$

C. Unsupervised Model

The Tensor-DLT approach was implemented in the supervised model, as recommended in the work "Unsupervised Deep Homography: A Fast and Robust Homography Estimation Model," because the supervised model is more biased and so has more room for development. This produces a homogeneous matrix from the expected H4pt. Additionally, a

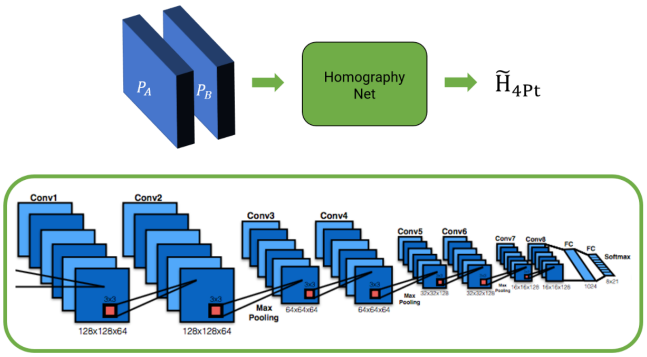


Fig. 21. Homography Net Architecture

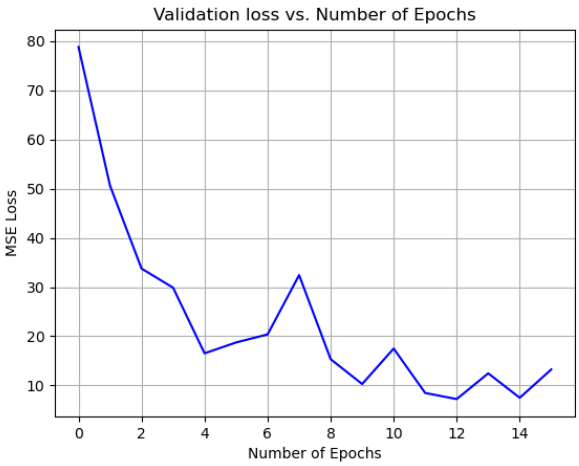


Fig. 22. Supervised Validation Loss vs Epochs

spatial transformation layer is used to compute patch B from the given data after which the L1 loss, or photometric loss is calculated between the data generated patch B and the output. In order to optimize the weights, we backpropagate the loss. Unlike the supervised model, this one is not biased.

I. **Architecture:** The architecture of the Unsupervised model consists of three parts:

a) *Homography Net:* This section of our network comprises of the network we constructed for our supervised network, which has two dense layers, three max pooling layers, eight convolution layers, and their corresponding batch normalization and RELU layers.

b) *Tensor Direct Linear Transform:* To compute a differentiable mapping from the 4-point parameterization H_{4pt} to H, the 3×3 parameterization of homography, we create a Tensor Direct Linear Transform layer. This layer remains differentiable to enable backpropagation during training, basically applying the DLT algorithm to tensors. The respective corners from the original patch CA and the 4-point parameterization H_{4pt} are the inputs of this function and the output is an estimate of the 3×3 homography parameterization H.

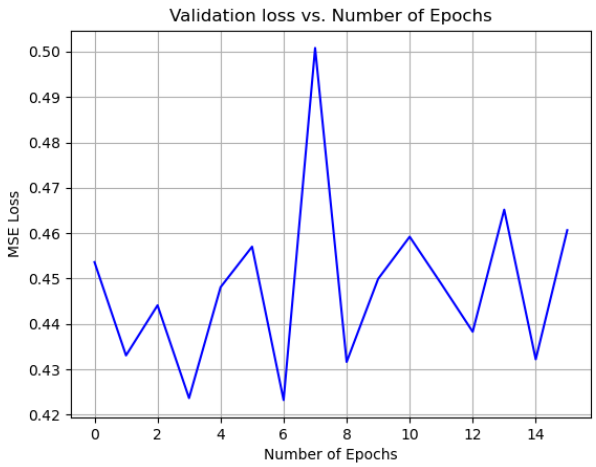


Fig. 23. Unupervised Validation Loss vs Epochs

c) *Spatial Transformer Network:* To obtain warped patch, the subsequent layer applies the 3 × 3 homography estimate H, which is produced by the Tensor DLT, to the pixel coordinates of patch A. In order to train our neural network, these warped coordinates are required in order to compute the photometric loss function. This layer has to be differentiable in addition to warping the coordinates in order for backpropagation to allow the erroneous gradients to pass through. Thus, we use the Spatial Transformer Layer to homography transformations.

II. **Training Parameters:** We employed a training configuration similar to the supervised model, utilizing the Adam optimizer, learning at a rate of $1e^{-3}$ for 100 epochs, with a batch size of 64 and employing photometric loss function to track the unsupervised model's training efficacy.

$$\text{Loss}_{\text{photometric}} = \|warp(\text{I}_A, \text{H}_{4pt}) \cdot \text{I}_B\|_1 \qquad (3)$$

D. Conclusion

In conclusion, our implementation employs two distinct networks: one operates in a supervised manner, yielding a promising EPE (End Point Error) loss of 4.23, while the other employs an unsupervised approach with a slightly higher EPE loss of 11.56. The supervised model excels with labeled training data, showcasing accurate correspondence estimation and leading to a more precise panorama reconstruction. In contrast, the unsupervised model relies on self-imposed constraints and learned representations, presenting a valuable alternative when labeled data is scarce.

Model	**EPE loss**
Supervised Model	4.23
Unsupervised Model	11.56

TABLE I
COMPARISON OF EPE LOSS BETWEEN DIFFERENT MODELS

(a) Test Set 1

(b) Test Set 2

(c) Test Set 3

Fig. 26. Warped Test Sets from our Network

Despite performance differences, both models contribute to panorama generation understanding. The supervised model shines in scenarios with abundant labeled data, while the unsupervised model highlights the potential of learning from unannotated data. This duality allows flexibility in adapting the approach to real-world scenarios, shedding light on the trade-offs between supervision and unsupervision in panoramic image synthesis. Further research and experimentation could refine these models, potentially bridging the performance gap and enhancing the robustness of panoramic image generation systems.

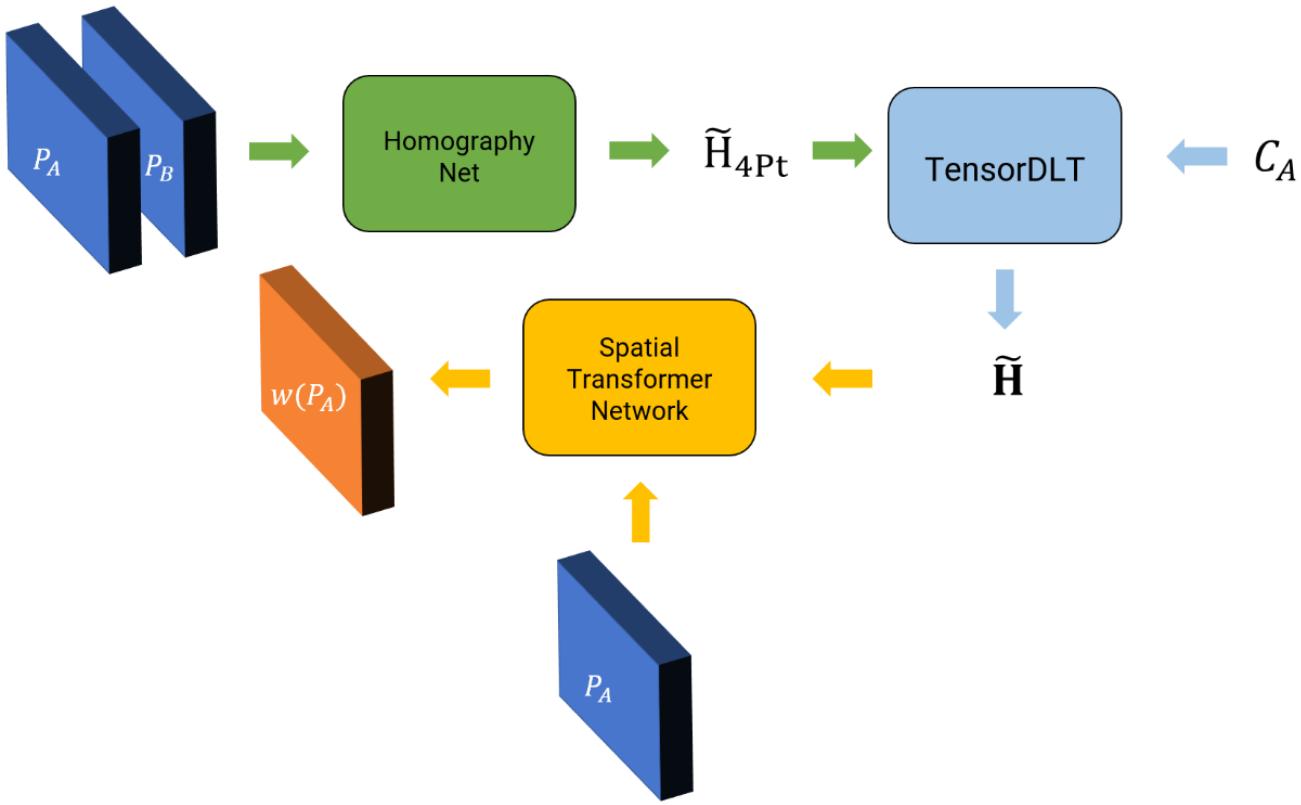


Fig. 24. Unsupervised Network Architecture

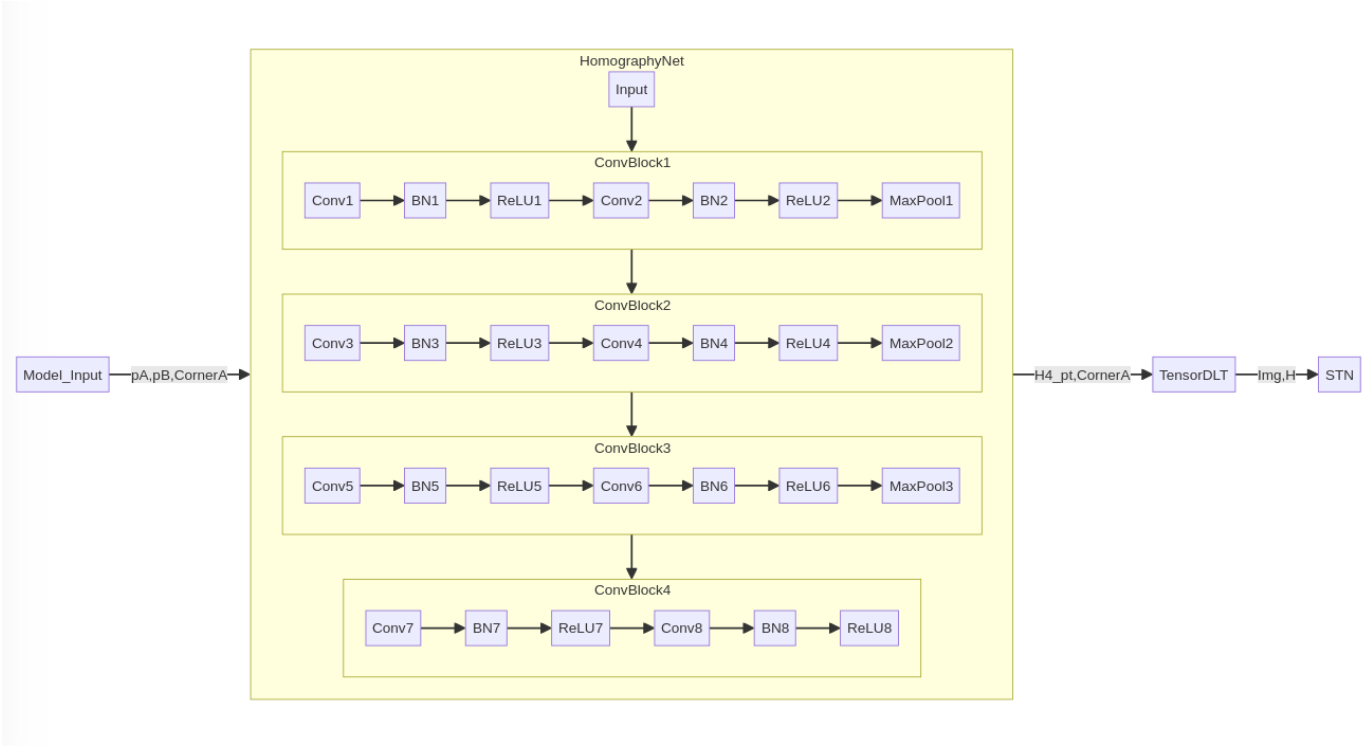


Fig. 25. Unsupervised Network Model