CMSC 733: Project 1 - My AutoPano

Hetansh Patel
Masters of Engineering in Robotics
Universty of Maryland, College Park
Email: hpatel57@umd.edu

Aditya Vaishampayan
Masters of Engineering in Robotics
Universty of Maryland, College Park
Email: adityav@terpmail.umd.edu

I. INTRODUCTION

In this paper we will solve the problem of stitching two or more images to create one seamless panoramic image. The task central to creating panoramic images involves finding the Homography matrix. In this project, we implement various method to estimate the Homography matrix. Phase 1 discusses the traditional approach and phase 2 discusses the deep learning approaches, specifically Supervised and Unsupervised approach.

II. PHASE 1 : TRADITIONAL APPROACH

The objective of this phase is to solve the problem of panorama stitching using traditional approach. We describe the pipeline of the algorithm in detail in the following steps

A. Corner Detection

The first step of the method is to detect corners in the image. For this, we have used Harris corners detector(cv2.cornerHarris).The output of this operation will give all the corners in the image as shown in the image.

B. Adaptive Non-Maximal Suppression(ANMS) Algorithm

The output of Harris Corner detection will give us all the corners present in the image. However, not all the corners are required or are favourable for our goal of panorama stitching. We need corners whose intensity is high and are evenly spaced throughout the image. This is when ANMS algorithm comes into play. This algorithm takes "N-Best"(number of corners) as an input and gives N-Best corners as an output which are evenly spaced and have highest intensity in the descending order. The algorithm of ANMS is shown in the figure and the output of the ANMS algorithm is shown in the images.

C. Feature Descriptor

In the previous step, we found N-Best corners, also known as feature points, in an image. Now our goal is to find feature descriptors for these feature points. We can create our own feature descriptor by doing the following steps

1) Take a patch of size 4040 centered around the corner/feature point.

2)Now apply gaussian blur to that image patch.

3)Now, sub-sample the blurred output (this reduces the dimension) to 88. Then reshape to obtain a 641 vector.

4)Standardize the vector to have zero mean and variance of 1(Standardization is used to remove bias and to achieve some amount of illumination invariance).

This vector will be our Feature Descriptor.

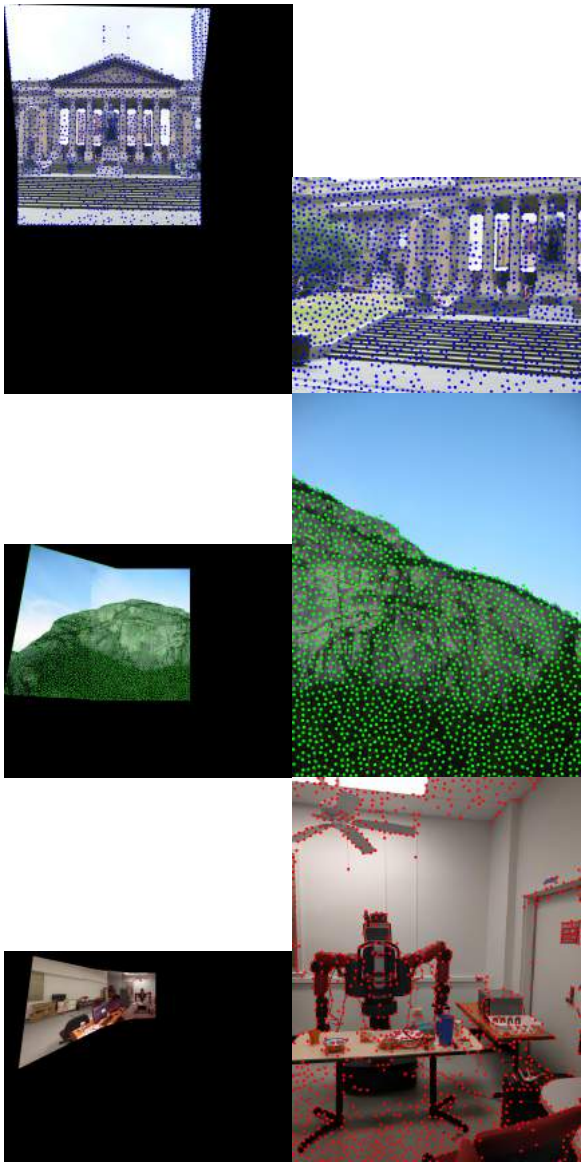

Fig. 1. Detected corner of images 1,2 and 3 from Train Set

Fig. 2. Detected corner of images 1,2 and 3 from Test Set

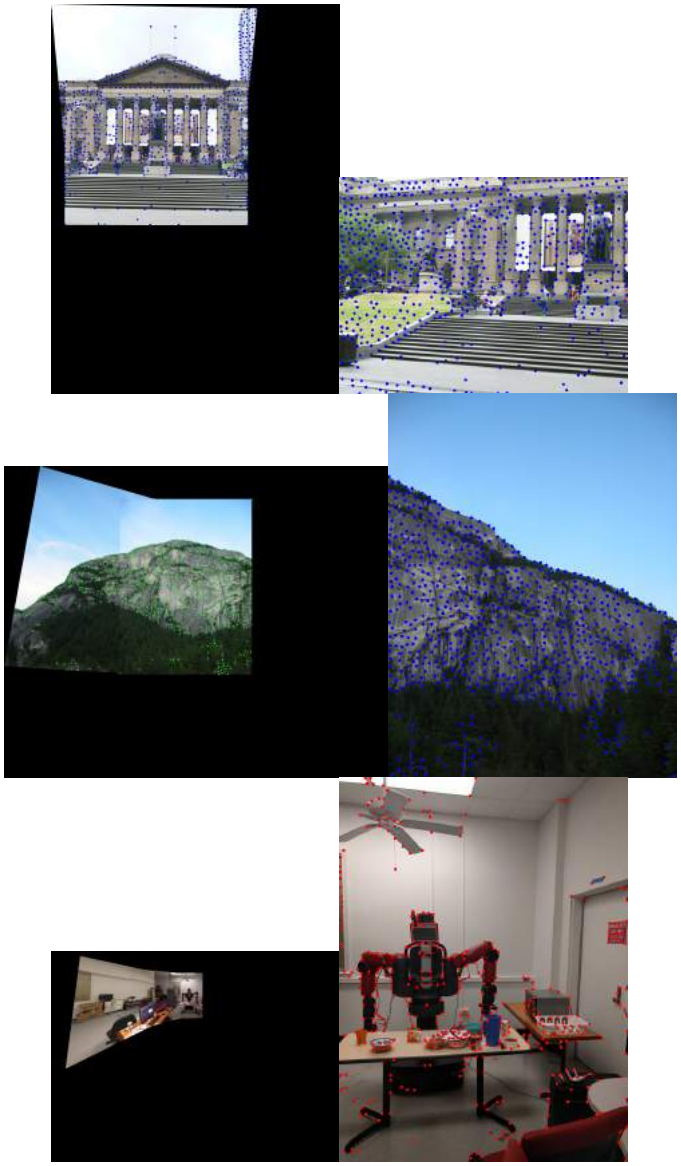

Fig. 3. Detected corner of images 1,2 and 3 from Train Set after ANMS

D. Feature Matching

In the previous step, we created a feature descriptor for each corner/keypoint by encoding the corner/feature point by 641 feature vector. Now, we want to match the feature correspondences between the 2 images. This can be done in following steps.

1)Pick a feature descriptor in image 1, compute sum of square differences between all the feature descriptors in image 2.

2)Find the feature descriptor in image 2 whose SSD with current feature descriptor in image 1 is minimum.

3)After computing SSD between all the feature descriptors in image 2, take the ratio of best match (lowest distance) to the second best match (second lowest distance). And, if this is below some ratio keep the matched pair or reject it.

4)Repeat this for all points in image 1. You will be left with only the confident feature correspondences and these points

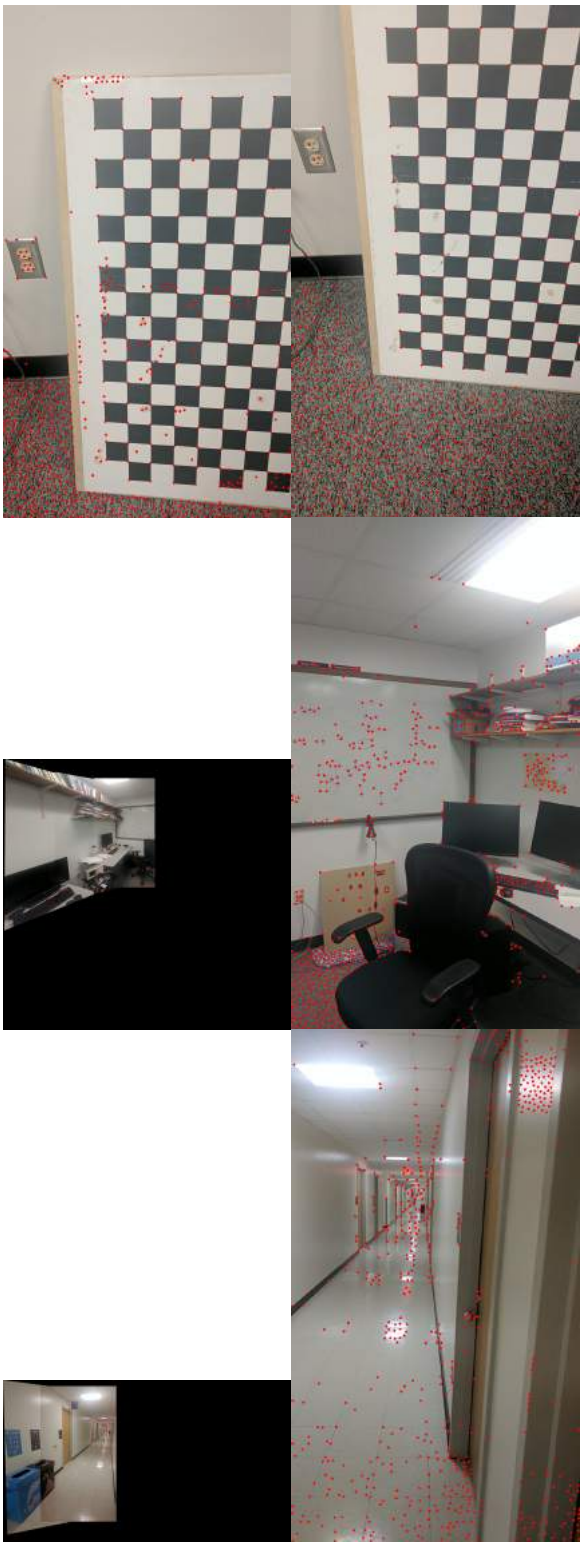

Fig. 4. Detected corner of images 1,2 and 3 from Test Set after ANMS

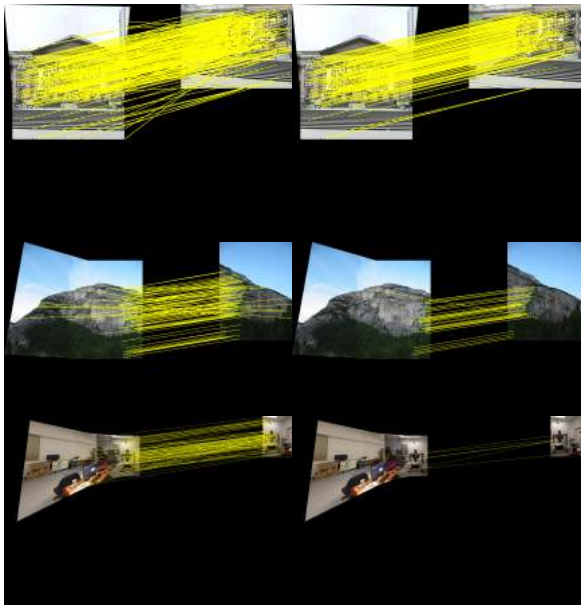

Fig. 5. Feature matching without and with ransac on images 1,2 and 3 from Train Set

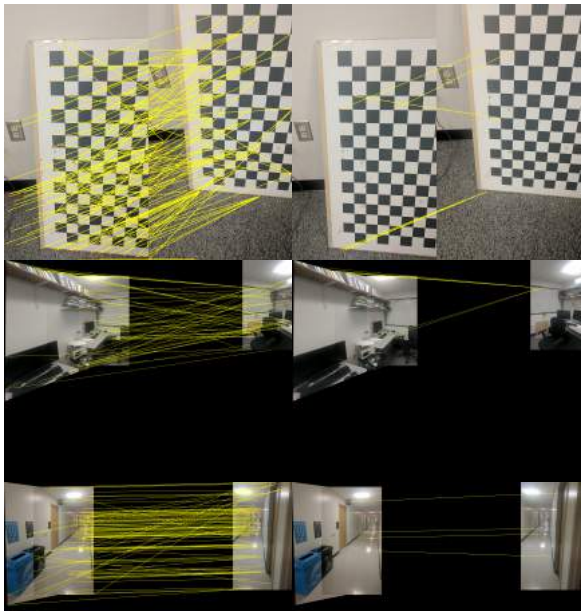

Fig. 6. Feature matching without and with ransac on images 1,2 and 3 from Test Set

will be used to estimate the Homography between two images.

E. Estimating Homography using RANSAC

We now have matched all the features correspondences. However, not all the matches are correct. Bad feature matching can lead to incorrect Homography and can seriously affect the panorama stitching process. Hence in order to decrease the amount of bad feature matching and correctly estimate Homography, we will estimate Homography via RANSAC. The process of estimating Homography via RANSAC is a followed:

Fig. 7. Panorama of image 1 from Train dataset

1)Select four random matched feature correspondences from image 1 and image 2.

2)Compute Homography matrix for these four random matches.

3)Let all points in image 2 be the target points. Multiply all the points in image 1 by Homography matrix. Let us call these points as predicted points.

4)Compute SSD between target points and predicted points and Compute number of inliners present. Inliners are those points whose SSD is less than some user chosen threshold.

5)Repeat the above-mentioned steps until you have found more than some percentage of inliers. Return the Homograpgy.

6)If number of inliners doesnt reach the threshold, then repeat steps (1-4) till you have exhausted number of iterations (specified by user) and return the corresponding Homography matrix which has the most number of inliners.

F. Stitching and Warping

Now that we have homography matrix, we can easily perform warping and stitching and ultimately create a panoramic picture. The procedure is being carried out in the following steps:

1)Lets say we have N (N=10) images to be stitched. Let the first image be the primary image and second image be the secondary image.

2)Compute the Homography between the two images.

3)Calculate the inverse of the above calculated Homography matrix and warp the secondary image using this inverted Homography matrix. However, the size of this warped image should be the sum of size of primary and secondary image.

4)Now add the primary image and newly warped secondary image such that leftmost corner of primary image is (0,0).

5)Now, consider the image created by adding primary image and warped secondary image as new primary image and 3rd image in the list to be new secondary image.

6) Repeat above steps until all the images are exhausted.

The output will be a panoramic image formed by stitching N number of images.

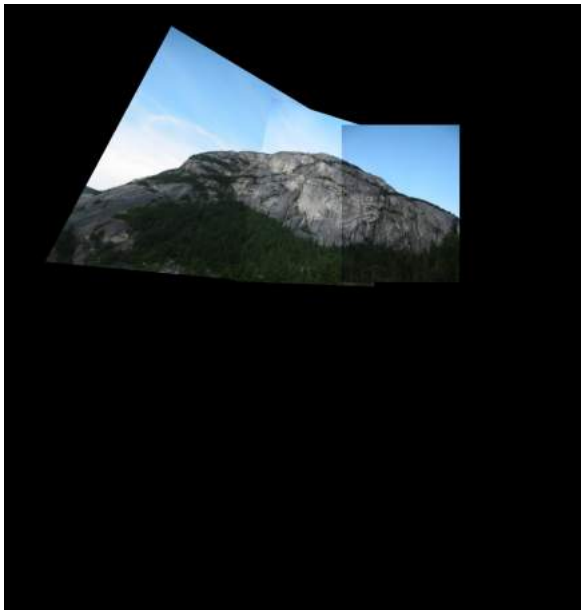

Fig. 8. Panorama of image 2 from Train dataset

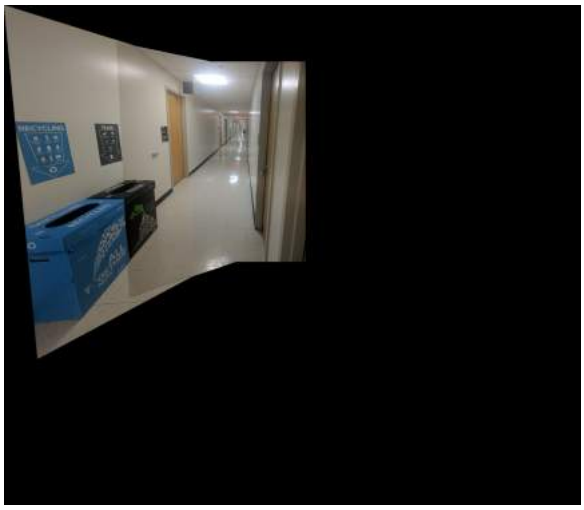

Fig. 9. Panorama of image 3 from Test dataset

G. Conclusions and Discussions: Phase 1

In this section we created a panorama using a traditional approach by matching features between two images. This approach gives satisfactory results which can still be improved using additional techniques like blending or averaging the pixel values of the two images at the same location while stitching. Also, the algorithm produces better results when all the image planes are at infinity. As seen in some cases the estimated homography is not very accurate. This happens because the both image planes are not at the same depth. In such cases better results can be produced by projecting these images on a cylinder and then performing the stitching.

When creating panoramic images by stitching more than two images it is possible that there exists two such images which has very less or zero overlap and hence stitching them can become tricky. In order to overcome this problem we

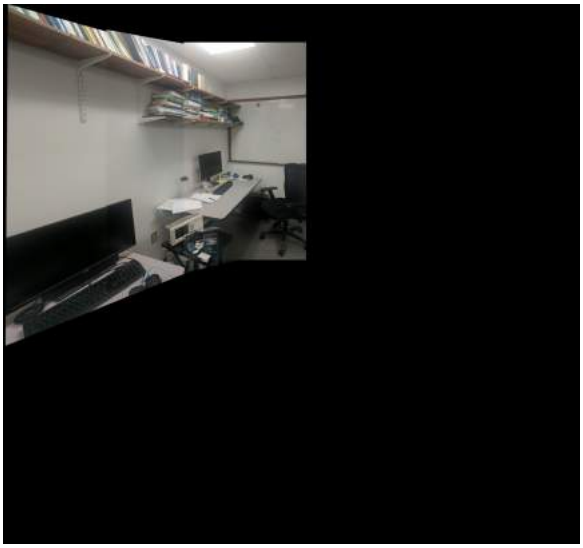

Fig. 10. Panorama of image 4 from Test dataset

set a threshold for minimum number of feature matches. If there are fewer feature matches than this threshold, skip the image and jump to the next image. Do the same till all the images are exhausted. After doing this we have a primary image with multiple images stitched. Now check for feature matches in new primary image and the ones which we skipped over. There are high chances that now the number of feature matches will be more than the threshold. Do this sequential steps of skipping the images if threshold not reached and then matching it with the new primary image till all the images are exhausted.

III. PHASE 2: DEEP LEARNING APPROACH

The objective of this phase is to implement the entire pipeline of traditional approach for panoramic stitching using Deep Neural Networks. Neural Networks takes images as its input and outputs a Homography matrix for those respective images. In this phase we will implement two different approaches: Supervised and Unsupervised Approaches. However, in order to implement these methods we, first, need to create synthetic data on which ou network will be trained. Hence, first, we will discuss about generating data followed by supervised and unsupervised approaches.

A. Generaion of Data

As its a case with any neural network, we need a lot of data to train our neural network. For training our netwrok, we create our synthetic dataset. Dataset of MSCOCO has been used to create our own synthetic dataset. The convolution doesn't accept images of arbitrary sizes. Hence, we first need to create images of the dimension 128x128. This is done by taking patches of dimension 128x128 from the images of dataset. We create two different patches, namely patch 1 and 2, from the same image patch. In order to create two different images from the same patch, the patch 2 is created by moving the vertices of the patch 1 by small pertubation ranging from [-p,p](user decided). This perturbation is saved as ground truth

Fig. 11. Patch1 marked in red and Patch 2 marked in blue before warping

Fig. 12. Patch1 marked in red and Patch 2 marked in blue after warping

for our supervised model. The perturbed patch is not a perfect square of dimension 128x128(dimensions accepted by our network) and thus, to get a perfect square patch, we inverse warp the original image using the homography matrix. This homography matrix is computed between corner point of patch 1 and corner points of patch 2.

B. Supervised Approach

In the paper the task of computing homography is formulated as a classification problem and regression problem and here we have implemented the regression network. The only difference is that in the classification we discretize the output into 8 points and 21 bins (8x21 output) while in the regression one the output is 4 point parameterization of homography .

Both networks take as input a two-channel gray scale image sized 128x128x2. In other words, the two input images, which are related by a homography, are stacked channel-wise and fed into the network. We use 8 convolutional layers with a max pooling layer after every two convolutions. The convolutional layers are followed by two fully connected layers. The first fully connected layer has 1024 units.The regression network directly produces 8 real-valued numbers as shown in figure and uses the Euclidean (L2) loss as the final layer during training. [2]

Approximately 40000 images were generated for the training of the network from train set of 5000 images of the

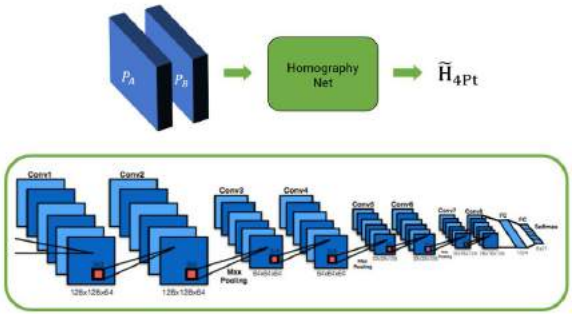

Fig. 13. Architecture of Supervised HomographyNet

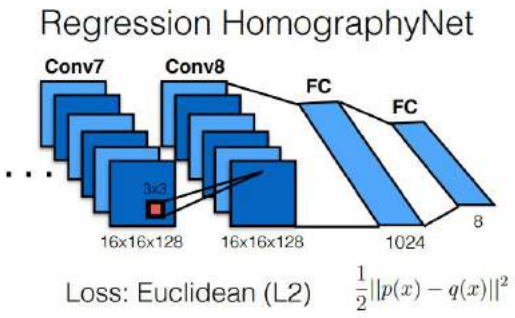

Fig. 14. Architecture of Regression HomographyNet

MSCOCO dataset. Owing to the nature of HomographyNet infinite number of images can be generated from a finite set of train images. The perturbation values were set to random values between -16 and +16.

C. Results for Supervised Approach

In our network we use Adam optimizer as the optimizer with a learning rate of 0.0001 with a batch size of 128. We trained the network for 80 epochs.

The final graph obtained with the updated weights takes an average of 0.69 seconds to produce a 4-point parametrized output between two images that are related by a homography matrix.

For getting the homography matrix between the images from the 4-point output of the network we take the corner locations of the original image and then add the 4-point output to this to get the corner locations in the second image. Now that we have corresponding corner locations in the images respectively, we calculate the homography between them.

D. Unsupervised Approach

Unlike the supervised approach, the unsupervised approach computes the homography between a pair of images without using explicit labels for the 4-point homography description. The network used for this approach consists of 4 parts as described below:

1) Regression Homography Model: This is exactly the same network as was used in the supervised homography

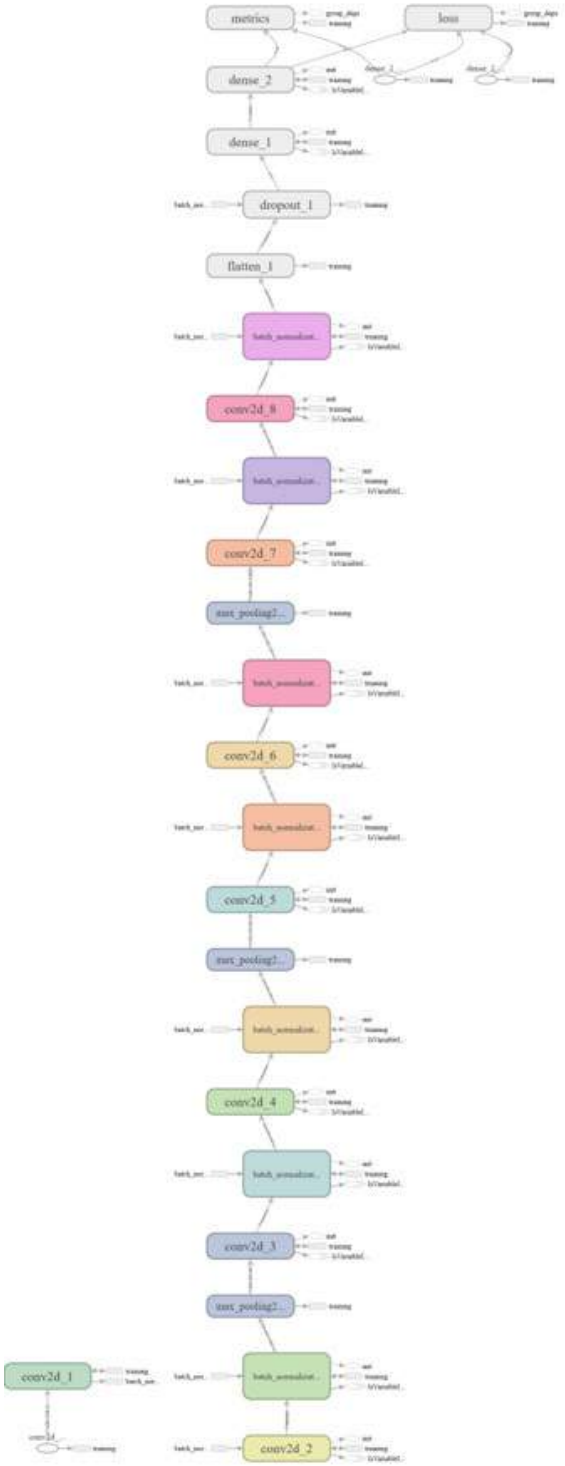

Fig. 15. Architecture of HomographyNet

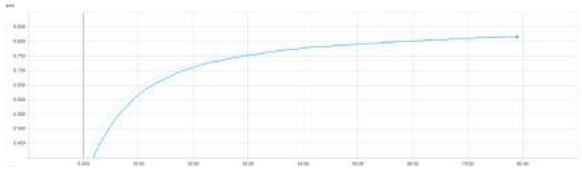

Fig. 16. Accuracy vs Epochs for Supervised Approach

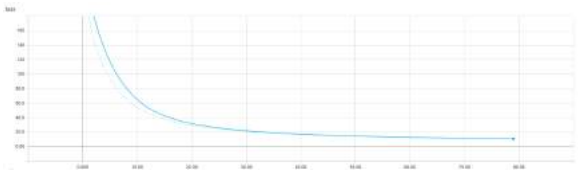

Fig. 17. Loss vs Epochs for Supervised Approach

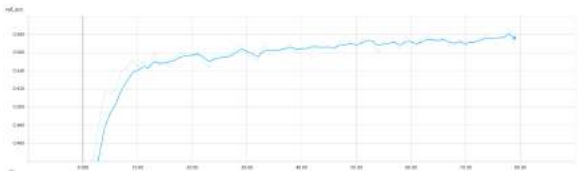

Fig. 18. Valiation accuracy vs Epochs for Supervised Approach

detection approach. This network takes as input, 2 patches extracted from the same image and one of them is warped with certain homography. This model learns and predicts the homography with which the two patches are related, in the 4 point formulation. This 4 point estimate is then added to the 4 corners of the original patch to get an estimate of the 4 corners of the warped patch.

2) TensorDLT Network: This network takes as input the 4 corners of both the original and the warped patches. It then formulates a system of linear equations and solves it using singular value decomposition in a differentiable way so that errors can be backpropagated through the network. This gives us the (3x3) homography matrix. The network in a way, learns the functionality of the opencv cv2.getPerspectiveTransform() function in a differentiable way.

3) Spatial Transformer Network: This network takes as input, the homography matrix predicted by the TensorDLT network and the original image from which the original patch was extracted. It then learns the application of the homography on the original image to get the warped patch. The network in a way, learns the functionality of the opencv cv2.warpPerspective() function in a differentiable way.

4) Calculating Photometric loss: The warped patch extracted from the Spatial Transformer Network is then compared with the ground truth warped patch using the L1 photometric loss. This loss is backpropagated through the network.

E. Conclusions and Discussions: Phase 2

Although the supervised approach performs well in both validation and test set, some results in supervised section suffered from minor error due to changes in illumination of the patches. Also when the test was performed on patches that were perturbed more than the perturbation range on which the network was trained on, the error in ground truth and predicted homography was large(which is quite obvious).

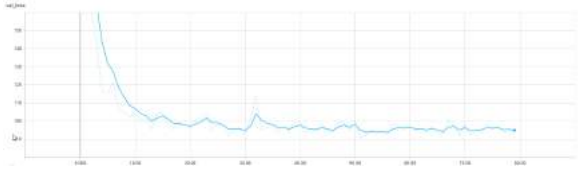

Fig. 19. Validation Loss vs Epochs for Supervised Approach

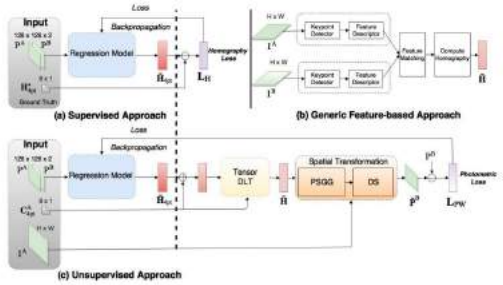

Fig. 20. Overview of homography estimation methods; (a) supervised deep learning approach; (b) Feature-based methods; and (c) unsupervised method

While the supervised deep learning method has promising results, it is limited in real world applications since it requires ground truth labels. Unsupervised approach, on the other hand, achieves faster inference speed, while maintaining comparable or better accuracy and robustness to illumination variation. In addition, the unsupervised method has superior adaptability and performance compared to the corresponding supervised deep learning method.

We were unable to train the unsupervised network and there for do not have any results for it. The network did not become better over time which we thought indicated that it was not working. We gave it our best shot

REFERENCES

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

[2] Ty Nguyen, Steven W Chen, Shreyas S Shivakumar, Camillo Jose Taylor, and Vijay Kumar. Unsupervised deep homography: A fast and robust homography estimation model. IEEE Robotics and Automation Letters, 3(3):2346– 2353, 2018. .

[3] Iuri Frosio Hang Zhao, Orazio Gallo and Jan Kautz. Loss functions for neural networks for image processing. CoRR, abs/1511.08861, 2015.

[4] https://github.com/tynguyen/unsupervisedDeepHomographyRAL2018/blob/402248e4e49

[5] https://github.com/Pratiquea