Project 1: MyAutoPano

Krunal M. Bhatt
Masters of Science in Robotics Engineering
Worcester Polytechnic Institute
Worcester, Massachusetts 01609
Email: kmbhatt@wpi.edu

Using 1 Late Day

Jesulona Akinyele
Masters of Science in Robotics Engineering
Worcester Polytechnic Institute
Worcester, Massachusetts 01609
Email: jfakinyele@wpi.edu

Using 1 Late Day

***Abstract*—The report presents our understanding and experimentation with stitching two or more images to create one seamless panorama image. As asked, each set of custom images has some repeated local features. Phase 1 shows the basic traditional approach to solving the problem. Phase 2 focuses on the deep learning approach and we implement 2 approaches to estimate the homography between two images. We use the MSCOCO dataset for the HomographyNet. Both phases show their respective analysis, outputs, and graphs.**

I. Phase 1: Traditional Approach

Phase 1 shows a traditional approach as to how panorama images are created. It includes various sub-components like Detecting corners, Adaptive non-maximal impressions, and extract features. The image Fig.1 shows the same.

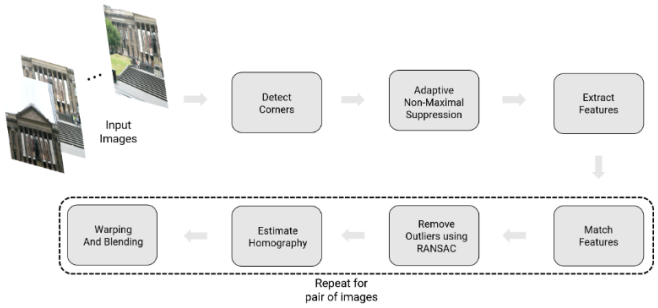

Fig. 1. Overview of Image stitching

We will now look at how the output looks after each of the rudimentary steps is performed.

A. Corner Detection:

Corners are regions with large variations in intensity in all directions. We use a Harris Corner Detector that finds differences in intensity for a displacement (u, v) in all directions. This is expressed as below:

$$E(u,v) = \sum_{x,y} w(x,y)\ [I(x+u, y+v) - I(x,y)]^2$$

The equation has a window function w that is either a Gaussian window or a rectangular window which gives weights to pixels.

We have to maximize the equation above and to do that we maximize the 2^{nd} term by using Taylor series expansion. Then a score is calculated based on the following equation which decides whether or not a region in the image is a corner.

$$R = det(M) - k(trace(M))^2$$

M is a matrix containing the image derivatives I_x and I_y in the x and y directions. The eigenvalues of this matrix decide whether the region is a corner or not. We directly use a function cv2.cornerHarris which does majorly most of the calculation shown above. Fig. 2 accounts for the same. Fig. 3 shows the detected corners.

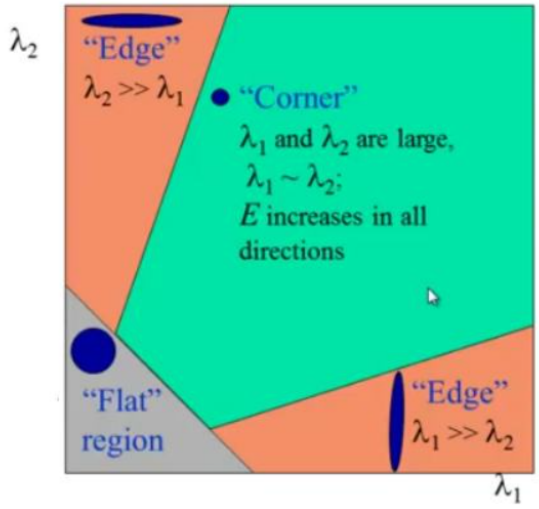

Fig. 2. Criteria for Corner definition

B. Adaptive Non-maximal Suppression (ANMS):

In this step in the process, as shown in Fig. 3, we use the corner score image and process it further. The corner image has corners unevenly distributed. Hence, we use the Adaptive Non-Maximal Suppression(ANMS) function to make them evenly distributed. ANMS will try to find the corners which are local maxima. Fig. 4 shows ANMS function output.

C. Feature Descriptor:

Feature Descriptor is generated after we get the corner points. We use this descriptor to describe the feature for each

Fig. 3. Detected Corners

Fig. 4. ANMS Output

point. To obtain the descriptor, a patch of 41 x 41 centered at each point is used. This patch is then blurred and sub sampled to 8 x 8. We then one hot encode this patch to a size of 64 x 1 vector. Fig 5. Shows a visualization of the patches being selected.

D. Feature Matching:

After getting a feature descriptor for each corner in the image, we find point matches between the two images. We calculate the sum of squares of differences between all the points in image 2, from a corner point in image 1.

Moving further, we found the best match as the point with the lowest distance and second best with the second lowest distance. If the ratio of this 1^{st} lowest and 2^{nd} lowest distance is less than a particular value we accept the pair. Fig. 6 shows the output.

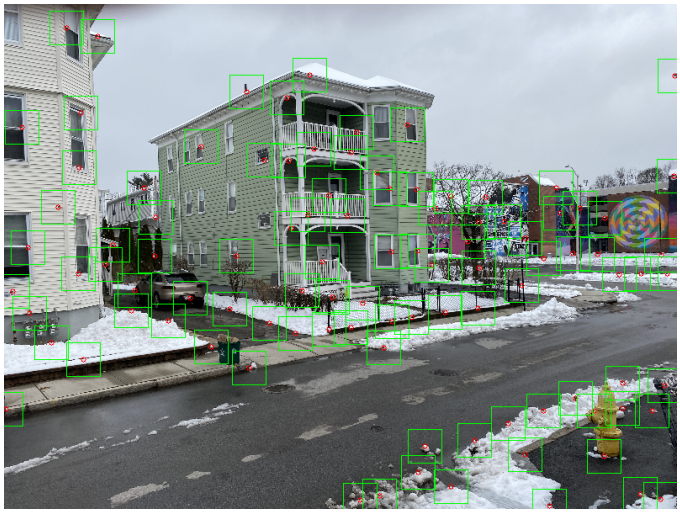

Fig. 5. Patches

Fig. 6. Matched Features

E. RANSAC for outlier rejection and Robust Homography:

Since we have matched all the features, they need not be necessarily true. To remove the faulty matches, we use Random Sample Consensus or RANSAC to compute the Homography. We can see in Fig. 6 that there are some wrong-matched pairs. Fig. 7 shows the filtered output.

Fig. 7. RANSAC Output

F. Blending Images:

We use a simple overlapping method for blending two images. Using the homography matrix, we transformed image 1 on the plane of image 2. Fig. 8 shows the issue in the output while blending the image. Although the approach described was implimented, we encountered some issues as described below.

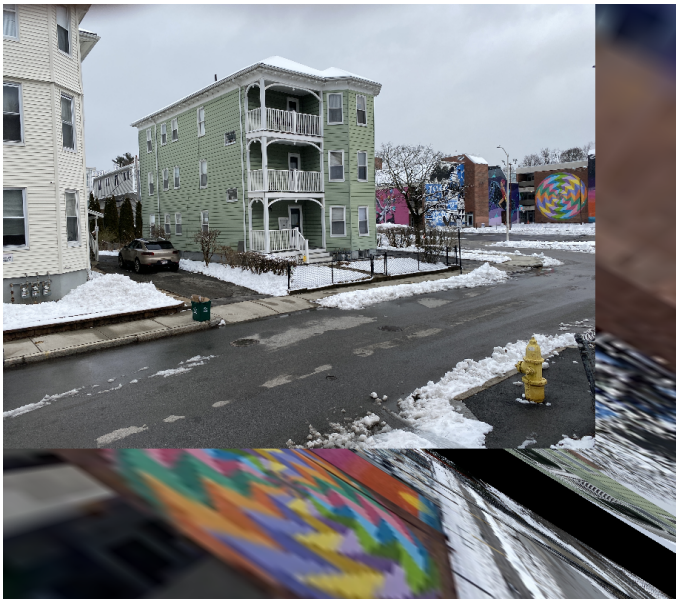

Fig. 8. Blended Images

G. Issues:

In practice, the feature matching proved to be less effective than anticipated. The keypoints, despite being logically placed and distributed across the images, did not translate into robust and distinctive descriptors necessary for accurate matching. This led to a cascade of issues in subsequent stages.

The inaccuracies in the homography matrix, stemming from the weak feature matching, resulted in warped images that did not align correctly. This misalignment was a significant roadblock, as precise image alignment is crucial for seamless blending.

H. Analysis and learnings:

Given these challenges, our approach to blending, particularly in scenarios involving more than three images, was severely hampered. The compounded effect of these challenges highlighted the critical importance of robust feature detection and matching in the panorama stitching process, aspects that require further refinement for successful implementation.

II. Phase 2: Deep Learning Approach

Phase 1 shows a traditional approach to creating panorama images, while Phase 2 shows the modern deep learning approach to the manual method. A supervised and an unsupervised learning approach is used to estimate the homography between two images.

The networks use an input of 128 x 128 image patch from both the images stacked by depth resulting in a image pair shape of 128 x 128 x 2. we use MSCOCO dataset to generate synthetic training or testing examples by applying random projective transformations on natural images.

A. Data Generation

For the data generation part of the project, a data generator script generates patch A, and patch B and calculates the tomography matrix H for the supervised learning part. The output for the same is shown in Fig. 9

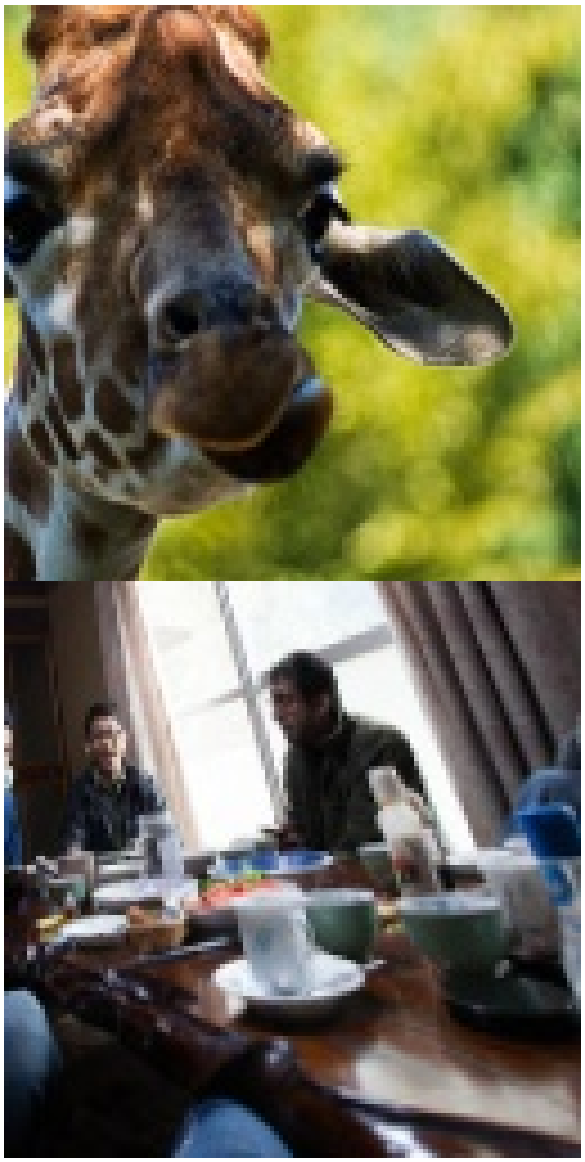

Fig. 9. Generated patches

The patches are generated for the train, test, and val data set provided to us. Now we will look at the the approach taken to implement the Homography Net with learning-based approaches.

B. Supervised Learning Approach

1) Architecture: Traditionally, Homography Net gained both homography estimation and feature extraction at the same time using a purely convolutional design. Although useful, this may confuse situations involving massive occlusions or repeating textures. Supervised learning techniques provide

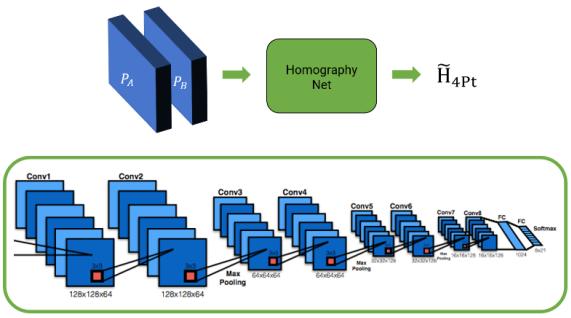

Fig. 10. Supervised Learning

important ground truth assistance to overcome these obstacles and improve the accuracy of the network.

One common approach with the network is training it using labeled data. This data includes image pairs with corresponding homography matrices pre-calculated in the data generation step. We optimize the network directly to predict the known homographies and make the learning process more targeted.

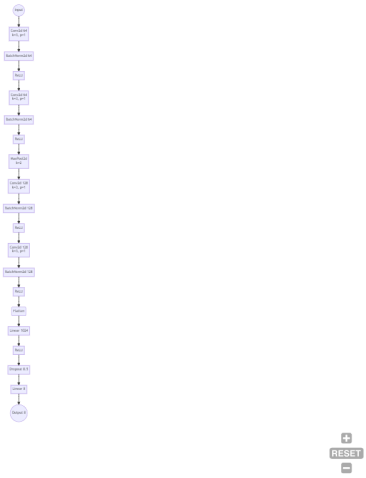

Fig. 11. Supervised pipeline

2) Training Parameters: For this homography model, we use an AdamW optimizer and a learning rate of $1e^{-4}$. Additionally, we have used a minibatch size of 32 and have trained the model for 30 epochs

C. Results

The Fig. 12 shows validation over epochs and Table I

Approach	Train Data	Test Data	Val Data
Supervised Learning	4.2	8.9	8.6

TABLE I
EPE IN PIXELS

III. Unsupervised Learning Approach

A. Architecture

The unsupervised learning model for homography estimation employs a deep convolutional neural network, structurally

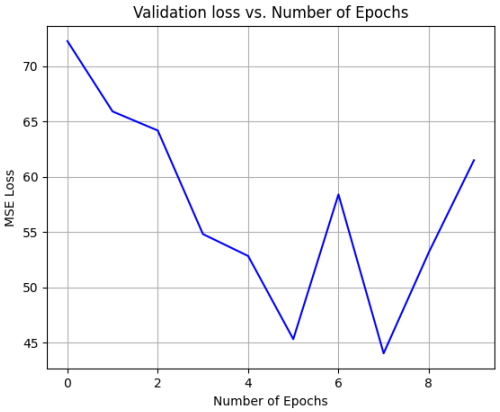

Fig. 12. Supervised Architecture

inspired by the VGGNet architecture. This can be seen in Fig. 13 The model's input consists of 2-channel images, each channel representing one of the image pair used in homography estimation. The network outputs a 4-point parameterization estimate of the homography transformation, subsequently utilized in a differentiable warping process.

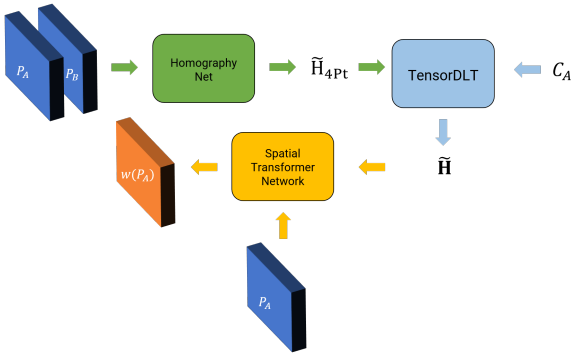

Fig. 13. Unsupervised Learning Architecture

B. Training Parameters

The training of the unsupervised model was executed using an Adam optimizer with a learning rate set to 0.0001. The training process spanned over 30 epochs. A minibatch of 32 was used for the training of the model.

C. Tensor Direct Linear Transform (Tensor DLT)

A critical component of the architecture is the Tensor DLT layer, designed to transform the 4-point parameterization output into a full 3x3 homography matrix. This layer is key for the network's learning, as it allows for backpropagation while performing this transformation."

D. Spatial Transformation Layer

The Spatial Transformer Layer extends the capabilities of the network by enabling inverse warping of the images. This layer computes the normalized inverse of the homography

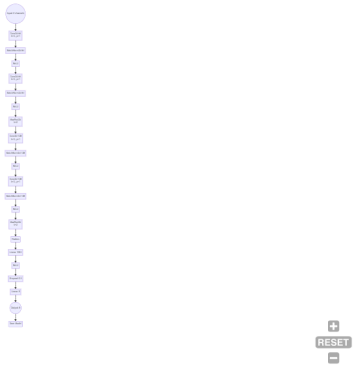

Fig. 14. Unsupervised Learning Model

matrix, generating a grid of pixel coordinates used for the warping process. The implementation uses bilinear interpolation for image sampling, maintaining differentiability for the backpropagation of the loss function.

E. Unsupervised Loss Function

The unsupervised model employs an L1 pixel-wise photometric loss function. This function measures the difference between the warped image and the target image, driving the network to minimize these discrepancies. The loss function's design allows the network to learn homography transformations without the need for ground truth labels, a significant advantage over supervised method.

$$L_{\text{photometric}} = \frac{1}{N} \sum_{i=1}^{N} \left| I_{\text{warped}}(i) - I_{\text{target}}(i) \right|,$$

where:

- N is the total number of pixels,
- $I_{\text{warped}}(i)$ is the pixel value of the warped image at pixel i,
- $I_{\text{target}}(i)$ is the pixel value of the target image at pixel i.

F. Results and Discussion

Our unsupervised model demonstrated good performance in estimating homography transformations, particularly in scenarios involving large image displacements and significant illumination variations. The model's adaptability and speed, attributed to its deep learning foundation and unsupervised nature, make it highly suitable for real-time applications in imaging and robotics. Table II shows EPE values for the same.

Approach	Train Data	Test Data	Val Data
Unsupervised Learning	24.9	25.4	20.3

TABLE II
EPE UNSUPERVISED IN PIXELS

The model's robustness against large displacements and illumination changes is noteworthy, especially considering the challenges these factors pose in traditional homography estimation methods. Our results indicate that the unsupervised learning approach, by leveraging the strengths of deep learning, offers a promising avenue for advanced homography estimation in complex real-world scenarios.

IV. CONCLUSION

In this project, we have explored two distinct approaches for estimating homographies between image pairs - a traditional approach and a deep learning approach encompassing both supervised and unsupervised methodologies. We began with a traditional approach, where we engaged in corner detection, adaptive non-maximal suppression, feature descriptor extraction, feature matching, and homography calculation using RANSAC. Although this method provided foundational insights, it also revealed limitations, particularly in feature matching and handling large-scale variations and occlusions.

Shifting to the realm of deep learning, we harnessed the power of neural networks for homography estimation. In the supervised learning approach, despite its efficacy in certain scenarios, we encountered limitations in generalization, especially when dealing with data discrepancies between training and real-world applications. This led us to the unsupervised learning approach, which emerged as a robust and adaptable solution. The unsupervised model, built upon a deep convolutional neural network, demonstrated remarkable performance in estimating homography transformations, especially in challenging conditions involving large image displacements and significant illumination variations.

Though we missed the visualization of the output, our findings underscore the potential of unsupervised deep learning in image stitching and homography estimation. This approach not only addresses the limitations observed in traditional and supervised methods but also opens avenues for real-time, adaptive homography estimation in dynamic environments. As we move forward, the integration of such advanced computational methods in imaging and robotics signifies a leap towards more intelligent and versatile systems capable of handling the complexities of the real world.

REFERENCES

[1] https://docs.opencv.org/4.x/dc/d0d/tutorial_py_features_harris.html
[2] https://github.com/tynguyen/unsupervisedDeepHomographyRAL2018/blob/master/code/utils/utils.py
[3] https://www.mdpi.com/2079-9292/12/24/4977
[4] https://arxiv.org/pdf/1606.03798.pdf
[5] https://arxiv.org/pdf/1606.03798.pdf
[6] https://arxiv.org/abs/1709.03966
[7] https://arxiv.org/abs/1709.03966