Project 1 - My AutoPano
CMSC733
Using 5 Late Days

Abubakar Siddiq
M.Eng Robotics
UID: 120403422
UMD College Park
Email: absiddiq@umd.edu

Gayatri Davuluri
M.Eng Robotics
UID: 120304866
UMD College Park
Email: gayatrid@umd.edu

Srividya Ponnada
M.S. Computer Science
UID: 120172748
UMD College Park
Email: sponnada@umd.edu

***Abstract*—The report focuses on two methods to stitch two or more images and create a panorama, the methods include the traditional computer vision method which includes feature detection and matching and computing the homography to warp and stitch the images, and the second method uses the deep learning models both supervised and unsupervised to compute the homography between a pair of images, more information is covered in the further sections of the paper. The results of the traditional approach and the deep learning approaches are compared and analyzed.**

I. Phase 1: Traditional Approach

We first detect the corners and features from the images and then match the features (feature matching) to compute the homography between the pair of images.

A. Feature Detection

For feature detection we used SIFT. Detecting the good features between two images is necessary so that they can be matched to compute the homography further. And also Harris method is used for detecting the corners. Then Adaptive non-maximal suppression(ANMS) is implemented to filter the noisy points.

To obtain uniform corner points we use the Shi-Tomashi Corners Detection method from OpenCV by calling the function cv2.goodFeaturesToTrack which gives the corner co-ordinates.

B. Adatptive Non-Maximal Suppression (ANMS)

ANMS finds corners that are true local maxima and evenly distributes the unevenly distributed detected corners.

C. Feature Descriptor

To describe the features at each point after detecting the corner points a feature descriptor is defined. A 40x40 patch centered around the specified point is extracted. Subsequently, this patch is downsampled to 8x8 and subjected to Gaussian blurring. The resulting output is reshaped into a 64x1 vector, and standardization is applied to ensure a mean of 0 and a variance of 1.

Fig. 1. Feature detection using cv2.goodFeaturesToTrack Image 1 Set 1

Fig. 2. Feature detection using cv2.goodFeaturesToTrack Image 2 Set 1

Fig. 3. Corner detection using cv2.cornerHarris Image 1 Set 1

Fig. 4. Corner detection using cv2.cornerHarris Image 2 Set 1

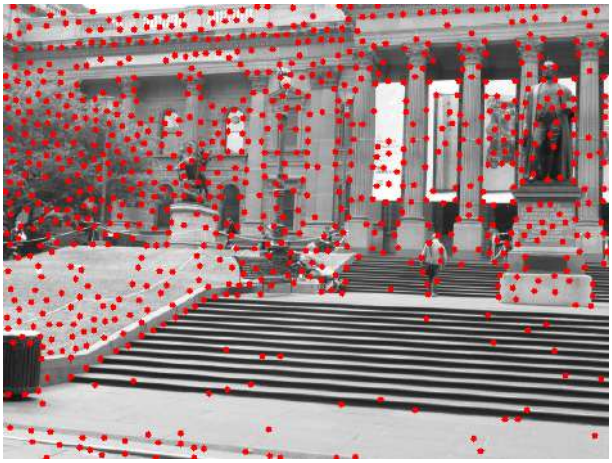

Fig. 5. Corner detection using cv2.cornerHarris Image 3 Set 1

Fig. 6. AMNS of Image 1 in Set 1

Fig. 7. AMNS of Image 2 in Set 1

Fig. 8. AMNS of Image 2, 3 Set 1

D. Feature Matching

In feature matching, we compute the matches by first considering the sum of the differences between a corner point in image 1 and all points in image 2. The ratio of the best match (the point with the lowest distance) and the second-best match (the point with the second-lowest distance) is taken, and if it is more than a considered threshold we accept the value, else we reject it.

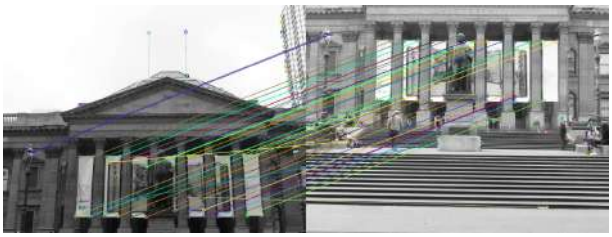

Fig. 9. Feature Matching of Set 1

E. RANSAC for outlier rejection and to estimate Robust Homography

To filter out wrong matches (Outliers) and obtain reliable homography Random Sample Consensus method is used. This involves the following steps: Firstly, four feature pairs are randomly selected from both image 1 and image 2. Then the homography (H) between the previously selected point pairs is computed. The inliers are determined based on a user-defined threshold for the Sum of Square Differences (SSD) function. These steps are repeated until a specific percentage of inliers is surpassed. Subsequently, the largest set of inliers is retained. Finally, the least-squares estimation of H is recomputed using all of the inliers.

F. Stitching Images

We employed a straightforward blending technique to merge two images. Utilizing the computed homography matrix, we mapped image 1 onto the projective plane of image 2. Subsequently, to prevent negative values, we shifted the projected image and seamlessly overlaid it onto image 2.

Fig. 10. Stitching of Images 2 and 3 in Set 1

G. Analysis

While stitching Sets having more than 4 images we faced an issue. For given set 3 and Test Set 2 we had to consider each pair individually and since the view was vast could not result in accurate stitching. The resulting images are shown in Figures 13-23.

Fig. 11. Final Panorama of Set 1

Fig. 12. Final Panorama of Set 2

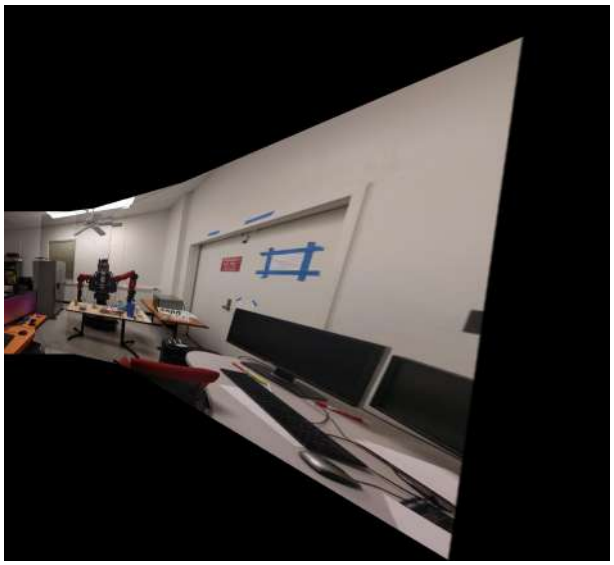

Fig. 13. Partial panorama of Set 3

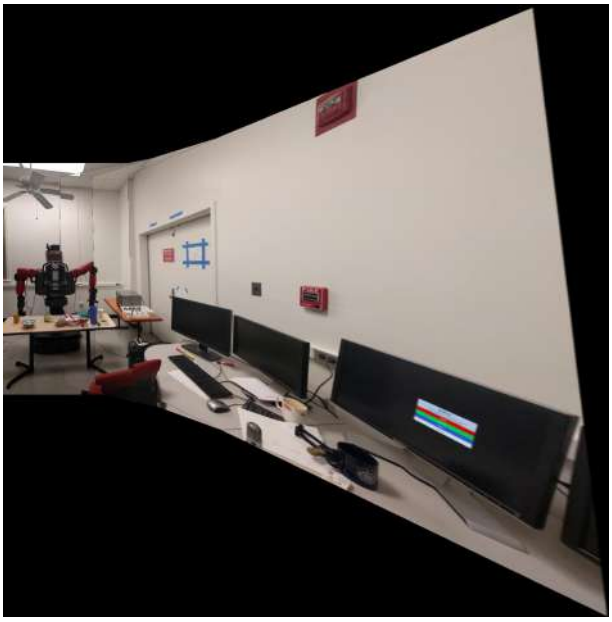

Fig. 14. Partial panorama of Set 3

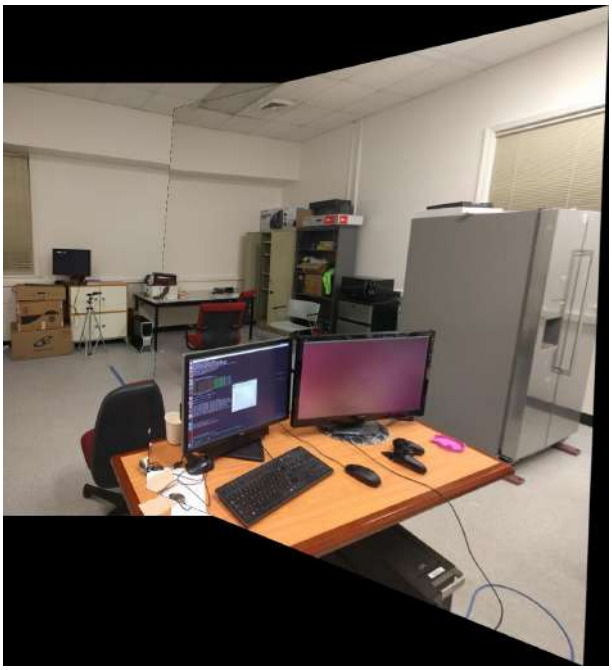

Fig. 15. Partial panorama of Set 3

II. Phase 2:

Here we implement supervised [1] and unsupervised [2] deep learning approaches to estimate homography between 2 images. The 2 image patches Pa and Pb are given as input to the deep neural networks and stacked resulting in an output of 100x100x6 where each patch is of size 100x100x3. The dataset used to implement these models is the MS-COCO dataset. By applying random projective transformations on the actual images we generate synthetic training and testing samples.

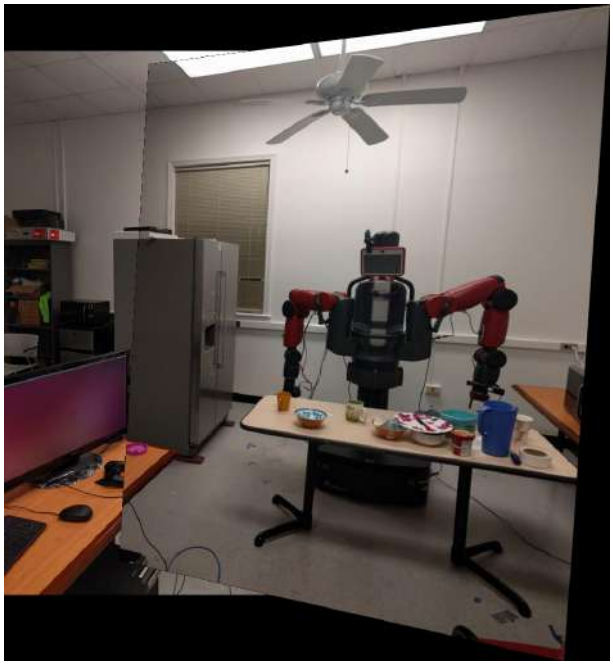

Fig. 16. Partial panorama of Set 3

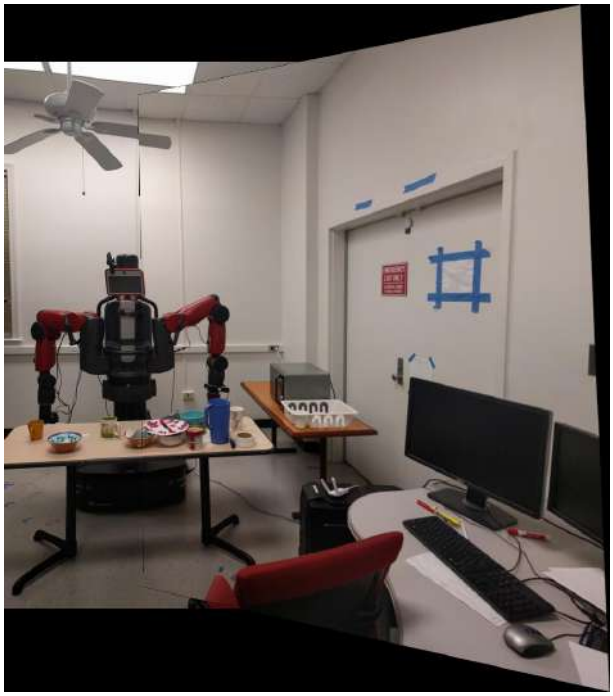

Fig. 17. Partial panorama of Set 3

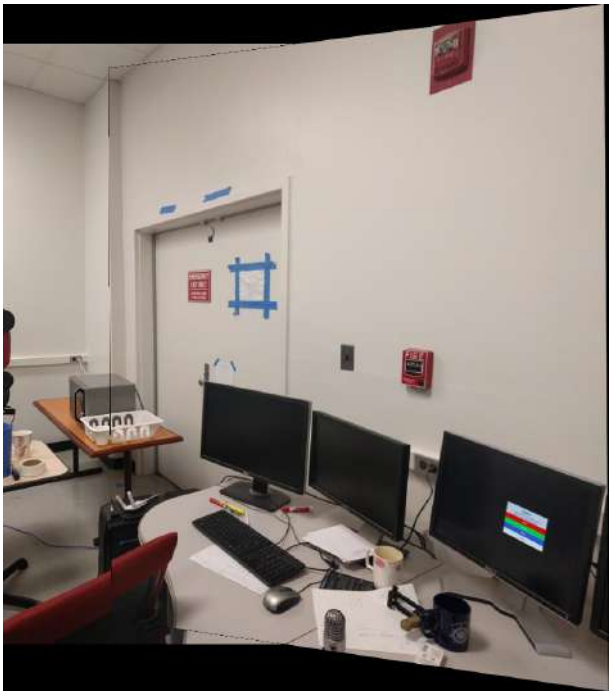

Fig. 18. Partial panorama of Set 3

Fig. 19. Final Panorama of Test Set 1

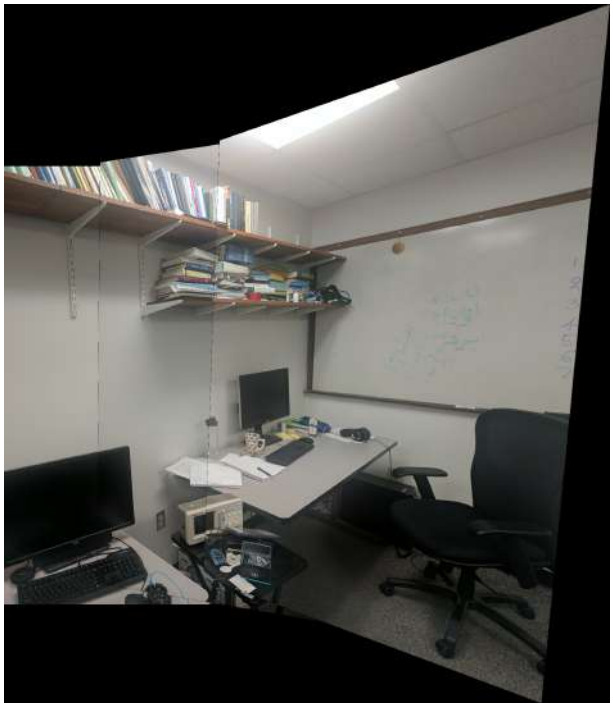

Fig. 20. Partial panorama of Test Set 2

Fig. 21. Partial panorama of Test Set 2

Fig. 22. Partial panorama of Test Set 2

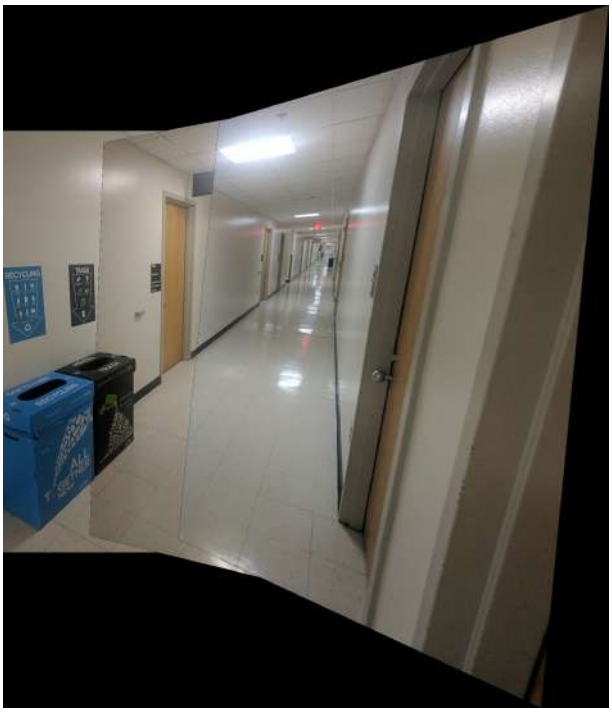

Fig. 23. Final panorama of Test Set 3

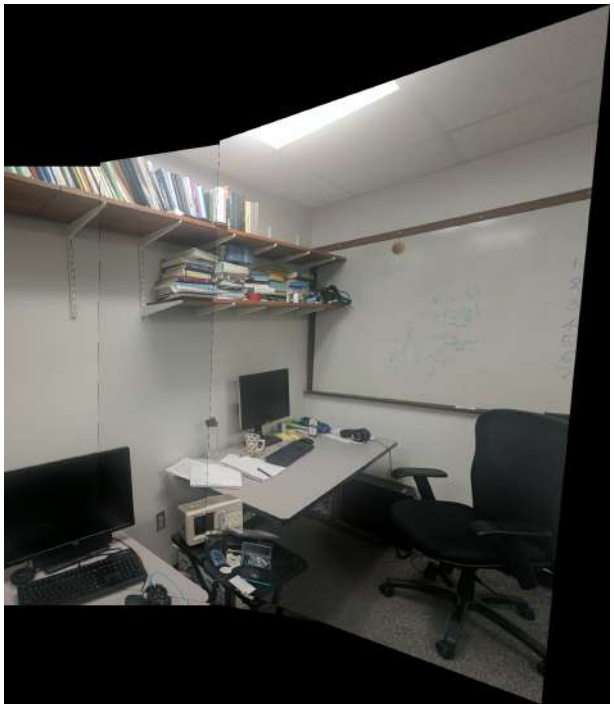

Fig. 24. Partial panorama of Test Set 4

Next, we describe the data generation procedure.

A. Data Generation

We create synthetic data for training a homography estimation model through a multi-step process. Initially, we define a function called "warp_image_with_random_patch" to generate random patches from input images and warp them using randomly generated homographies. This function outputs the original patch (Pa), the warped patch (Pb), and the corresponding 4-point homography matrix (H4pt). Following this, we designate input and output directories: the former containing original images and the later storing the generated patches (Pa and Pb). Then, we iterate through each image file in the input directory, loading them with OpenCV functionalities. For each loaded image, synthetic data is generated using the "warp_image_with_random_patch" function, producing random patches and their associated ground truth homographies. These generated patches are saved to the output directories. Additionally, we record the 4-point homography matrices corresponding to each image pair in a human-readable format. This output file serves as a lables for training the models and also for the ground truth homographies linked to the generated image patches. Ultimately, this process yields a dataset of paired image patches and their respective ground truth homographies, essential for training accurate and robust homography estimation models.

B. Supervised Learning Model

The supervised learning model predicts homography matrices from paired image data. It utilizes a convolutional neural network (CNN) architecture. The model is trained on image pairs where the corresponding homography matrices are known. The goal is to learn the mapping between the input image patches and their corresponding homography matrices. The Figure 25 depicts the training loss and validation loss of the supervised model training over the epochs. We have also obtained a Mean Corner Error(MCE) of 5.5556 for the training and 5.4520 for the validation datasets respectively. A Mean Corner Error(MCE) of 6.1560 is obtained on the testing dataset which signifies a decent model performance in estimating the four point homography (H4pt).

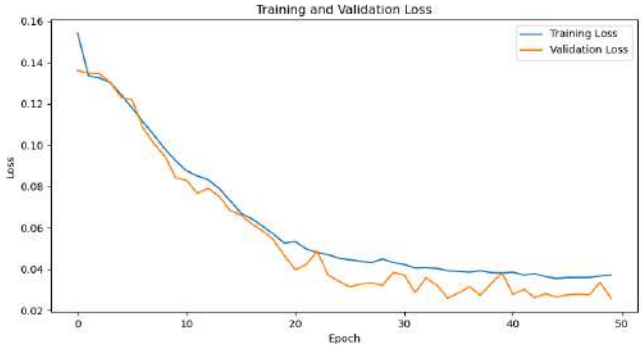

Fig. 25. Training and Validation Loss of Supervised Model per epoch

Architecture The model architecture comprises eight convolutional layers followed by fully connected layers. Each convolutional layer is followed by batch normalization and ReLU activation functions to enhance training stability and promote non-linearity. The convolutional layers capture spatial features from the input images, while the fully connected layers perform the final transformation to predict the homography matrix parameters.

Training Parameters We use the Adam optimizer and mean squared error (MSE) loss function to compute the discrepancy between predicted and ground truth homography matrices. The model is trained for 50 epochs. . A mean absolute error is employed as the training performance metric.

C. Unsupervised Learning Model

The practical utility of deep homography estimator networks is constrained by the need for a substantial quantity of ground truth data in the supervised learning paradigm. Consequently, [2] introduced an alternative unsupervised learning methodology for homography estimation networks.

Architecture The unsupervised homography estimation model features a robust architecture tailored to predict homography matrices directly from input image patches without the need for labeled data. At the core of the model lies the HomographyNetPyTorch, a neural network composed of convolutional layers followed by fully-connected layers. These layers work synergistically to extract intricate features from the input patches, with batch normalization layers ensuring stable training by normalizing the activations. ReLU activation functions introduce non-linearity, crucial for capturing complex relationships within the data, while max-pooling layers downsample feature maps, enabling the model to learn hierarchical representations efficiently. The final fully-connected layers produce a vector representing the predicted homography parameters, facilitating the subsequent construction of full homography matrices.

Once the predicted homography parameters (H4) are obtained, they are processed using auxiliary functions like construct homography matrix and apply homography. construct homography matrix augments H4 to form complete 3x3 homography matrices, essential for transformation operations. On the other hand, apply homography leverages the constructed homography matrices to warp the corner points of the input patches, yielding transformed coordinates. These transformed coordinates provide insights into the geometric distortions induced by the predicted homographies, aiding in tasks such as image registration and object recognition. By seamlessly integrating these components, the unsupervised homography estimation model offers a versatile solution for scenarios where labeled datasets are scarce, enabling accurate homography prediction without the need for extensive manual annotation efforts..

Training Parameters

We employed a training configuration similar to the supervised model, utilizing the Adam optimizer for 50 epochs. Monitoring the training performance of the unsupervised model involved the utilization of the photometric loss function described in Equation 1. And we didn't get any pretty results with the unsupervised model.

$$\text{Loss}_{\text{Photometric}} = ||w(I_A, H_{4AB}), I_B||_1 \tag{1}$$

D. Performance Results

Supervised Learning Model After training the Supervised Model for 50 epochs, we obtained a mean corner error of 5.5556. For testing we obtained a mean corner error of 6.1560.

Unsupervised Learning Model After training the Un-Supervised Model for 50 epochs, we obtained a mean corner error of 49.85, which is a poor performance in the unsupervised model.

REFERENCES

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

[2] Nguyen, Ty, et al. "Unsupervised deep homography: A fast and robust homography estimation model." IEEE Robotics and Automation Letters 3.3 (2018): 2346-2353.

[3] M. Jaderberg, K. Simonyan, A. Zisserman,et al., “Spatial transformer-networks,” inAdvances in Neural Information Processing Systems,2015, pp. 2017–2025
[4] https://cmsc733.github.io/2022/proj/p1/