RBE 549: Project 1 My AutoPano

Thabsheer Jafer Machingal
Robotics Engineering
Worcester Polytechnic Institute
Worcester, Massachusetts 01609
tmachingal@wpi.edu

Krishna Madhurkar
Robotics Engineering and
Electrical and Computer Engineering
Worcester Polytechnic Institute
Worcester, Massachusetts 01609
ksmadhurkar@wpi.edu

***Abstract*—The Purpose of the project 1, My AutoPano, is to stitch two or more images in order to create one seamless panorama image. One of the criterion for the image stitching process is that both the images must share at least 30-50% similarity between them. The Project has two phases; phase 1 is traditional approach and phase 2 is deep learning based image stitching**

Submission Note: 1 late day used

I. Introduction

In this project we explore the feasibility of producing a seamless panorama image that is put together by image stitching multiple images, ideally with 30-50 percent of matching features.

Image stitching or photo stitching is the process of combining multiple photographic images with overlapping fields of view to produce a segmented panorama or high-resolution image, This can be achieved using traditional approach as explained ion the phase 1 of this project. The phase 2 of this project focuses on image stitching using deep learning method.

We detail the steps taken and results in the Phase 1 and Phase 2 sections.

II. PHASE 1: Traditional Approach

Using the traditional approach of computer vision, we are stitching two or more images together to form one seamless image. An overview of the method is given below. A pair of images is stitched together for any number of images with similarities (similar feature). The features used are corners in the images. A sample image set is stitched below using the traditional approach.

A. *My AutoPano: Algorithm*

1) Corner Detection: The first step is to predict corners, features, in the image. For this purpose we are using Harris Corner Detector to spit out the confidence score of a feature being a corner and Shi-Tomasi algorithm to predict the coordinates of these corners.

2) Adaptive Non-Maximal Suppression (ANMS): ANMS algorithm is given below. ANMS suppresses poor features and extract N number of best corners.

3) Feature Matching and Drawing the matches: In this step, we are matching the common features in both image 1 and image2 and constructing a line between them to visualize the match between two images.

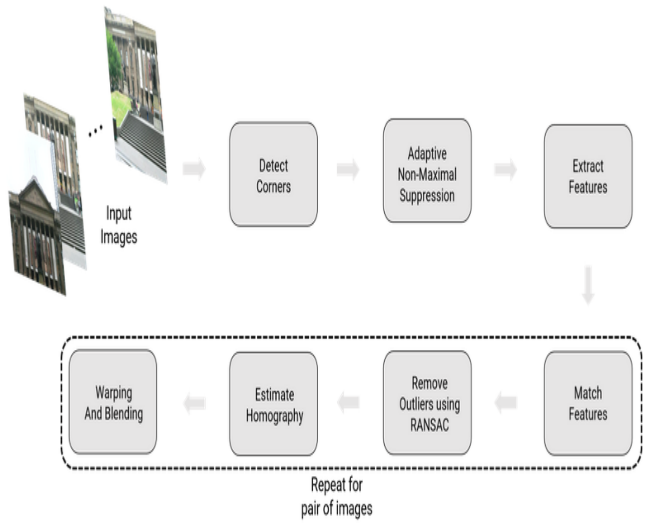


Fig. 1. An overview of Panorama stitching using traditional method

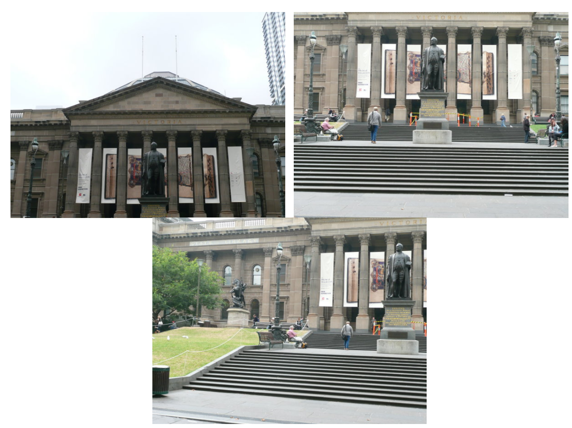


Fig. 2. A sample of three images

4) Random Sample Consensus or RANSAC: RANSAC is used for outlier rejection and to estimate Robust Homography. IN the below given ifigure, it is evident that, many of the bad mathces are removed and only the good ones are preserved.

5) Blending of images: The final step in the image stitching process is to blend the image pairs to matches brightness and

Fig. 3. Output image after stitching three images together

Fig. 4. corner detected in the two images

Input : Corner score Image (C_{img} obtained using `cornermetric`), N_{best} (Number 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**
      $\text{ED} = (x_j - x_i)^2 + (y_j - y_i)^2$
    end
    if $ED < r_i$ **then**
      $r_i = \text{ED}$
    end
  end
end

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

Fig. 5. ANMS Algorithm

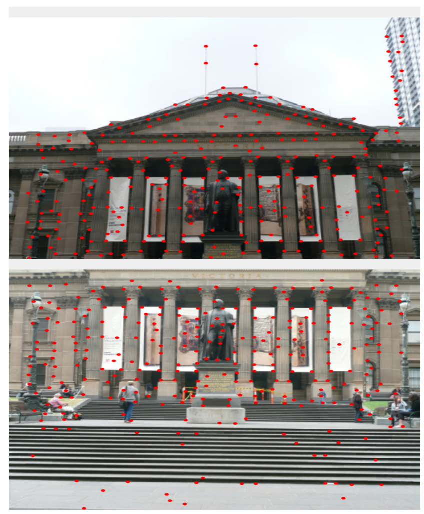


Fig. 6. ANMS applied to the corners in the two images

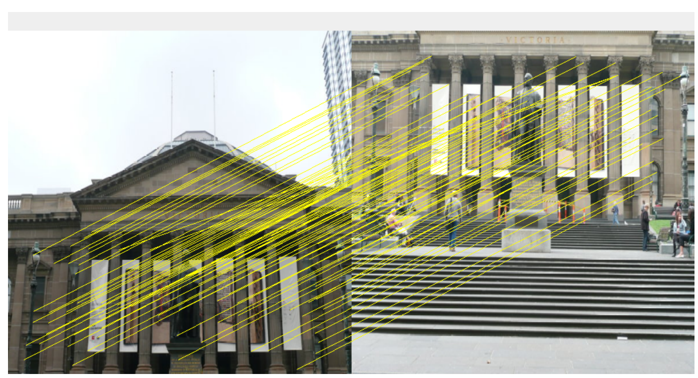


Fig. 7. Feature matching between the two images

exposure. There are multiple ways to employ bending in image stitching. We used mean method, were the average values of intensities in every pixel is applied to the previous values in the original image.

B. Parameters in the algorithm

The algorithm explained in the previous section has many parameters that need to tuned. Harris corner detector or Shi-Tomasi detector has parameters that controls the threshold and sensitivity for feature extraction.

C. Testing the algorithm

The algorithm is tested on more test images given.

1) Test set 1:

2) Test set 2: For test set 2, our algorithm did not find enough matches, the inlier count was three, which is less to make corners.

3) Test set 3:

Fig. 8. RANSAC applied to the same set of images

Fig. 9. TEST SET 1

4) Test set 4: The test set 4 has inconsistent images, the algorithm returned no common features between all the images.

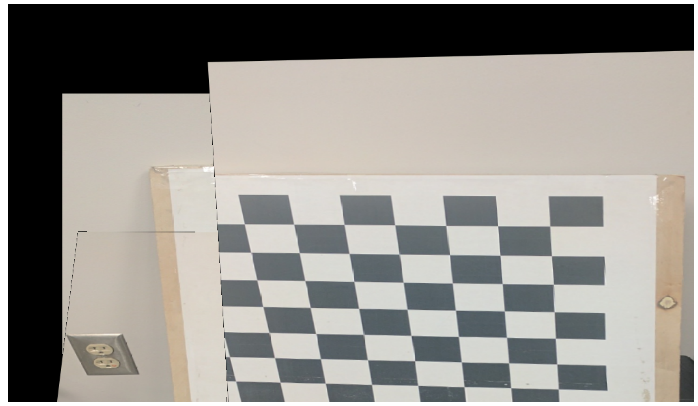


Fig. 10. result from TEST SET 1

Fig. 11. TEST SET 3

III. PHASE 2: DEEP LEARNING APPROACH

A. Data Generation

To train a convoluational neural network and estimate homography between a pair of image, we need two images with know homography between them.

Fig. 12. result from TEST SET 3

Fig. 13. TEST SET 4 images

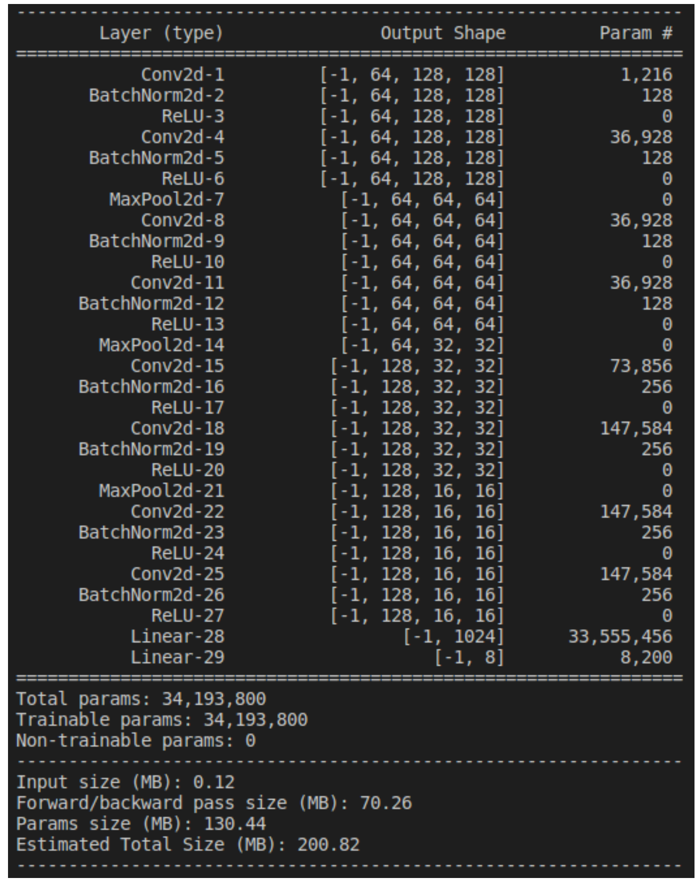
B. HomographyNet

The deep learning model used in this project is named as HomographyNet. HomographyNet has eight convoluational layers and two fully connected linear layers. The model takes in two grayscale images stacked in channel dimesnion and outputs 1x8 vector of four point homography (H_{4pt}).

C. Supervised Learning

Supervised deep learning adjust the weights of parameters and the biases according to the error in previous epoch. An overview of the Supervised deep learning algorithm and the network is given in figure 2. Loss metric used in this architecture is L2 Loss (Mean Squared error) between predicted 4-point homography ($\hat{H}_4pt$) and Ground truth 4-point homography vector (H_4pt).

1) Parameters and tuning of hyper parameters: The loss function used is L2 loss and Adam optimizer is used. Learning rate used was to 0.0001. We noticed that when learning rate decreased from 0.001 to 0.0001, the loss was significantly reduced. As for Number of epochs, more the number of epochs, the loss significantly reduced over time. Increasing batch size, makes the loss go high and it was set as 15.Also noticed that when Batch size was 8, the loss was significantly low compared to that at batch size=15. So, setting low batch size actually is favourable as more corrections in weights are made with diverse batches.



```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1        [-1, 64, 128, 128]           1,216
       BatchNorm2d-2        [-1, 64, 128, 128]             128
              ReLU-3        [-1, 64, 128, 128]               0
            Conv2d-4        [-1, 64, 128, 128]          36,928
       BatchNorm2d-5        [-1, 64, 128, 128]             128
              ReLU-6        [-1, 64, 128, 128]               0
         MaxPool2d-7          [-1, 64, 64, 64]               0
            Conv2d-8          [-1, 64, 64, 64]          36,928
       BatchNorm2d-9          [-1, 64, 64, 64]             128
             ReLU-10          [-1, 64, 64, 64]               0
           Conv2d-11          [-1, 64, 64, 64]          36,928
      BatchNorm2d-12          [-1, 64, 64, 64]             128
             ReLU-13          [-1, 64, 64, 64]               0
        MaxPool2d-14          [-1, 64, 32, 32]               0
           Conv2d-15         [-1, 128, 32, 32]          73,856
      BatchNorm2d-16         [-1, 128, 32, 32]             256
             ReLU-17         [-1, 128, 32, 32]               0
           Conv2d-18         [-1, 128, 32, 32]         147,584
      BatchNorm2d-19         [-1, 128, 32, 32]             256
             ReLU-20         [-1, 128, 32, 32]               0
        MaxPool2d-21         [-1, 128, 16, 16]               0
           Conv2d-22         [-1, 128, 16, 16]         147,584
      BatchNorm2d-23         [-1, 128, 16, 16]             256
             ReLU-24         [-1, 128, 16, 16]               0
           Conv2d-25         [-1, 128, 16, 16]         147,584
      BatchNorm2d-26         [-1, 128, 16, 16]             256
             ReLU-27         [-1, 128, 16, 16]               0
           Linear-28                 [-1, 1024]      33,555,456
           Linear-29                    [-1, 8]           8,200
================================================================
Total params: 34,193,800
Trainable params: 34,193,800
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.12
Forward/backward pass size (MB): 70.26
Params size (MB): 130.44
Estimated Total Size (MB): 200.82
----------------------------------------------------------------
```

Fig. 14. Summary of the model, HomographyNet

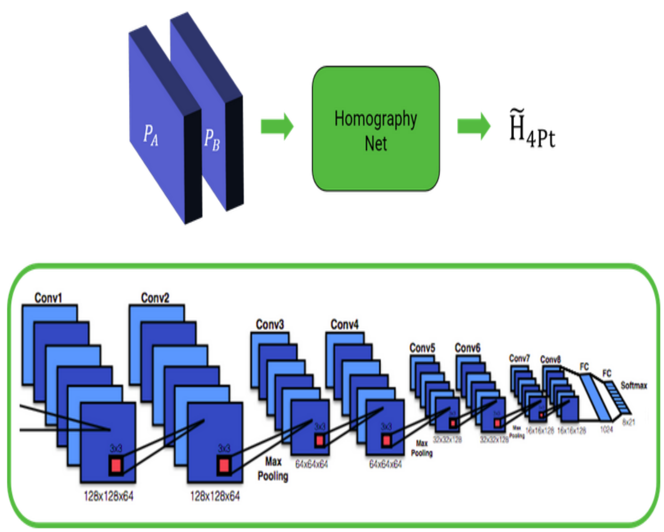


Fig. 15. Supervised Learning architecture [1]

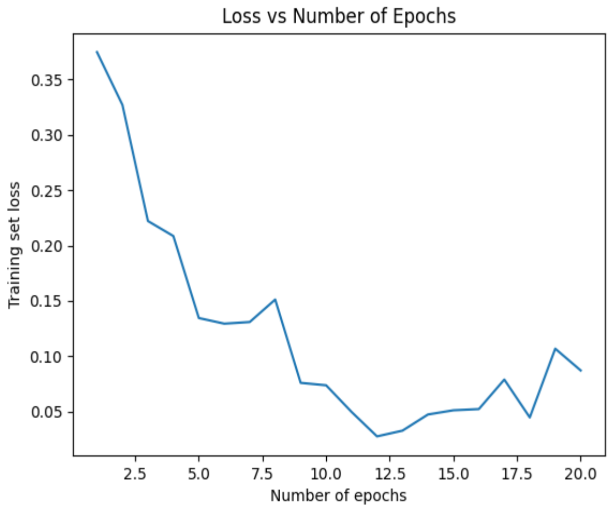


Fig. 16. Training set loss over number of epochs

REFERENCES

[1] *Course material reffered*, http://prg.cs.umd.edu/cmsc733

[2] *Unsupervised Deep Homography*, https://doi.org/10.48550/arXiv.1709.03966

[3] *Poission Image Editing*, $2003_{s}iggraph_{p}erez$

[4] *Deep Image Homography Estimation*, https://arxiv.org/pdf/1606.03798.pdf