COMP562 Project Report: An Evaluation of the Pyramid Stereo Matching Network (PSMNet)

Helen Qin
Siqing Xu
Huihao Peng
Yijie Sun

Abstract

Recent work proposed by Jia-Ren Chang and Yong-Shen Chen has suggested that Pyramid Stereo Matching (PSM) Network, can potentially be an efficient solution to robust and accurate 3D reconstructions. The purpose of our project is to evaluate the performance of the PSMNet on different combinations of datasets and then to seek potential improvements for the network. To achieve the goals, we will test the PSMNet first by validating pre-trained network on public dataset such as KITTI 2012 and KITTI 2015 and then training and testing on two synthetic datasets generated by ourselves. During the evaluation, we would like to see the robustness and accuracy of PSMNet in various scenes. We also propose to add additional training layers to enhance network performance.

1. Introduction

3D reconstruction pipeline with stereo vision has been widely employed in real-world applications such as Augmented Reality surgeries. However, most current approaches on this technique show a series of shortcomings significantly impacting the robustness and accuracy of 3D reconstruction. First, the basic block matching algorithm on CPU is computationally expensive and its performance can vary with the cost function. Secondly, the applicability of block matching algorithm on depth camera is restricted by the capturing device and its physical size. Therefore, recent work has been focusing on improving the traditional approaches with machine learning and neural networks (Florence et al., 2018). In this paper, we plan to evaluate Pyramid Stereo Matching Network (PSMNet) proposed in Chang’s paper (Chang, 2018) which is a deep learning approach to this problem.

2. Methodology

2.1. Pyramid Stereo Matching Network

As explained in Chang’s paper, our model Pyramid Stereo Matching Network (PSMNet) consists of two main modules: a spatial pyramid pooling and a stacked hourglass 3D CNN. The network will first take a pair of raw stereo Left-and-Right images as input. The spatial pyramid pooling module takes advantage of the capacity of global context information by aggregating context in different scales and locations to form a cost volume. The 3D CNN learns to regularize cost volume using stacked multiple hourglass networks in conjunction with intermediate supervision. The final output is a left disparity map of corresponding image pair. The paper uses three dataset to evaluate its performance: KITTI 2012, KITTI 2015 and SceneFlow (Chang, 2018).

2.2. Retraining the Whole Network

We retrain the whole network with 500 epochs on both the datasets given in Chang’s paper (Chang, 2018) and our customized synthetic dataset. The first purpose of retraining the network is to evaluate its robustness on different, unseen datasets. Second, we also use the retrained network as the initialization of the training / tuning in other steps.

2.3. Fine-Tuning the Whole Network

The second attempt of improving this network result is to fine-tune the pre-trained network on more datasets. Our expectation is that by tuning the network on a greater number of diverse data, the robustness of this model will improve. We start with 300 epoch at training to ensure convergence.
2.4. Transfer Learning

During transfer learning, we freeze the parameters of all layers other than the last three parallel convolutional layers. Thus, this will be less computationally expensive compared to the previous fine-tuning step. We start with 300 epoch at training to ensure convergence.

Figure 1. Network structure: the unfrozen layers during transfer learning are inscribed in the red rectangle at the bottom right of the graph.

2.5. Evaluation

During evaluation, we will use and modify the error formula in Chang’s paper, which is called the three pixel error. This error counts the percentage of output disparities whose differences from their ground-truth disparities are greater than three pixels.

Mathematically, the three pixel error percentage in one image of provided dataset is defined as below:

\[
\text{New Three Pixel Error} = \frac{p_e}{\sum_{i=1}^{N} p_i} \times 100\%
\]

For a averaged three pixel error percentage in a set of disparity maps, it is written as below:

\[
\text{Averaged Three Pixel Error} = \frac{\sum_{i=1}^{N} \text{Three Pixel Error}_i}{N}
\]

Based on the given evaluation method, we have made two modifications. First, we derive a three pixel error without background by modifying its denominator to exclude counting black pixels. Given that most training and testing data contain a significant amount of black background, this will help the error evaluation to focus on regions of interested objects. The new three pixel error without background is defined as follows:

\[
\text{Three Pixel Error Without Background} = \frac{p_e}{p_i} \times 100\%
\]

Second, we loosen the error strictness by adding a ten pixel error evaluation. Our synthetic datasets generated contain objects that are significantly closer (600X) to the camera. This will result in larger disparity values and thus the estimation error produced by the network is expected to be greater as well. Hence the ten pixel error evaluation is introduced to accommodate this disparity change.

3. Experiments

We evaluated different combinations of our methods in Section 4 on four stereo datasets: KITTI 2012, KITTI 2015, Synthetic Dataset 1 and Synthetic Dataset 2. We then further divide our experiments into three categories according to their foci: the network’s performances on the synthetic dataset 1, performances on transfer learning, and robustness, as described starting from Section 3.2.

3.1. Dataset Details

We used four datasets for our training and evaluation:

1. Synthetic Dataset 1: A software-generated dataset that contains 180 image pairs of pegboard objects. Each image has dense ground-truth disparities. The objects are only around 20 centimeters away from the stereo cameras.

   Figure 2. Example image pair in the synthetic dataset 1.

2. Synthetic Dataset 2: A software-generated dataset that contains 180 image pairs of pegboard objects. Each image
has dense ground-truth disparities. The objects are only around 20 centimeters away from the stereo cameras.

Figure 3. Example image pair in the synthetic dataset 2.

3. KITTI 2012 and KITTI 2015: Both KITTI 2012 and KITTI 2015 are real-world datasets of street views. Each dataset contains around 200 training image pairs and 200 testing image pairs with sparse ground-truth disparities. Chang’s paper used both datasets for training its model. In our experiments, we first tested the PSMNet with pre-trained networks provided by the paper’s source code on the KITTI datasets.

3.2. Performances on the Synthetic Dataset 1

For the first set of experiments, we performed three previously described methods on Synthetic Dataset 1 to compare these methods. From the error table (Figure 3), we can see that the finetune on all layers have the least error rate in all categories. The comparison between the finetune and retrain results demonstrates that the legacy provided by the pretrained neural network, even though it is trained on a dataset which has significant difference from the current ones, could still help in disparity detection in current testing datasets. The comparison between the finetune on all layers and transfer learning only on last three convolutional components shows that the previous two modules, feature extraction module and stack-hourglass module, have great dependence on the datasets. If the two modules are pretrained on one dataset, and tested on other datasets which have huge difference from the pretrained one, the two modules cannot extract enough information for accurate disparity generation. Only transfer learning on last three components cannot make error quickly converge (We used 300 epochs but the error still could not converge).

3.3. Performances on Transfer Learning

For the second set of experiments, we took a deeper look at transfer learning on last three components method. From the previous experiment, we observed that the loss of this method is hard to converge if the method is transfer learning between two datasets with huge difference. In this experiment, we tested with two datasets with small difference (objects are simple, detailed and in similar distance). Starting from the saved network which is already finetuned on Synthetic Dataset 1, we tested testing set of Synthetic Dataset 2 without transfer learning as the baseline and with transfer learning to compare the results. (Figure 5) From the baseline result, we can see that the result is visually good, the shape and contour of the objects are distinguishable and clear. This baseline results show that the PSMNet can generalize some of tasks such as segmenting the objects with depth from the black background. However, the error rate of the baseline result shows that the disparity value of the region with depth are almost all incorrect. (Figure 3) By comparison, the transfer learning results have shown huge improvements from the baseline one. For example, the contour of the toy car is distinguishable compared with the baseline result. The error rate has even larger improvement. All categories of error rate converge to almost similar to those of the finetuning on all layers method from the previous experiment. This result demonstrates that the transfer learning on last three components can be effective if the datasets are similar. The previous two modules can be robust to similar datasets.

3.4. Network Robustness

Lastly, we simply want to see if the PSMNet can be generalized on two datasets with huge difference. For the
Siqing Xu, Helen Qin, Huihao Peng, Yijie Sun

first trial, we tested the test set of Synthetic Dataset 1 on the network which was pretrained on the KITTI 2015 dataset. For the second trial, after finetuning all layers on the Synthetic Dataset 1 with the pretrained network of KITTI 2015 for 300 epochs, we tested finetuned network with test set of KITTI 2015 to see if the network can still perform well after the finetuning procedure. The results for both trials are bad and the disparity map is indistinguishable to even see the objects’ shapes. This experiment shows that the PSMNet can only get reasonable results on the datasets which are similar to the one that is most recently trained or finetuned on.

Figure 6. Results of disparity estimation focusing on robustness testing. (a) KITTI 2015 network being tested on synthetic 1, and (b) network fine-tuned with first KITTI 2015 and then the synthetic dataset 1 being on KITTI 2015.

4. Conclusions

In this project, we evaluated the performance of the PSMNet and designed different improvements on it. Based on our experiments, we made three conclusions. The first conclusion is that the PSMNet is data dependable and cannot generalized well on a combination of diverse datasets, despite efforts through fine-tuning, transfer learning, as well as retraining. Second, in terms of the effectiveness of improving the network, the feature extraction and stack-hourglass module needs to be retrained or finetuned for the network to produce better results. Lastly, the PSMNet is, after all, robust enough to perform good disparity estimation if the objects are similar in terms of camera distances and backgrounds.

<table>
<thead>
<tr>
<th>Method / Test</th>
<th>Synthetic Set 1 &amp; Fine-tune</th>
<th>Synthetic Set 1 &amp; Transfer Learning</th>
<th>Synthetic Set 2 &amp; Transfer Learning</th>
<th>Synthetic Set 2 as Unseen Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;3 px (%) w/o background</td>
<td>39.52</td>
<td>69.34</td>
<td>77.93</td>
<td>44.63</td>
</tr>
<tr>
<td>&gt;3 px w/ background</td>
<td>9.07</td>
<td>16.25</td>
<td>20.4</td>
<td>9.79</td>
</tr>
<tr>
<td>&gt;10 px w/ background (%)</td>
<td>6.39</td>
<td>10.7</td>
<td>12.18</td>
<td>2.14</td>
</tr>
</tbody>
</table>

Figure 7. Three & ten pixel error evaluation table.

5. Future Work

There are still more potentials with transfer learning that we could explore in this project. We expect that training on various dataset while choosing to freeze some different layers might improve the performance of the network. We can try so and see which layers/module choices can lead to a quicker and better loss convergence. Additionally, we specifically want to unfreeze some parameters at feature learning layers which we expect to add accuracy and robustness to the network.

References
