Stereo Depth Estimation Using *DispNet* CNN

Nikola Popović
University of Belgrade, School of Electrical Engineering, Department for Signals & Systems

Introduction

This article will focus on the work in [1], a paper which has two contributions: proposing three synthetic datasets for depth, optical flow and scene flow estimation and proposing the *DispNet* Convolutional Neural Network (CNN) architecture for end-to-end depth estimation. The results of reproducing the *DispNet* CNN architecture[1] and testing it on the *FlyingThings3D* dataset, which is one of the proposed datasets, will be presented here. The dataset has 21818 labeled training stereo image pairs, and 4248 labeled test ones. There are 3D sensors like LIDAR which measure depth, but they are very expensive. Solving this problem with a cheaper pair of stereo cameras is very useful. When using Deep Learning methods, which currently have the best results on this problem, it is hard to obtain a good dataset. Datasets are usually obtained using LIDAR sensors. The problem is that even if these datasets are gathered in the real world, their labels are still estimations of depth, since they are sensor measurements. Because of that, the proposed *FlyingThings3D* synthetic dataset is of great use.

In these problems it is often the case that the two cameras lie on the same plane and are at the same height (like on some popular smartphones nowadays). If that is satisfied, to calculate the depth of objects on an image we need to know the baseline and focal length of the stereo cameras and the disparity of objects pixels - how much the object pixels shift from the left to the right image [2]. When using a stereo camera we usually know the baseline and focal length, so the problem comes down to disparity estimation. There are three groups of techniques for estimating depth that are related to this work: supervised learning with labeled stereo images, supervised learning where we only have one image and its labels and unsupervised depth estimation [3]. There are some proposed Deep Learning techniques with efficient runtime [4].

DispNet

The architecture of the *DispNet* is presented in Fig. 1. Details about each layer like the number of channels, filter dimensions, etc. can be found in [1]. The authors got an inspiration for this architecture from the *FlowNet* architecture [5]. This is an end-to-end approach because the stereo images are concatenated and passed as an input image, and the CNN predicts the disparity map as an output image. The dataset consists of stereo image pairs of flying objects with disparity map labels. In Fig. 1 we can see that this is CNN has an autoencoder architecture: the first half of the network compresses the input image to a lower dimensional feature space and

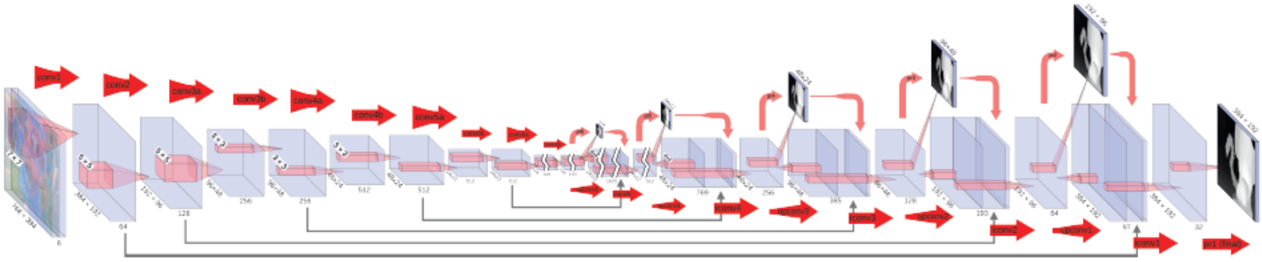


Figure 1: *DispNet* CNN architecture

[1] The code can be found at https://github.com/nikola3794/Depth-Stereo-Estimation-using-DispNet-CNN

the second half of the network uses those features to estimate the disparity map. The autoencoder architecture is useful for solving this problem because we can successfully train a deep network. If this was not the case, the number of network parameters would be much bigger and we would need considerably more hardware resources and the chances of overfitting would rise.

One of the problems that can happen in deep networks like this is that if the gradient is propagated only from the output layer, it can vanish before it reaches the beginner layers. This causes the training procedure to be very slow. Because of this, beside the loss function calculated from the output, five auxiliary loss functions are compute from the layers in the decoder, which are marked in Fig. 1. To form the output loss term L_o, mean square error (MSE) between the disparity prediction and the label is computed. To form the auxiliary loss terms L_i, MSE between the output of the layers marked in Fig. 1 and the label is computed. The final loss function is formed as $w_o L_o + w_1 L_1 + w_2 L_2 + w_3 L_3 + w_4 L_4 + w_5 L_5$, where w_i are parameters which are changed during training time and $\sum_i w_i = 1$. At the start of training, auxiliary losses at the beginning of the decoder are active. As the training goes on weights of auxiliary losses toward the output are being activated, while the weights from the ones in the beginning are slowly deactivated. Finally, at the later training stages only w_o is nonzero, which means we only care about making a good prediction at the output layer.

Results

The *DispNet* was implemented in the TensorFlow Python library on a laptop Nvidia GeForce GTX 1050 with 4gb of RAM memory. The Adam optimizer was used. Because of the network size and hardware limitations, the batch size during training was 4 images. Also, batch normalization was used, because it accelerates the training process. The network trained for 89000 batches, which lasted about a day and a half. Fig. 2 shows the learning curve, while Fig. 3 shows one prediction from the test set. At the end of the training the MSE on the test set was 10.5565. Below in the Appendix A, more images of predictions from the test set are shown. In Appendix B, prediction evolution on one test image during training is shown.

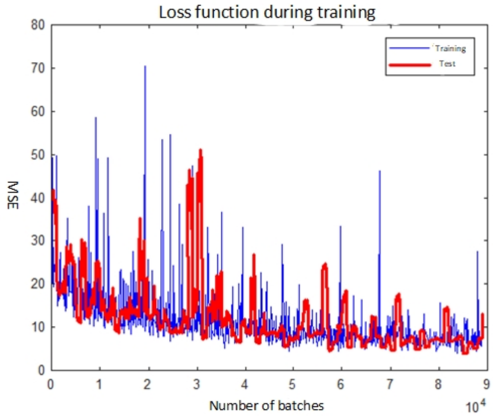
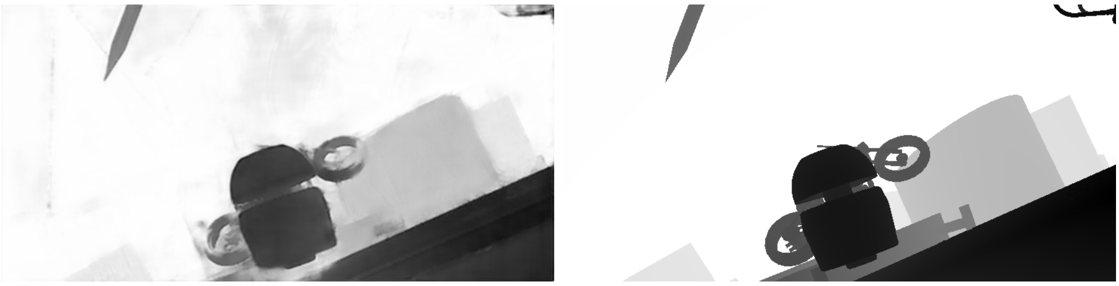


Figure 2: The value of the loss function during training



(a) Estimated depth

(b) True depth

Figure 3: Results after 89000 training batches

References

[1] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. arXiv:1512.02134.

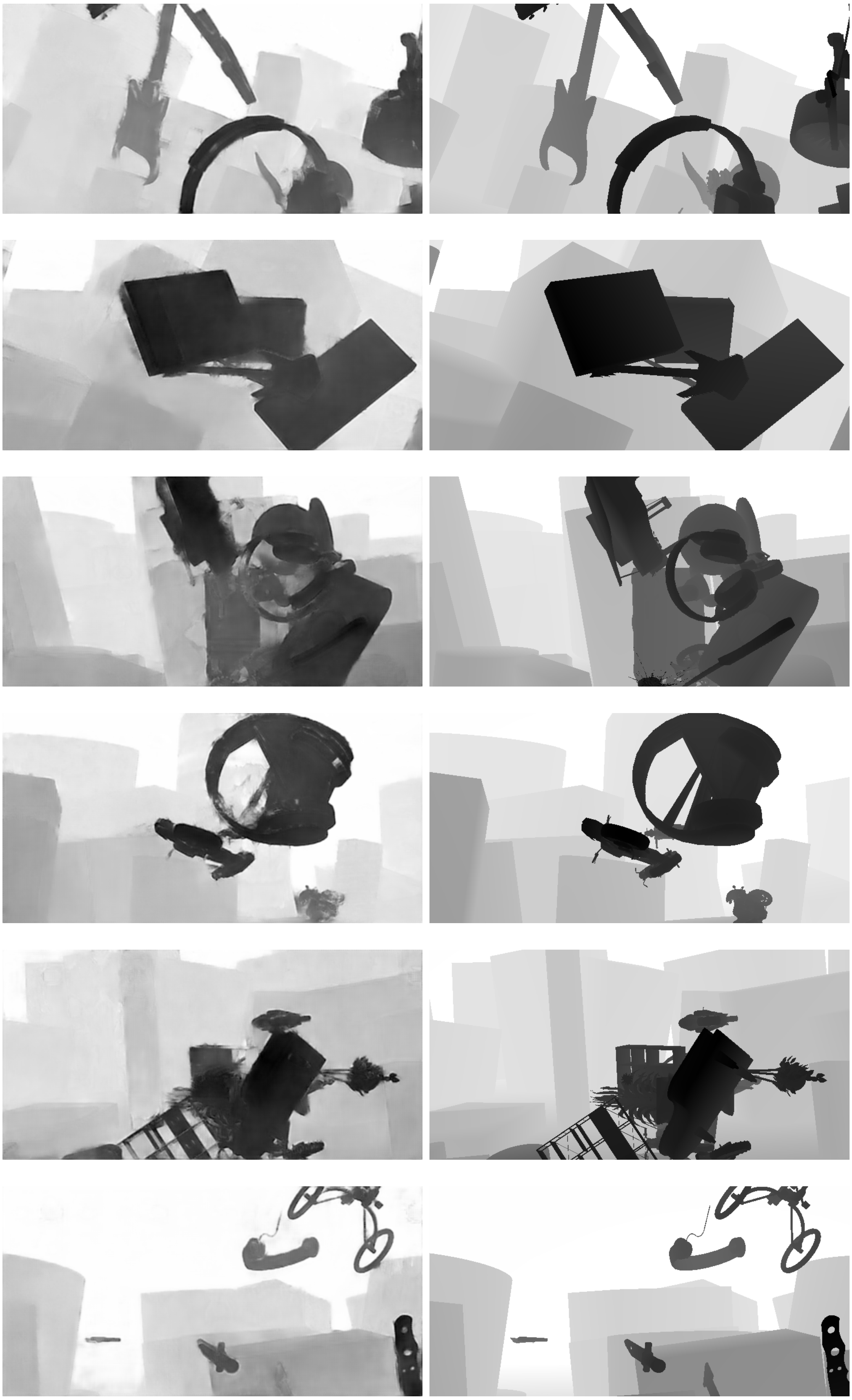
[2] R. Szeliski, *Computer Vision: Algorithms and Applications*. Berlin, Heidelberg: Springer-Verlag, 1st ed., 2010.

[3] C. Godard, O. M. Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6602–6611, 2017.

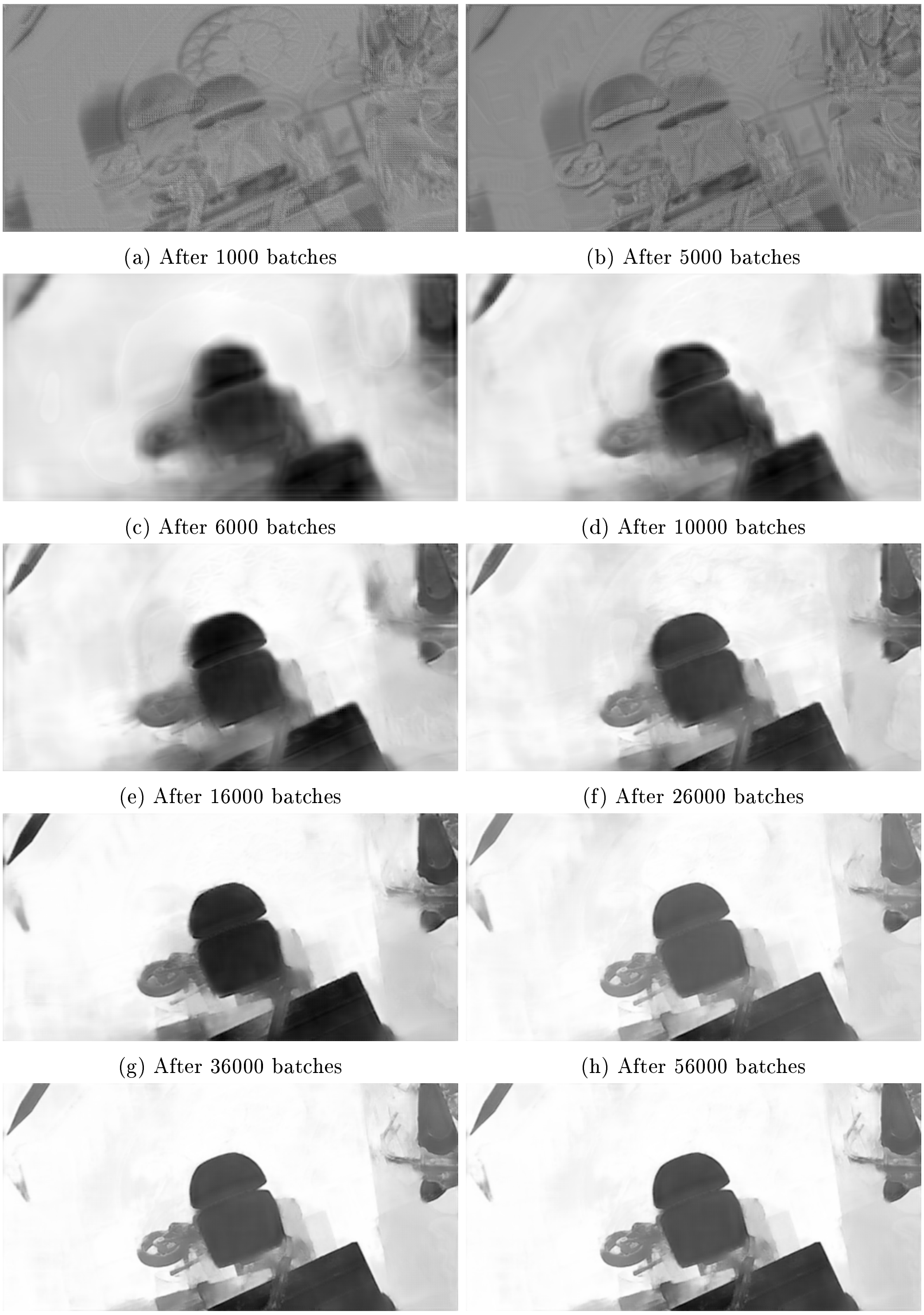
[4] W. Luo, A. G. Schwing, and R. Urtasun, "Efficient deep learning for stereo matching," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5695–5703, June 2016.

[5] P. Fischer, A. Dosovitskiy, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," *CoRR*, vol. abs/1504.06852, 2015.

Appendix A



Appendix B



(a) After 1000 batches

(b) After 5000 batches

(c) After 6000 batches

(d) After 10000 batches

(e) After 16000 batches

(f) After 26000 batches

(g) After 36000 batches

(h) After 56000 batches

(i) After 76000 batches

(j) After 89000 batches