Efficient Convolutional Neural Networks for Depth-Estimation using Atrous Spatial Pyramid Pooling and Residual Decoder Modules

Raphael Pisoni
Department of Computer Science
Free University of Bolzano
Bolzano, 39100 IT
raphael.pisoni@gmail.com

Abstract

This paper proposes a new architecture for an efficient yet scalable Fully Convolutional Neural Network (FCNN) that can be trained to estimate detailed depth maps from single or stereo images, while being fast enough for mobile or real-time applications. We apply recent architectural refinements such as atrous spatial pyramid pooling, residual decoder modules and a novel loss function and set a baseline for depth estimation on the ApolloScape real-world driving dataset as part of the evaluation.

1 Introduction

Depth perception is the process of extracting 3D information out of planar representations of a scene and has been a longstanding area of interest in computer vision. Traditional approaches include disparity estimation from binocular or multi-view stereo and motion parallax as well as shape from X [2]. Humans typically use a combination of all of these methods and thus perform very well at binocular and even monocular depth perception. With the advent of deep neural networks many kinds of models have been trained to take advantage of these simple geometrical approaches as well as learning more complex knowledge about light, object shapes, textures and the structure of a scene. This has led to a number of monocular, binocular or multi-view-stereo approaches with very promising results. Although there are some exceptionally interesting approaches to treat depth estimation as an unsupervised learning process, mostly by synthesizing it as an intermediate during stereo or motion parallax image reconstruction [10] [21], the state of the art is still training on large sets of scenes with ground truth depth obtained via LIDAR or from simulated environments. There are many use cases for accurate depth estimation from single or stereo images including augmented reality applications, simultaneous localization and mapping, scene understanding, image segmentation, object pose estimation, 3D-conversion of 2D imagery or for automated driving assistance and self-driving cars. Our approach uses a novel loss function as well as a number of recent advances and architectural refinements in convolutional neural networks to achieve state of the art results for a network of its size in both monocular and binocular depth estimation, using end-to-end training. Specifically, we propose the following:

1. A network architecture that performs well on both monocular and binocular depth estimation and can easily be scaled up or down according to performance and accuracy needs as well as computational budget.
2. An improved and efficient decoder design, based on transpose convolutions and depthwise separable convolutions and extended residual blocks, that allows us to reproduce a much higher level of detail with respect to previous models.

Preprint. Work in progress.

3. An advanced objective function that enforces orthogonality between predicted gradients and ground truth normals, as well as offering a parameter to balance the sharpness of edges and smoothness of surfaces.

2 Related Work

Convolutional Neural Networks were first used for depth estimation by Eigen et al. [8] and have subsequently significantly improved the state of the art in non-parametric monocular and binocular depth estimation. In the following we will discuss some important advances and related works in the area.

2.1 Encoder-Decoder Model

These models consist of two parts, where 1) the encoder gradually reduces the spatial dimension of the feature maps and encodes longer range information into a deeper output, which 2) gets then reexpanded in the spatial dimension by the decoder whereby object details are gradually recovered. An approach that has shown to greatly improve object-boundary reconstruction are skip-connections introduced in U-Net [18]. Hereby encoder features are directly connected to the corresponding decoder activation allowing the network to pass on fine-grained information about boundaries.

2.2 Atrous Spatial Pyramid Pooling

In order to capture both local and global details on multiple scales there have been attempts at using a multi resolution approach where the same image is typically fed into an encoder with shared weights at multiple resolutions, thus enhancing the receptive field of the networks [14]. To obtain a similarly scalable receptive field at a much lower computational cost Atrous Spatial Pyramid Pooling (ASPP) has been introduced [3]. Hereby atrous convolution layers with different rates are applied in parallel to collect data at multiple scales. Modifying the rate of atrous or dilated convolution has been shown to affect the capturing of long range information [20], resulting in promising performance in the area of semantic segmentation [3, 20], but also for monocular depth estimation [9].

2.3 Residual Networks

As simple neural networks get deeper they start getting harder to train due to vanishing gradients, greatly degrading training accuracies with our current solvers. To deal with this problem He et al. [11] introduced a network architecture using residual building blocks, resulting in a much easier optimization of very deep networks while still retaining lower complexity than many previous nets and subsequently improved image recognition accuracies by a large margin. It has since been hypothesized that the optimization of the residual mapping is easier than optimizing the original unreferenced mapping [11].

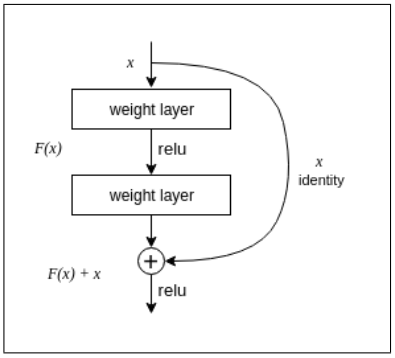


Figure 1: Residual block

2.4 MobileNetv2

MobileNetv2 [19] has been introduced as a significant improvement from MobileNet [12] and applies a very efficient combination of depthwise separable convolution, inverted residuals and linear bottlenecks, achieving an excellent balance between performance and accuracy while maintaining a remarkably simple architecture. It has been used as a feature extractor for semantic segmentation in DeepLabv3 [4], beating YOLOv2 [17] on the COCO-dataset [15] while being 20x more efficient and 10x smaller[4].

2.5 Modified Aligned Xception

The Xception Model [6] has shown promising results in image classification and has since been refined [5, 7] with deeper Xception, replacing all max-pooling operations with a set of depthwise-separable convolutions, batch normalization and ReLU activation. Being used as an encoder in

DeepLabv3+ it has been shown to significantly improve segmentation accuracy along object borders [5].

3 Methods

In this section we briefly review atrous convolution in the context of Atrous Spatial Pyramid Pooling[4]. We then discuss XceptionV2 [5] and MobileNetv2 [19] as part of the Encoder-Decoder Models introduced in DeepLab3 [4] and DeepLab3+ [5] and their implementation as backbone in our proposed architecture, followed by the introduction of extended residual blocks and their application as well as a novel loss function for depth estimation employed by our model.

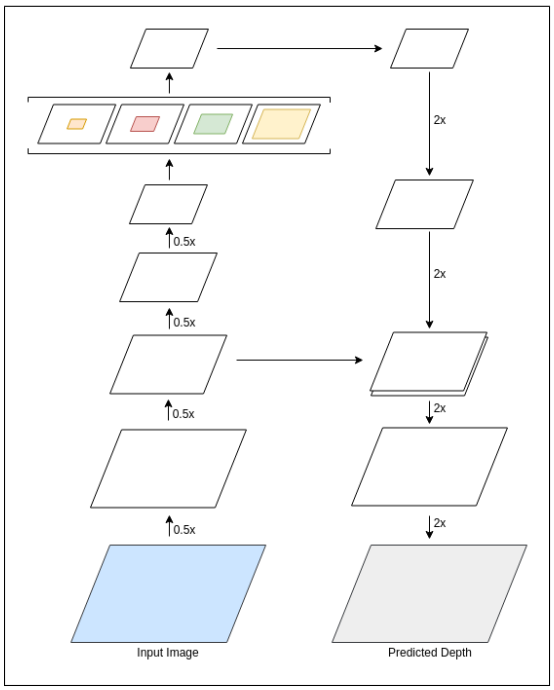


Figure 2: DeepLabv3+

3.1 Atrous Spatial Pyramid Pooling

Atrous convolution is a powerful method by which the field of view of a particular filter can be adjusted in order to capture multi-scale information, resulting in much better generalization with regards to standard convolution [5]. The atrous rate r determines the stride with which the input is sampled, such that for a two-dimensional signal each location i of the output feature map y and a convolution filter w is computed from the input feature map as follows:

$$y[i] = \sum_k x[i + r \cdot k]w[k] \qquad (1)$$

Thus the standard convolution operation is just an atrous convolution where the rate $r = 1$. In our implementation we use an Atrous Pyramid Pooling architecture introduced by Chen et al. [5] as part of the encoder (shown in Figures 3 and 4) where atrous convolution is applied at multiple scales and concatenated with a depthwise separable convolution and a pooling layer. To reduce the number of filters and thus save on computation we feed the concatenated outputs into a depthwise separable convolution layer whose output is then passed on to the CND-Decoder.

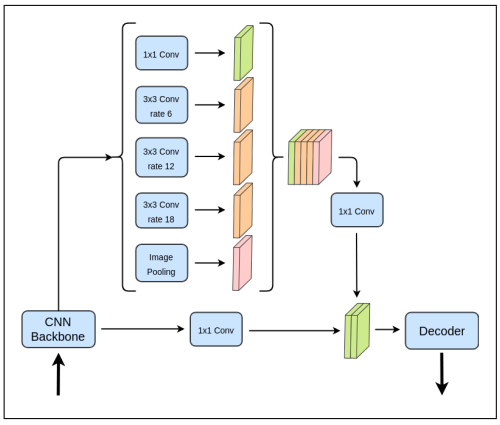


Figure 3: Atrous Spatial Pyramid Pooling

3.2 Encoder Architecture

Since depth estimation is mostly interesting for applications that are also time-sensitive or even performed in real time, our goal for the proposed CND-Architecture was to aim for state of the art accuracy, while also being as efficient as possible. Therefore we chose MobileNetv2 [19] with its high accuracy, yet low number of operations as our encoder backbone. We feed the model with 256x160 images that we do not preprocess apart from rescaling and horizontal flipping during training. While MobileNetv2 offers parameters to scale its size and performance to the users needs, we keep them at their default values. For the stereo models we share the weights between left and right encoder branch to save on memory which allows us to train all of our model on a single Nvidia GTX1080 at a batchsize of 16. After the MobileNetv2 stack reduces the filter size to 16 times smaller than the original size of the image at a depth of 256, we apply the ASPP block to extract the encoded features at multiple resolutions. In detail it consists of three 3x3 atrous convolution layers with dilation rates of 6, 12 and 18, that are

put in parallel to a depthwise separable convolution and a max-pooling layer. All of them are then batch-normalized and concatenated before we apply a depthwise separable convolution to reduce the filter depth to 256. This output is then concatenated with a layer that has skipped the ASPP block and has been fed through a depthwise separable convolution to the stack and is then again reduced to a filtersize of 256 through a depthwise separable convolution, before being passed on to the decoder.

3.3 Decoder Architecture

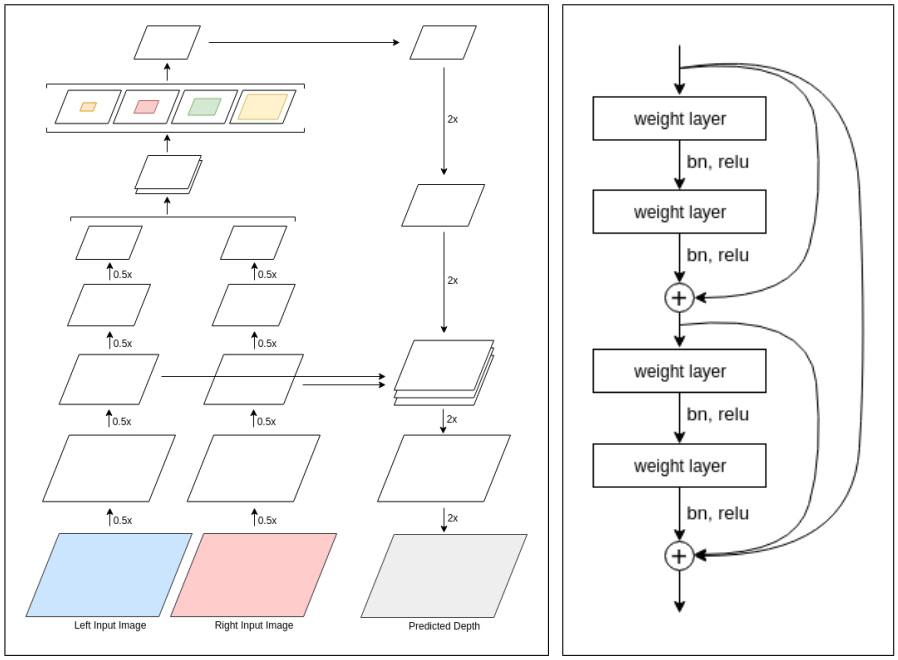


Figure 4: General CNDstereo architecture (left), Residual Decoder Module (right)

While other publications [3, 5, 16] use decoders applying basic stacks of Upsampling-Layers with factor 16, 4 or 2, we explore a more complex approach where every one of the four 2x upsampling operations is followed by a 3x3 transpose convolution layer with the same spatial resolution, a batch-normalization and a ReLU-activation layer. This approach has been applied in XceptionV2[5] in an attempt to eliminate all pooling operations. Pooling is typically used to achieve spatial invariance, which is undesirable in cases like segmentation or depth perception, where spatial awareness is essential. In our most basic encoder-architecture called CNDmicro, which is intended for low latency or mobile applications, this block is repeated four times, gradually reducing the filter depth to 16, and concluded with a final 2x upsampling and a consecutive 7x7 convolution layer with stride 2 that outputs the final 256x160 depth map. Additionaly, in order to refine detail around object borders we concatenate a skip-connection after the second reduction block of the encoder, and concatenate it with the corresponding decoder layer of the same size after feeding it through a depthwise separable convolution. For the purpose of further improving the quality of the results we introduce an extended residual block architecture (Figure 4), based on the successful residual block by He et al. [11]. In our implementation

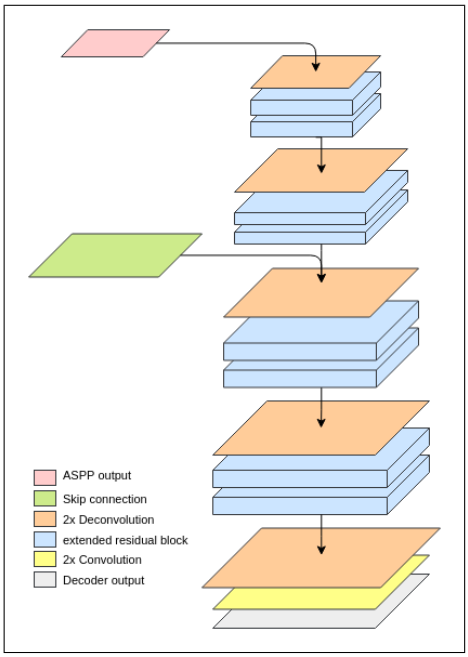


Figure 5: CND-Decoder Architecture

we create a residual connection, that internally consists of m blocks who in turn contains n weight layer blocks. These are formed by a transpose convolutional layer followed by a batch normalization and a ReLU layer. For the final implementation of CNDmono and CNDstereo we used two of these extended residual blocks with $m = n = 2$ between the upsampling blocks.

3.4 Extended Depth Loss

After experimenting with a selection of successful loss-functions that have been used in the context of depth estimation, we chose to expand on the extended scale invariant log RMSE introduced by Mancini et al. [16]:

$$L_{depth} = \frac{1}{n}\sum_i d_i^2 - \frac{1}{2n^2}\Big(\sum_i d_i\Big)^2 + \frac{1}{n}\sum_i [\nabla_x D_i + \nabla_y D_i] \cdot N_i^* \tag{2}$$

where $d_i = \log D_i - \log D_i^*$, D_i and D_i^* are the predicted depth and the ground truth depth at pixel i respectively. $\nabla_x D_i$ and $\nabla_y D_i$ are the horizontal and vertical predicted depth gradients and N_i^* is the ground truth 3D surface normal. While the first two terms correspond to the scale invariant log RMSE loss introduced in [8], the third term was introduced to enforce the orthogonality between predicted gradients and ground truth normals, aiming to preserve geometric coherence by Mancini et al. [16]. We propose to augment this loss in the following way, aiming to improve the detailed reconstruction of sharp edges and foreground objects.

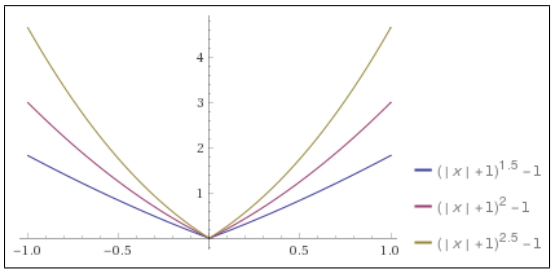


Figure 6: Extended loss behavior assuming different values for φ

$$L_{ext} = (|L_{depth}| + 1)^{\varphi} - 1 \tag{3}$$

Hereby we introduce the parameter φ that can be tuned to achieve the desired balance between sharp edges and smooth surfaces. We achieved notable improvements by setting $\varphi = 2$ without extensive optimization. The behavior of L_{ext}, assuming different values for φ can be seen in Figure 6.

4 Experimental Evaluation

In the following section we present results obtained by several versions of our proposed architecture. All results have been obtained through end-to-end training without pretraining or postprocessing. Training and testing of our TensorFlow-based[1] implementation has been performed on a single Nvidia GTX1080 for models with a batchsize of 16, and on a cluster of 8 GTX1080s for models with a batch size of 128. We also present results on a 24k image subset of the recent ApolloScape dataset[13]. We report the following error measures that have been extensively used. Denote y as true depth, $\hat{y}$ as predicted depth and T as the set of all points in the image.

- Mean relative error (rel): $\frac{1}{T}\sum_i^T \frac{|\hat{y}_i - y_i|}{|y_i|}$
- Mean $\log_{10}$ error ($\log_{10}$): $\frac{1}{T}\sum_i^T |\log_{10}\hat{y}_i - log_{10} y_i|$
- Root mean squared error (RMSE): $\sqrt{\frac{1}{T}\sum_i^T |\hat{y}_i - y_i|^2}$

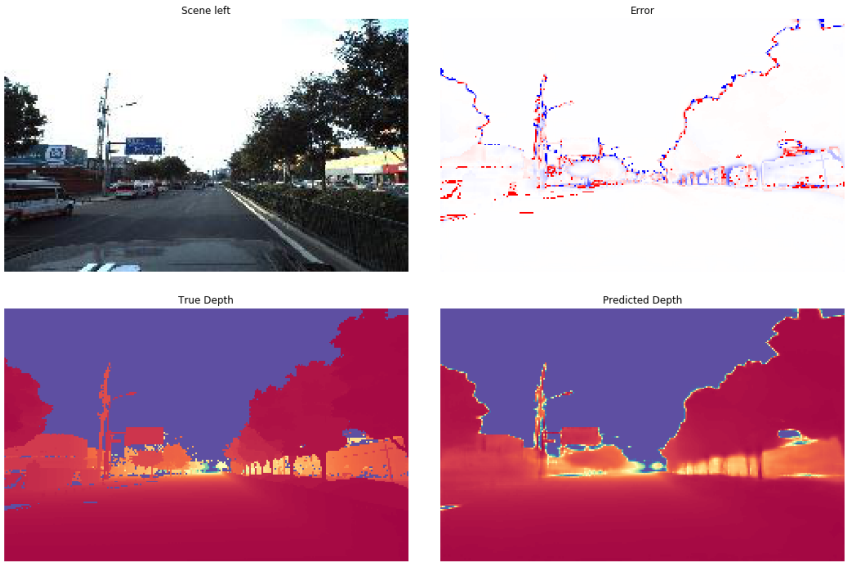
Table 2: Prediction times for the proposed models.

Name	Description	# of params	Prediction time
CNDmicro	mono, no residual blocks	5.8M	$13.8ms \pm 258\mu s$
CNDmono	mono, 2x2 residual blocks	6.7M	$15.3ms \pm 435\mu s$
CNDstereo	stereo, 2x2 residual blocks	11.1M	$27.5ms \pm 318\mu s$

Table 1: Model results for a 24k image subset of the ApolloScape Dataset[13]

Name	Description	Batch Size	rel	$\log_{10}$	RMSE
CNDmicro	mono, no residual blocks	16	0.15989	0.04315	0.02247
CNDmono	mono, 2x2 residual blocks	16	0.15509	0.04112	0.02147
CNDstereo	stereo, 2x2 residual blocks	16	0.14820	0.04105	0.02024
CNDstereo	stereo, 2x2 residual blocks	128	0.14297	0.03899	0.01948

Results on other common datasets like KITTI will be published soon. To date, to the best of our knowledge, no depth estimation results have been published on the recent ApolloScape dataset, thus our results can be regarded as a first baseline. We would like to note however the CNDmicro model achieves notable results despite its very low number of parameters and low execution time. CNDmono substantially improves on the CNDmicro accuracy through the newly introduced extended residual blocks without big changes in performance or parameter count. CNDstereo justifies its increase in execution time through a further substantial increase in accuracy, that can be further improved by training it with greater batch sizes. In the following some example results for CNDmicro and CNDstereo will be shown:



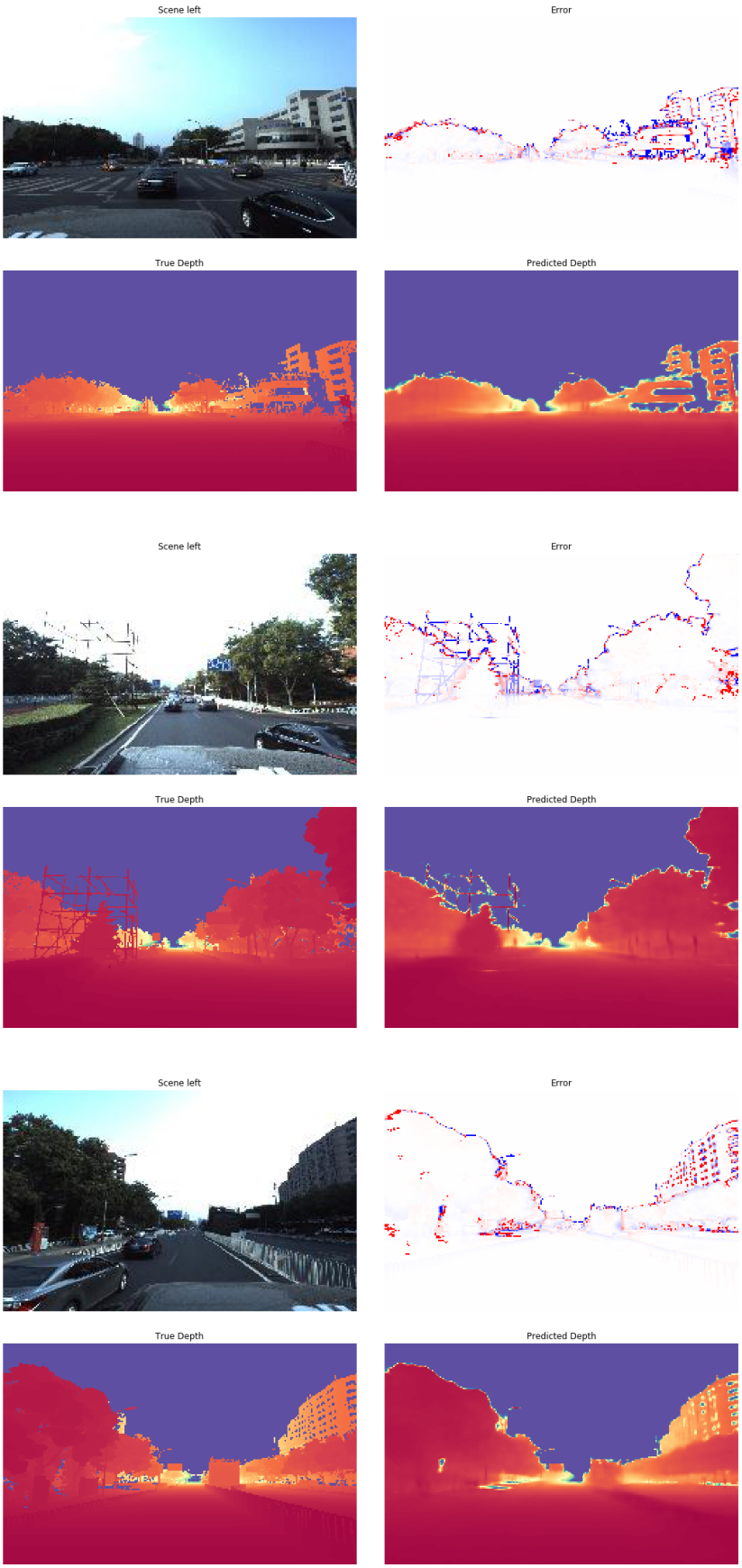
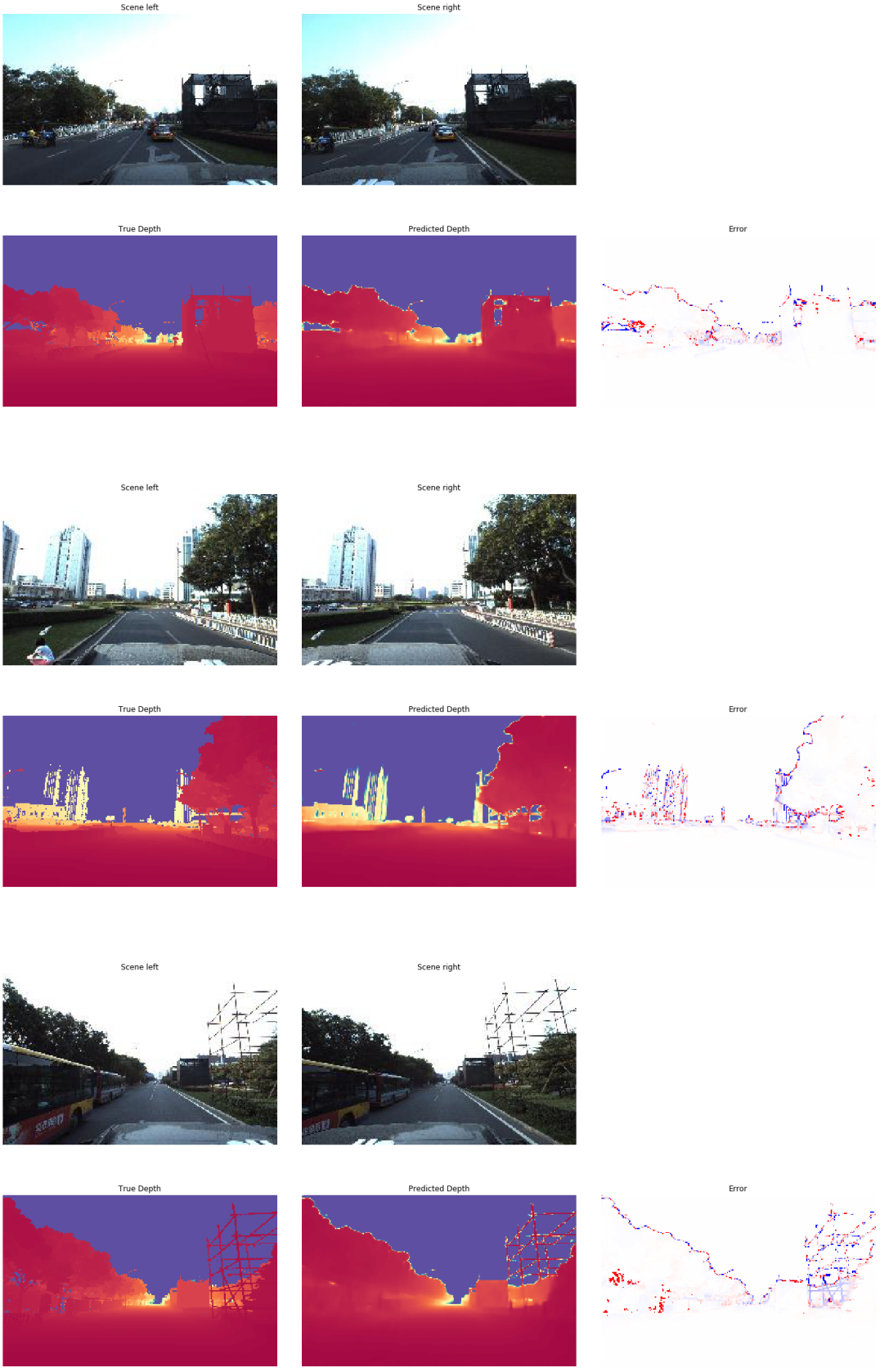


Figure 7: CNDmono depth map estimation results


Scene left
Scene right
True Depth
Predicted Depth
Error
Scene left
Scene right
True Depth
Predicted Depth
Error
Scene left
Scene right
True Depth
Predicted Depth
Error

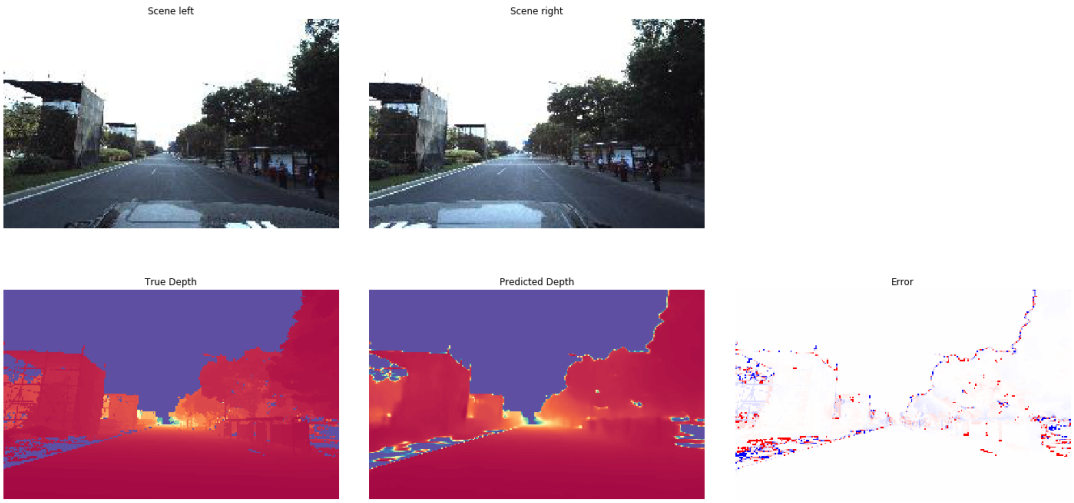


Figure 8: CNDstereo depth map estimation results

5 Conclusion

In this work we propose an efficient and scalable architecture for monocular and binocular depth estimation. It aims to refine detail reproduction around object borders using atrous convolutions, skip connections and a new residual decoder module. We employ an advanced loss function to further improve our results and achieve remarkably detailed and visually accurate results. Further work will be put into achieving results for the most important benchmarks in depth estimation as well as hyperparameter exploration on our current architecture followed by exploring more computationally challenging encoder-backbones such as XceptionV2.

Acknowledgments

We would like to thank Prof. Tammam Tillo for many helpful discussions and tips. Furthermore Esther Schaiter and Matteo Nardini were essential for the creation of this paper with their support and helpful comments.

References

[1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. (2016). Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283.

[2] Blake, A. and Bulthoff, H. (1991). Shape from specularities: Computation and psychophysics. *Philos. Trans. R. Soc. Lond. B Biol. Sci.*, 331(1260):237–252.

[3] Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2018a). DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(4):834–848.

[4] Chen, L.-C., Papandreou, G., Schroff, F., and Adam, H. (2017). Rethinking atrous convolution for semantic image segmentation.

[5] Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., and Adam, H. (2018b). Encoder-Decoder with atrous separable convolution for semantic image segmentation.

[6] Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[7] Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., and Wei, Y. (2017). Deformable convolutional networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*.

[8] Eigen, D., Puhrsch, C., and Fergus, R. (2014). Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374.

[9] Fu, H., Gong, M., Wang, C., and Tao, D. (2017). A compromise principle in deep monocular depth estimation.

[10] Godard, C., Aodha, O. M., and Brostow, G. J. (2017). Unsupervised monocular depth estimation with Left-Right consistency. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[11] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

[12] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). MobileNets: Efficient convolutional neural networks for mobile vision applications.

[13] Huang, X., Cheng, X., Geng, Q., Cao, B., Zhou, D., Wang, P., Lin, Y., and Yang, R. (2018). The apolloscape dataset for autonomous driving. *arXiv: 1803.06184*.

[14] Jammal, S., Tillo, T., and Xiao, J. (2017). Multi-resolution for disparity estimation with convolutional neural networks. In *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*.

[15] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In *Computer Vision – ECCV 2014*, pages 740–755. Springer International Publishing.

[16] Mancini, M., Costante, G., Valigi, P., and Ciarfuglia, T. A. (2017). J-mod$ˆ{2}$: Joint monocular obstacle detection and depth estimation. *CoRR*, abs/1709.08480.

[17] Redmon, J. and Farhadi, A. (2017). YOLO9000: Better, faster, stronger. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[18] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. In *Lecture Notes in Computer Science*, pages 234–241.

[19] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and others (2018). Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. *arXiv preprint arXiv*.

[20] Wang, P., Chen, P., Yuan, Y., Liu, D., Huang, Z., Hou, X., and Cottrell, G. (2017). Understanding convolution for semantic segmentation.

[21] Zhou, T., Brown, M., Snavely, N., and Lowe, D. G. (2017). Unsupervised learning of depth and Ego-Motion from video. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.