Focus on defocus: bridging the synthetic to real domain gap for depth estimation — Supplementary Material —

Maxim Maximov
Technical University Munich

Kevin Galim
Technical University Munich

Laura Leal-Taixé
Technical University Munich

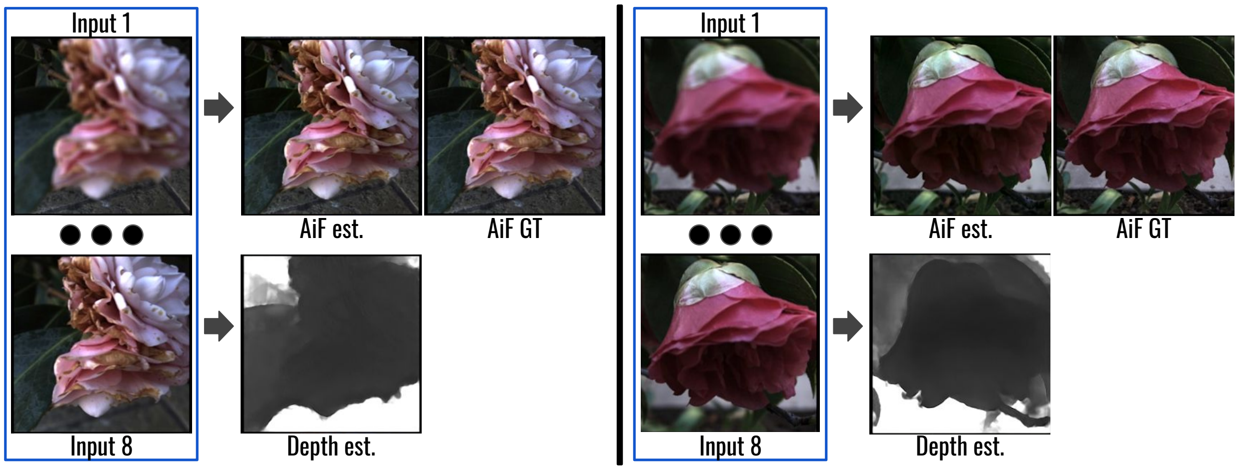

Figure 1: Two results of our approach on real data (Flower LF). Out input is a focal stack (left side on each panel) and the predicted outputs are depth and all-in-focus (AiF) images. For this dataset, we have only AiF ground-truth (GT) images but no depth ground-truth.

Abstract

In this supplemental document, we demonstrate qualitative results on dynamic stacks (Section 1), visually compare results of our method with monocular depth estimation ones (Section 2), show additional qualitative results on synthetic and real data (Section 3), describe the all-in-focus estimation method (Section 4), detail the architecture of our models (Section 5), and explain failure cases (Section 6).

1. Dynamic stacks

In this section, we show the capturing method with an android smartphone application and qualitative results on these data.

Real data capture. To capture images with constantly changing lens focus, we developed an android application which automatically changes the lens focus while capturing pictures. The advantage of using an android phone when recording a scene is that the focus distance can be set via software and the lens will automatically adjust itself. However, a drawback is that the camera usually possesses small lenses with a fixed focal length. This leads to a wide depth of field, meaning the range where objects appear sharp is very large.

We used a Samsung Galaxy S7 to capture dynamic focal stacks. Our application captures a limited number of frames in a burst manner with changing focus distance. The change in the focus distance is given by a linear function based on the current recording time as seen in Fig. 2. We sweep from the minimum to the maximum focus distance and capture 5 frames, then start again from the minimum focus distance. This removes the difficulty for the networks to recognize if the focus distances were increasing or decreasing.

Focus Breathing. Focus breathing or lens breathing is the slight change of the field of view when the focus distance changes. This effect is more severe when the focus distance changes by a very large distance, *e.g.* by focusing on a very close object and then focusing to the infinity. As a result, we have a slight magnification effect when decreasing the focus distance.

In what amount focus breathing is noticeable depends on the specific lens. For our approach, this might interfere with our depth estimation process since this distortion introduces additional change in the frames. However, our tests show that the CNNs are nevertheless able to produce proper predictions from focus-breathing contaminated data. Thus a special mechanism to remove the focus breathing is not necessary.

Qualitative results. In Fig. 3, we show depth prediction results of our method on real dynamic stacks. Dynamic stacks (with 4 frames each) were captured with our android application with slight change in camera position.

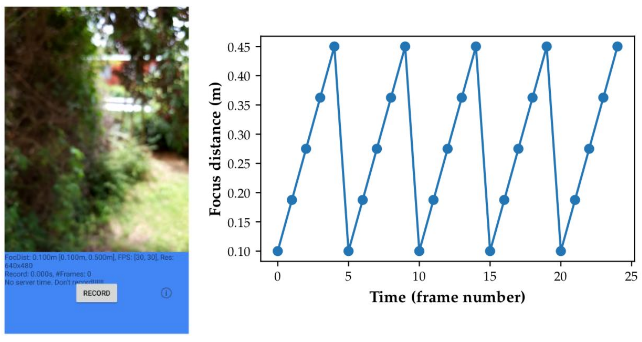
FocDist: 0.100m [0.100m, 0.500m], FPS: [30, 30], Res: 640x480
Record: 0.000s, #Frames: 0
No server time. Don't record!!!!!!

Figure 2: A screenshot of our android application on the left side. Focus distance change during capture on the right side.

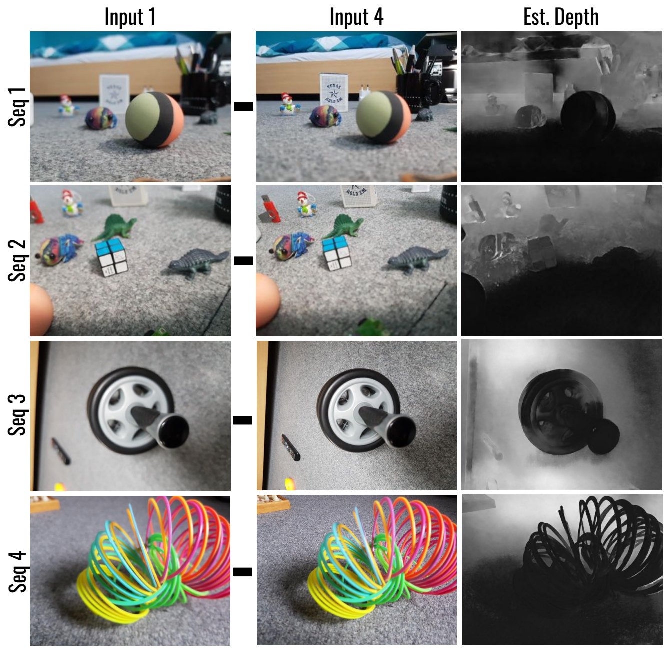

Figure 3: Estimated depth with our method on dynamic stacks with 4 inputs (only 2 are shown).

2. Comparison with a monocular depth estimation

In Fig. 5, we show results of our synthetically trained model on images from MobileDepth [8] dataset (first 3 columns) and on images taken with our consumer smartphone (last 3 columns). We capture a focal stack with only 2 images. We compare our method to recent state-of-the-art VNL [9] trained on indoor NYU Depth V2 dataset[6] and MegaDepth method [5] trained on outdoor MegaDepth dataset. We do not have ground truth depth available for those images, therefore we compare all method visually on the correctness of the relative depth.

In Fig. 5, we show that our method consistently estimates better relative depth, although it has never been trained on real images with real blur. Other methods have smoother depth predictions and perform well on certain images that rely more on perceptive cue (column 1 and 5), but they fail on harder scenarios. We especially can see the bias of these methods on column 6 where we capture an image from the top down.

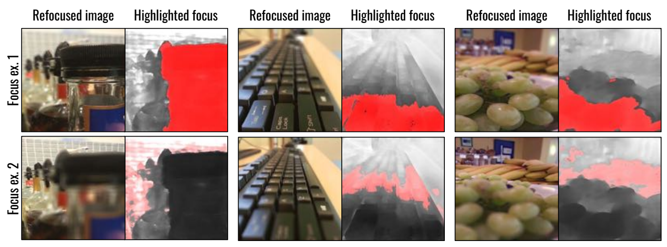

Figure 4: Refocus results based on estimated AiF and Depth images. In each panel, images on the left side show refocused images and images on the right side - depth map with highlighted in-focus areas.

3. Visual Results

In this section, we show our additional qualitative results on another set of images. In Fig. 1, we show AiF and depth estimation results on real data for our model *trained only on synthetic data*. In Fig. 6, 7 and 9, we show depth estimation on synthetic test data.Fig. 7 has inputs with a wide depth of field (DoF) and DefocusNet correctly estimates defocus maps, but larger DoF leads to a flawed depth map.

Defocus. Fig. 6, 7 and 9 show defocus on synthetic data. Fig. 8 shows results on real photos. In-focus areas shown darker in defocus maps. From these figures, we can see that DefocusNet works well even with unordered, arbitrary sized inputs. Note, in presented figures, the number of inputs varies from 2 to 10. Our model can also effectively deal with different image domains: real and synthetic.

Refocus. Fig. 4 shows examples of synthetically refocused images on two different focus distances. We use AiF and depth images that resulted from our joint model. We choose a depth segment that kept in-focus and sequentially blur everything else. We use a simple approach for refocusing: blur the whole image using convolution operations except the target in-focus area (highlighted in red). The amount of blur depends on distance from the camera and target in-focus depth.

Additionally, in Figure 10, we show another example of sequential defocus, depth and AiF refinement with the growing number of inputs.

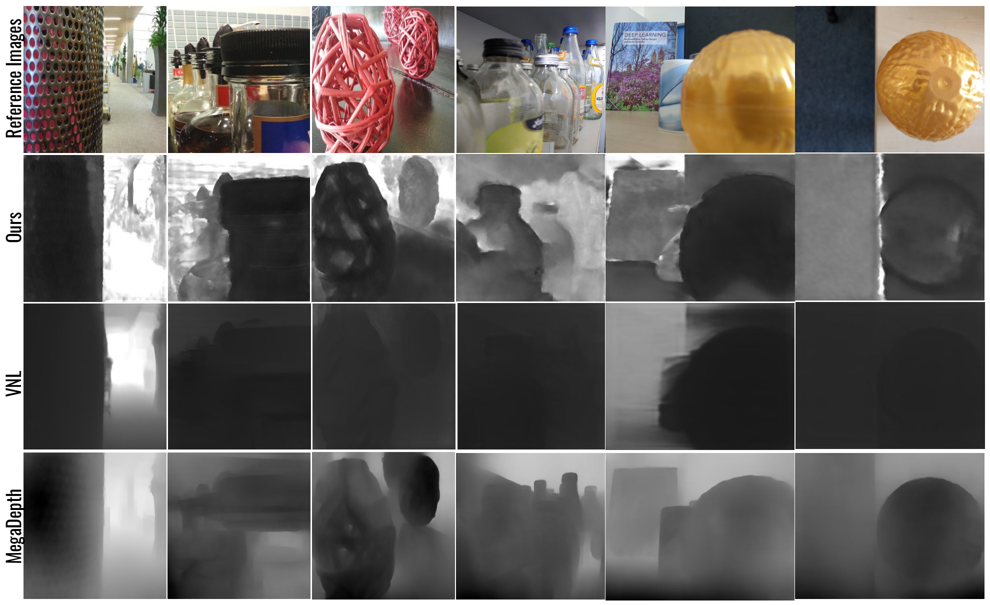

Figure 5: Depth estimation comparison on real data. 1st row: input images, 2nd row: our results on a focal stacks (with 2 images) , 3rd row: results of VNL [9], 4th row: results of Megadepth [5] - on single images. First 3 images are from MobileDepth [8] dataset and next 3 images are taken with a smartphone.

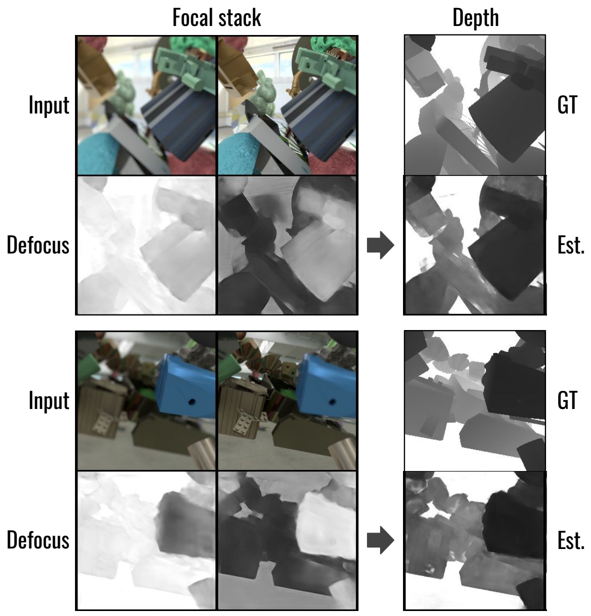

Figure 6: DefocusNet and DepthNet results on synthetic dataset with only two inputs, i.e., near and far focused.

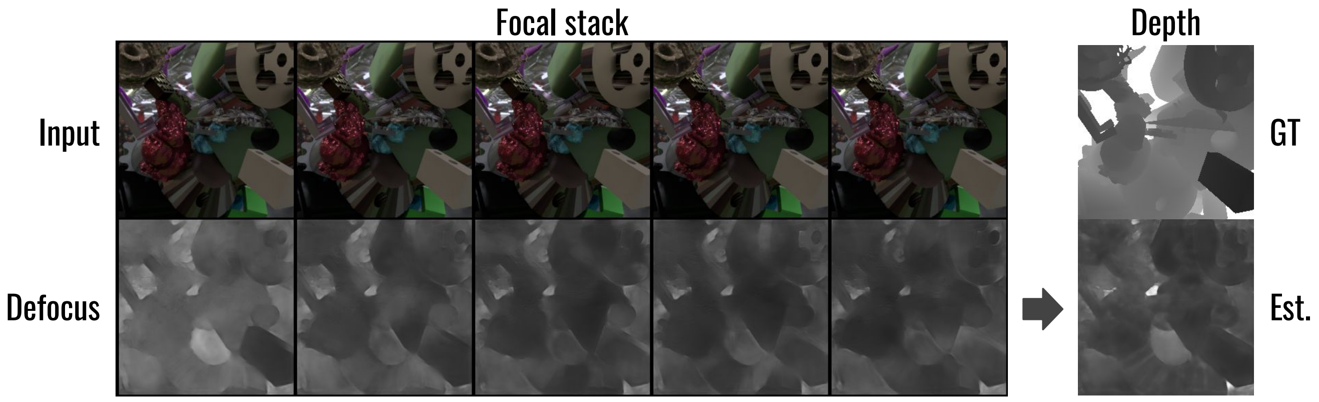

Figure 7: Defocus and depth estimation results on images with wide depth of fields. Most of the images are already sharp and our DefocusNet correctly estimates that (black pixel shows in-focus, white - out-of-focus). Depth estimation works a lot worse since defocus difference is small (unlike in Fig. 9) and training data has only narrow DoF images.

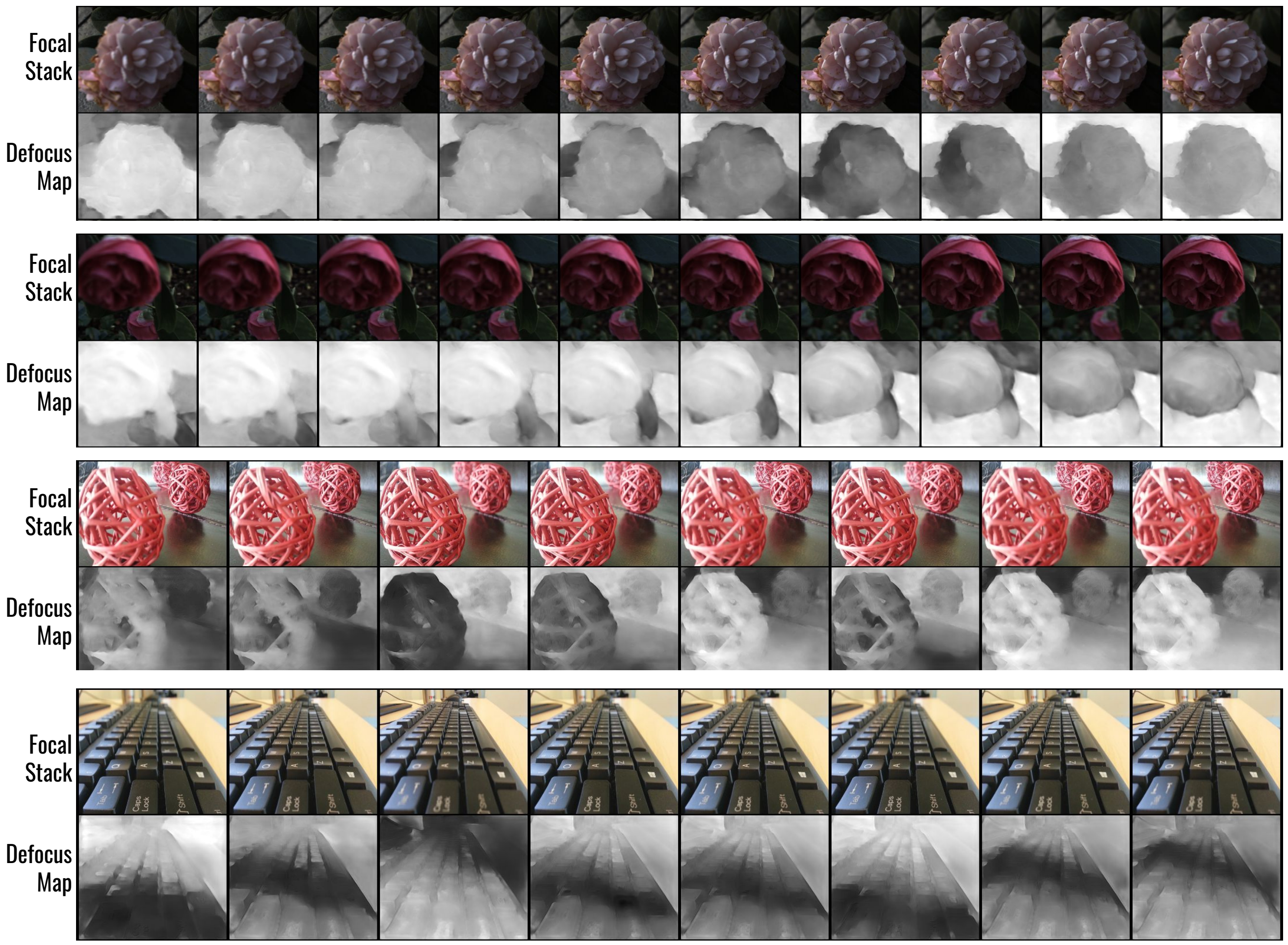

Figure 8: Defocus map estimation (the model is trained only on synthetic data). Darker pixels show regions with sharper details (more in-focus). Top rows show results on ordered input (1st and 2nd rows) in LF Flower data and bottom rows show results on unordered input (3rd and 4th rows) in Mobile Depth from Focus dataset.

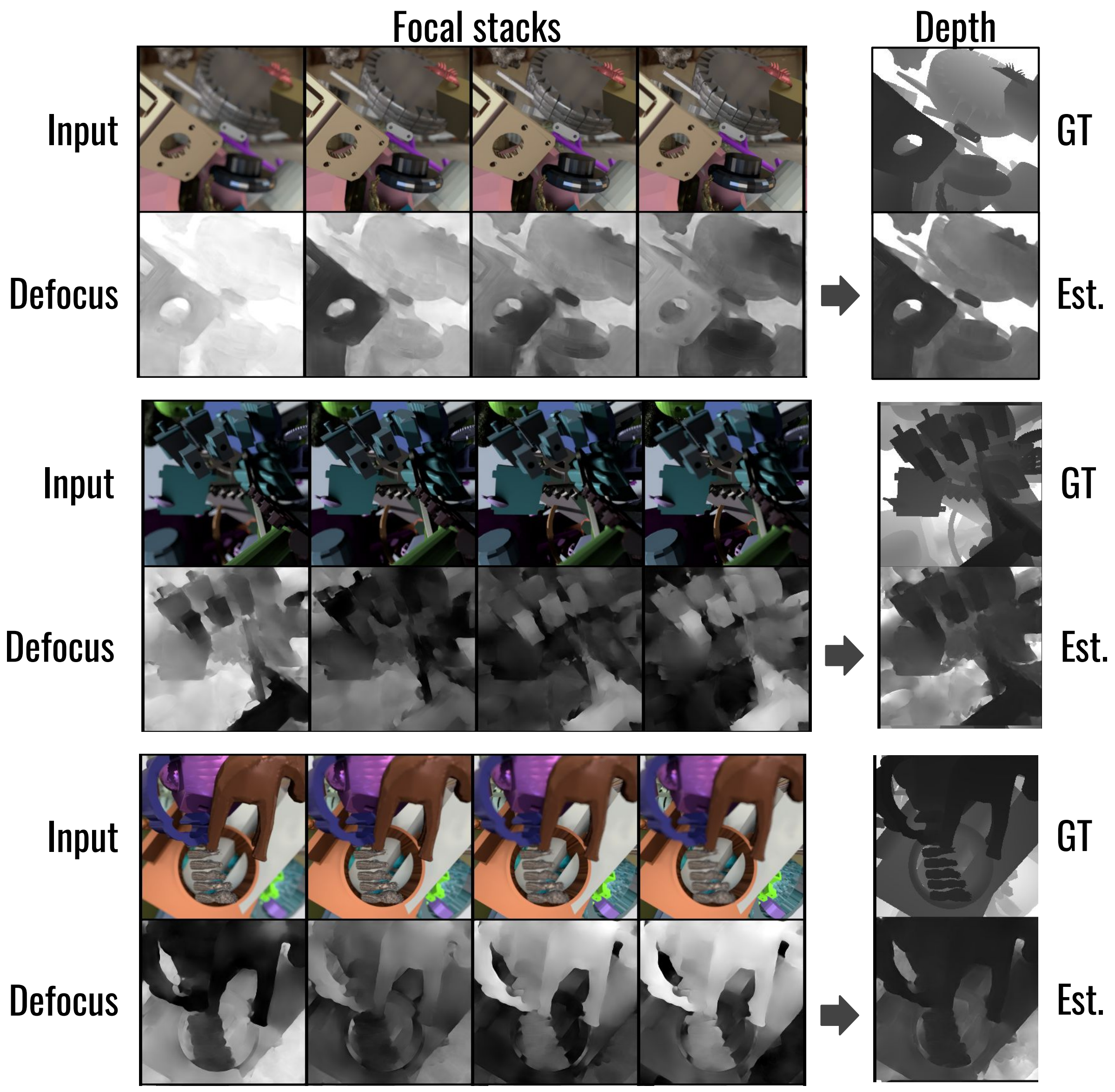

Figure 9: Defocus and depth estimation results on images with narrow depth of fields.

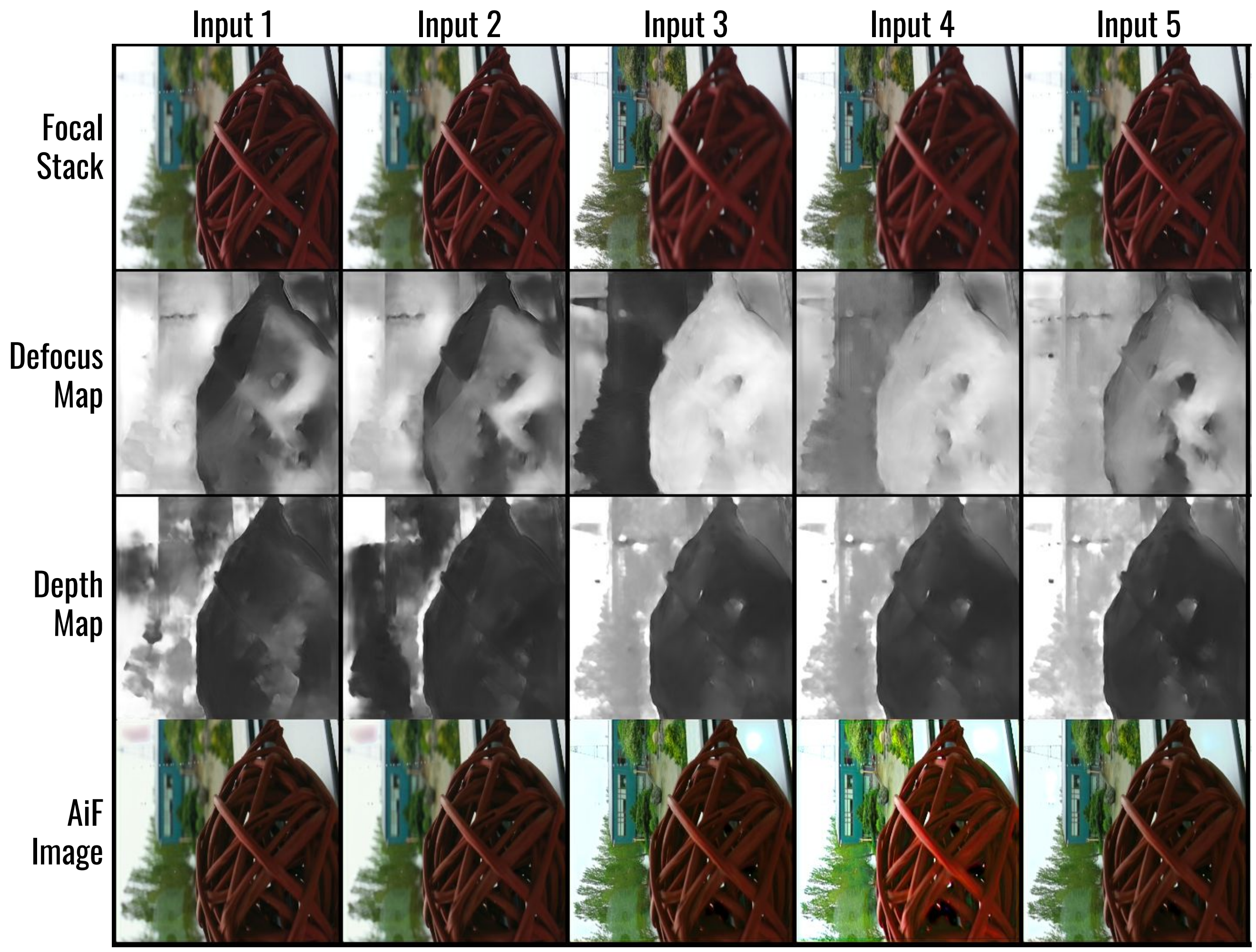

Figure 10: Our sequential estimates on a real focal stack. The first row are inputs to our pipeline. The other horizontal sequences show our outputs for a growing number of input frames. The results on the left use only the first input image and on the right use all five inputs. Note how adding inputs from different focus distances (inputs 2 and 3) improves depth and all-in-focus estimates.

4. All-in-Focus estimation

For image post-processing applications such as refocusing, additionally to depth map, we also need an all-in-focus (AiF) image where all pixels are appropriately sharp. Our model already estimates defocus maps that show image sharpness level given a focal stack. By combining different image parts from a focal stack and their corresponding defocus maps, we can estimate final all-in-focus image. For this reason, we propose to incorporate such estimation inside our network, reusing the focal stack and estimated defocus maps. The AiF image is computed by an additional CNN head, termed AiFNet (Fig. 11).

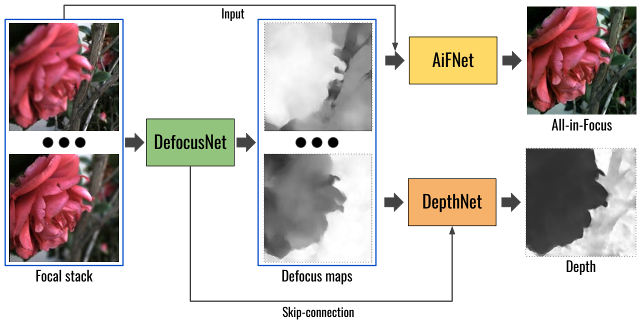

Figure 11: The pipeline of our approach.

4.1. Methodology

Similar to the described DefocusNet, AiFNet globally pools features from shared network branches, each of which processes an input image from the focal stack. The main difference in AifNet design is that we do not use an AE architecture, but rather a *refining* architecture. This consists of residual blocks that maintain the same resolution in between layers. This is inspired by super-resolution networks that have a similar architecture of residual blocks [4, 7]. The main goal of this architecture is to sharpen input images and combine all sharpest regions (indicated by a defocus map) in a single AiF image.

Additionally, to make training focus on sharp details, the AiFNet has a second output that regresses an edge map. The edge map is a binary single-channel image that indicates the presence of an edge. We pre-compute ground-truth edge maps with a Canny edge detector method [1] on all-in-focus images. We get it for free and it acts as additional supervision on sharp details.

For the training, we use L1 loss between estimated and ground-truth depth, perceptual loss on features[3] with pre-trained VGG and L1 loss on edge map.

$$\begin{aligned} Loss = \lambda_a \sum \|I_{defocus} - E_{defocus})\|_2 + \\ \lambda_c \sum \|I_{aif} - E_{aif}\|_1 + \\ \lambda_d \sum \|\Phi_{VGG}(I_{aif}) - \Phi_{VGG}(E_{aif})\|_1 + \\ \lambda_e \sum \|I_{edges} - E_{edges}\|_1, \end{aligned} \quad (1)$$

where $\lambda_{a,c,d,e}$ are weight coefficients and Φ_{VGG} represent a specific VGG-layer used to compute the perceptual loss.

4.2. Evaluation

Metrics. For all-in-focus comparison, we compare the structural similarity (SSIM) and the peak signal-to-noise ratio (PSNR) metrics. For SSIM and PSNR, higher value is better.

Datasets. We qualitatively evaluate our method on synthetic and real datasets.

(i) Synthetic dataset. Synthetic test data is the same as the synthetic test data in Section 5.2 of the main paper. We use all 4 test sets to evaluate generalization: Shape (new objects), Appearance (new materials and illuminations), Wide DoF (singinifanty bigger f-number) and Medium Dof tests (slightly bigger f-number).

(ii) Flower Light-Field (LF) dataset. This dataset was used in [7], but the test set has not been made publicly available for comparison. We used their original light-field dataset and we produced focal stacks using a toolbox from [2].

4.3. Ablation Study on Synthetic Dataset

We quantitatively analyze our method's generalization capabilities to new environments and camera settings on 4 different synthetic test modalities.

Architecture choice. From tests in Table 1, we can summarize that relying on defocus with a focal stack (row 4.) generalizes better than doing the direct estimation (row 3).

Is defocus needed? For all-in-focus estimation, see Table 1, our defocus approach on a focal stack (row 4.) shows a similar or better performance across all tests.

Single image vs. Focal stack. In Table. 1, we show the benefits of using a full focal stack for AiF prediction as opposed to using only a single image. Single image networks perform worse, as they are not able to rely on more information about sharp regions comparing to the focal stack.

4.4. Evaluation on Real data

Flower LF dataset. Table 2 shows the results of our method on Flowers dataset. We show the generalization capability of our model trained on synthetic data for all-in-focus estimation on real data, and compare it with a model trained directly on this real dataset.

Models	*Shape*		*Appearance*		*Wide DoF*		*Medium DoF*	
	All	Random	All	Random	All	Random	All	Random
1. Single RGB $\rightarrow$ AiF	0.935 / 82.4	-	0.940 / 82.2	-	0.963 / **84.4**	-	0.957 / 83.7	-
2. Single RGB $\rightarrow$ Defocus $\rightarrow$ AiF	0.927 / 81.2	-	0.935 / 81.1	-	0.953 / 82.5	-	0.950 / 82.1	-
3. FS $\rightarrow$ AiF	0.938 / 81.8	0.646 / 70.6	0.950 / 81.9	0.633 / 68.8	**0.967 / 84.4**	0.644 / 69.3	0.962 / **83.9**	0.642 / 69.0
4. FS $\rightarrow$ Defocus $\rightarrow$ AiF	**0.969 / 85.8**	**0.936 / 82.8**	**0.962 / 82.8**	**0.928 / 80.5**	0.965 / 82.6	**0.963 / 82.3**	**0.967** / 83.2	**0.955 / 82.2**

Table 1: Results on the synthetic data test sets for all-in-focus estimation models with focal stacks (FS) or single RGBs as input. All tests show SSIM/PSNR metrics values.

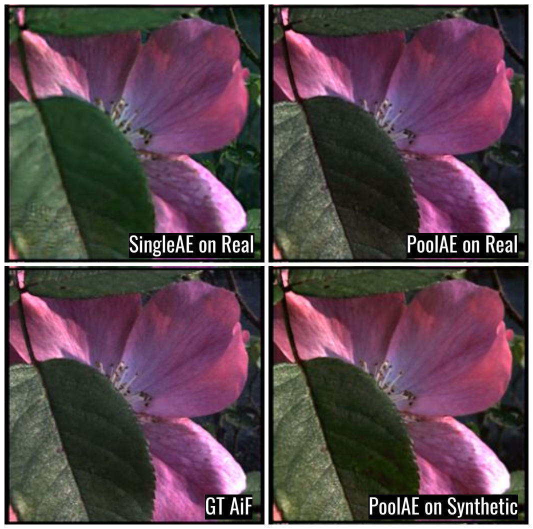

Figure 12: Qualitative all-in-focus results on Flower dataset. We show comparison between a joint architecture model trained on synthetic data (PoolAE on Synthetic) and direct approaches that are trained on real data.

We can see that our proposed model with focal stacks and defocus (row 7) perform better among models trained on synthetic data and close to models trained on real data. We also show qualitative results in Fig. 12. The main issues with synthetically trained models are that they produce AiF images with slightly different tone and slight artifacts on saturated areas as you can see in Fig.12. These issue lead to a small drop in quantitative results.

Single Image Models	SSIM	PSNR
RGB $\rightarrow$ AiF (S)	0.861	79.3
RGB $\rightarrow$ AiF (R)	**0.908**	**82.6**
RGB $\rightarrow$ Defocus $\rightarrow$ AiF (S)	0.858	78.2
RGB $\rightarrow$ Defocus $\rightarrow$ AiF (R)	0.904	82.3
Focal Stack Models	SSIM	PSNR
FS $\rightarrow$ AiF (S)	0.87.6	80.1
FS $\rightarrow$ AiF (R)	0.922	82.1
FS $\rightarrow$ Defocus $\rightarrow$ AiF (S)	0.899	78.9
FS $\rightarrow$ Defocus $\rightarrow$ AiF (R)	**0.938**	**82.4**

Table 2: Results of all-in-focus estimation on Flower LF dataset. (S) - trained on the synthetic data, (R) - trained on the real data

5. Network Architecture Details

The network architecture of the three inter-connected neural networks that make up our approach is detailed in Figures 13, 15 and 14. All models use the network design with global pooling across several inputs, but we show only one branch (input) since every branch uses an identical network with shared weights.

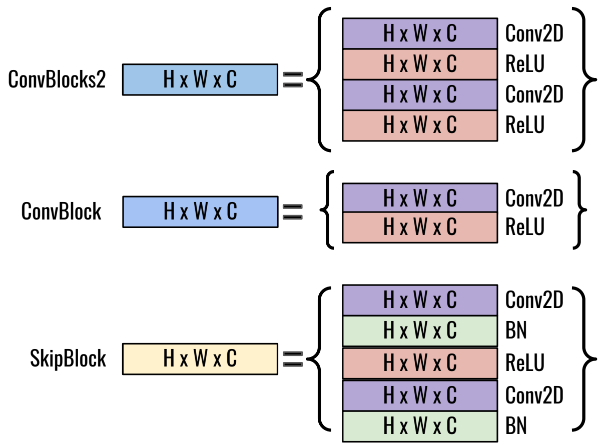

Figure 13: The architecture of several convolutional blocks mentioned in all networks, input is from the top side.

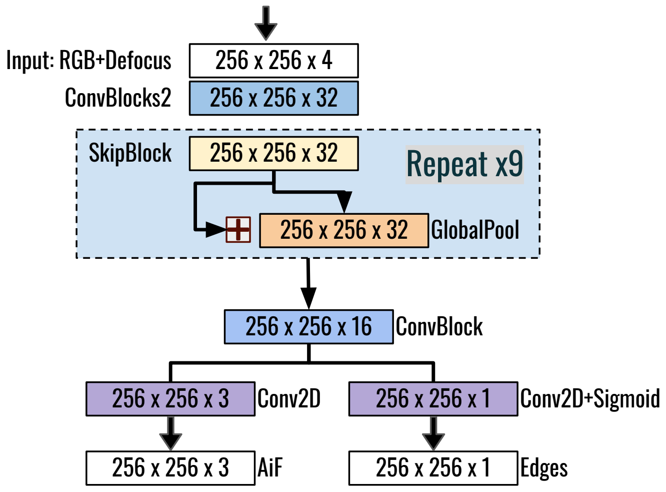

Figure 14: Our network architectures for one branch (input). The numbers in each box denote width$\times$height$\times$channels of the layer's output and a plus in box represents addition of feature maps. The architecture of AiFNet, which learns a color all-in-focus image from a color out-of-focus image and a corresponding grayscale defocus map. .

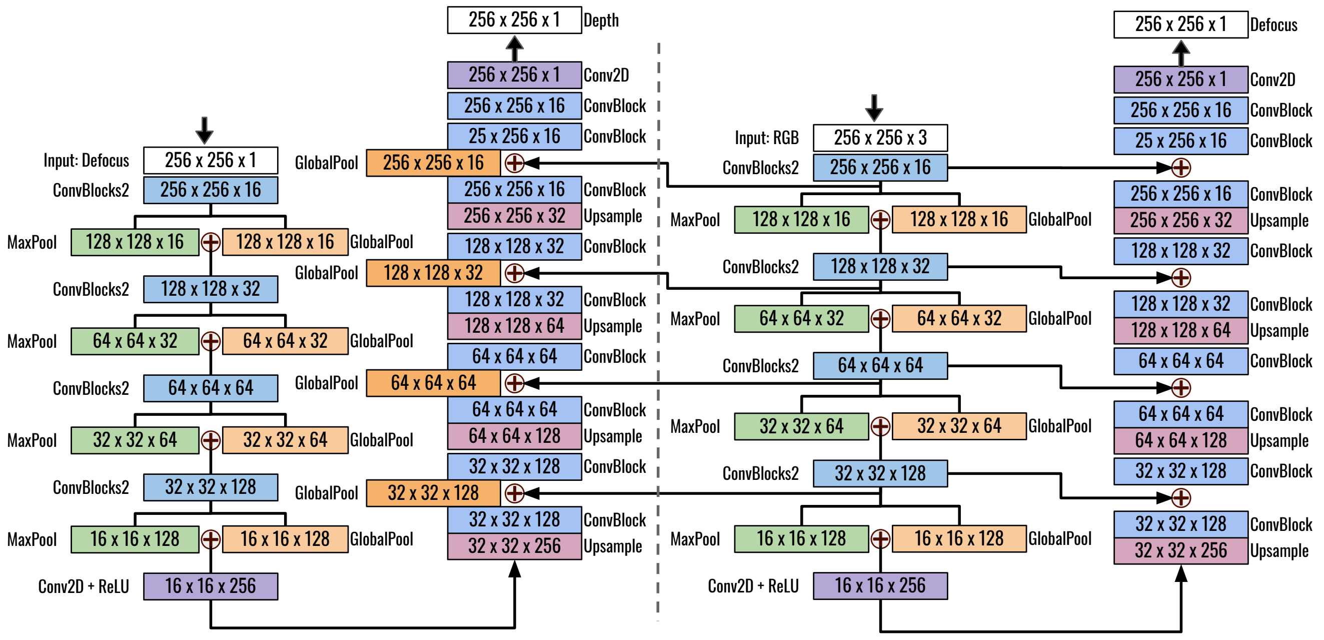

Figure 15: Our network architectures for one branch (input). The numbers in each box denote width×height×channels of the layer's output and a plus in circle represents concatenation of feature maps. **Right**: The architecture of DefocusNet, which learns a grayscale defocus map from a color input image. **Left**: The architecture of DepthNet, which learns a grayscale depth map from a grayscale defocus map. Note the skip-connections are all from the DefocusNet encoder.

6. Limitations

We show the main limitations in Fig. 16 and 17. If trained on synthetic data, in some examples, AiF model estimates have slightly different color tone. Such artifacts can be adjusted with post-processing techniques.

Depth estimation mainly struggles with large empty (texture-less) surfaces (e.g. walls, floors) and the depth gradient transition. Our synthetic data is rendered with a lot of random objects uniformly scattered around a scene, hence we lack scenes with large empty areas. So the solution is to complement training data with such scenes. Although, comparing to model-based approaches, our model correctly ignores small color variations on similar depth levels, e.g. depth map on the right side highlights the numbers on buttons which are supposed to have the same depth).

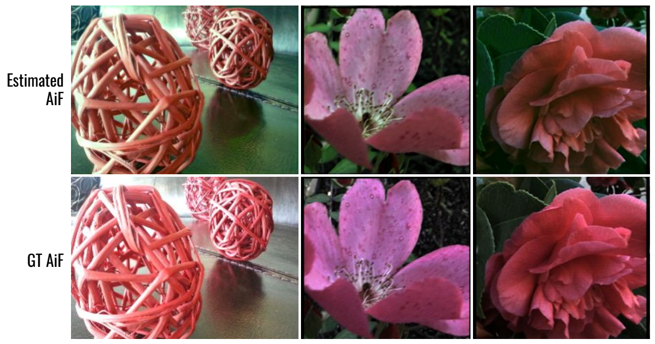

Figure 16: Our All-in-Focus (AiF) prediction model trained only on synthetic data creates, in some examples, a slight change in color tone in the output.

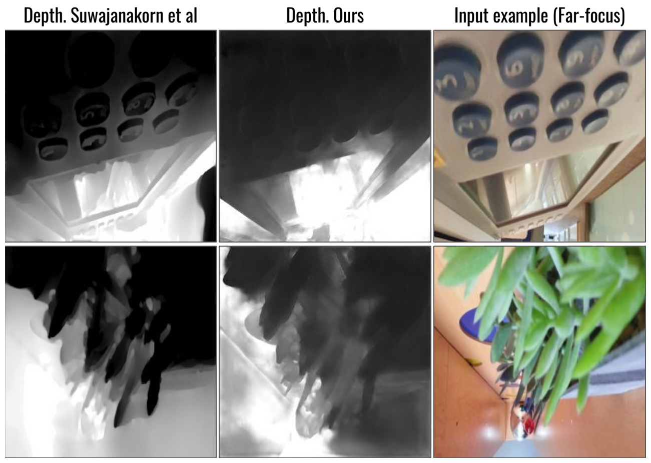

Figure 17: Failure examples for our depth estimation model (middle column) trained on synthetic dataset: large textureless (e.g. walls) and mirror surfaces. Right column shows one of the input images. Note, these are common failure cases, especially for data-driven models.

References

[1] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986. 7

[2] C. Hazirbas, S. G. Soyer, M. C. Staab, L. Leal-Taixé, and D. Cremers. Deep depth from focus. In *Asian Conference on Computer Vision (ACCV)*, December 2018. 7

[3] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II*, pages 694–711, 2016. 7

[4] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 105–114, 2017. 7

[5] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 3

[6] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 2

[7] Parikshit Sakurikar, Ishit Mehta, Vineeth N. Balasubramanian, and P. J. Narayanan. RefocusGAN: Scene refocusing using a single image. In *Computer Vision – ECCV 2018*, pages 519–535. Springer International Publishing, 2018. 7

[8] Supasorn Suwajanakorn, Carlos Hernandez, and Steven M. Seitz. Depth from focus with your mobile phone. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2015. 2, 3

[9] Wei Yin, Yifan Liu, Chunhua Shen, and Youliang Yan. Enforcing geometric constraints of virtual normal for depth prediction. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 2, 3