

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

Maxim Maximov

Technical University Munich

Kevin Galim

Technical University Munich

Laura Leal-Taixé

Technical University Munich

Abstract

Data-driven depth estimation methods struggle with the generalization outside their training scenes due to the immense variability of the real-world scenes. This problem can be partially addressed by utilising synthetically generated images, but closing the synthetic-real domain gap is far from trivial. In this paper, we tackle this issue by using domain invariant defocus blur as direct supervision. We leverage defocus cues by using a permutation invariant convolutional neural network that encourages the network to learn from the differences between images with a different point of focus. Our proposed network uses the defocus map as an intermediate supervisory signal. We are able to train our model completely on synthetic data and directly apply it to a wide range of real-world images. We evaluate our model on synthetic and real datasets, showing compelling generalization results and state-of-the-art depth prediction. The dataset and code are available at <https://github.com/dvl-tum/defocus-net>.

1. Introduction

In recent years, we have seen an increase in the number of smartphone photography users, bringing the need for image editing tools to a wider audience. Most of these tools are still limited to color adjustments and simple image transformations. More advanced post-capture changes such as focus and depth-of-field adjustments are not commonly available due to the need for depth maps of the captured scene. While there exist specialized hardware solutions to compute depth, *e.g.*, light-field cameras [14], nowadays, data-driven machine learning makes it possible to tackle the problem from the software side, predicting depth maps [9, 7, 36] from a single image. However, monocular depth estimation methods do not generalize well to unseen data/scenes, *e.g.*, different viewing angles, geometry and objects types. They heavily rely on perspective, size, texture and shading cues to measure distance. Those cues are dependent on the type of scene and objects, texture and illumination, which easily leads to overfitting to those memorized settings [20]. Other works on depth estimation show

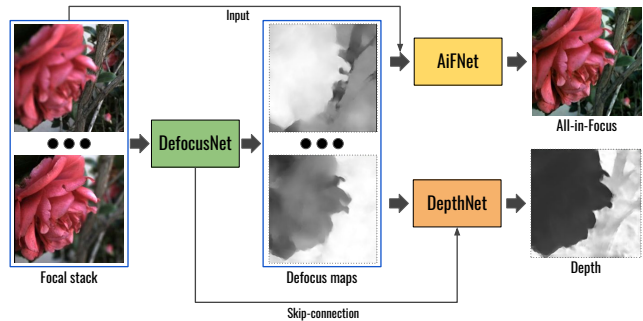


Figure 1. The pipeline of our approach. Our proposed end-to-end learned model combines depth and all-in-focus estimation from a focal stack using intermediate defocus map estimation and permutation-invariant networks, leading to a better generalization from a synthetic training to real photos.

better generalization by relying on comparison-based depth cues, such as depth from motion [34, 19] or stereo [37, 29].

An under-explored cue for depth estimation is *defocus*, given that an object’s depth dictates how sharp it will appear in the image. Depth-from-focus (or defocus) is defined as the task of obtaining the depth of a scene from a focal stack, *i.e.*, a set of images taken by the same camera but focused on different focal planes. Analytical approaches [21, 33] compute depth based on the sharpness of each pixel. Such approaches are time-consuming, and perform especially poorly for texture-less objects. Recent deep learning approaches address these challenges in a data-driven way by learning to directly regress depth from a focal stack [12]. Their main drawback is that they do not consider the underlying image formation process, therefore, such methods are also prone to overfitting to the specific training conditions.

Another challenge towards achieving generalization is the lack of high-quality and diverse training sets. Collecting focal stack images with registered depth maps is an extremely time-consuming task, not to mention the imperfect depth ground truth data obtained from hardware solutions like time-of-flight sensors [12]. One can rely on synthetic data as used in inverse-graphics tasks [31, 17, 18, 9], but not without addressing the problem of bridging the domain gap

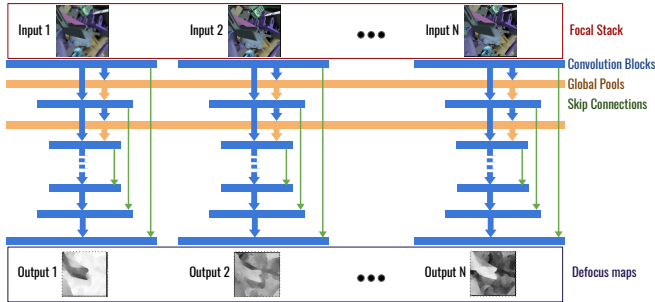


Figure 2. DefocusNet architecture. The proposed architecture takes a focal stack with an arbitrary size as an input and estimates corresponding defocus maps. The network uses an autoencoder as a basis and shares weights across all branches. Global pooling is used as a communication tool between separate branches.

between synthetic and real images [25].

Defocus blur is a well-modeled physical effect, and as such, straightforward to simulate in a realistic way. The main insight of our work is that, while appearance features greatly differ from synthetic to real images, blur does not. Such domain invariant measurement effectively aids in bridging the domain gap between synthetic and real data. We therefore propose to leverage defocus in a data-driven model to predict depth from focal stacks. By breaking the depth prediction into two steps, and using defocus maps as intermediate representations, we can train a neural network that generalizes from synthetic to real data without fine-tuning. Additionally, we show our architecture works for an arbitrary number of input images and propose an extension to dynamic focal stacks [15], where camera motion or scene motion is present. Our **contribution** is three-fold:

- We propose to use defocus blur as intermediate supervision to train data-driven models to predict depth from focal stacks. We show that this is key towards generalization from synthetic to real images, and show state-of-the-art results.
- We generate a new synthetic dataset with multiple objects, textures and varying illumination with depth, blur and all-in-focus information.
- We propose architectures for static scenes that can work with a varying number of inputs. In dynamic focal stacks, our model can handle scene or camera motions within the stack. We show their robustness in a comprehensive ablation study.

2. Related work

Depth estimation from defocus. Depth estimation is a popular topic and is being explored from multiple directions. The vast majority of work focuses on monocular

[9, 7, 36] or stereo [37, 29] depth estimation. Using defocus information for depth prediction is less common. Several optimization-based works [33, 32] estimate depth from a focal stack, while [15] extends such methods to videos. These are general approaches that work on a variety of scenes, though they struggle on texture-less surfaces, and produce compelling depth measurements, but they are highly time-consuming and require careful calibration. It takes up to minutes for these methods to estimate the sharpness of the image regions and compose a depth image. Recent methods leverage deep learning to bring this process closer to real-time. [12] uses convolutional neural networks (CNNs) to estimate depth directly from input focal stacks, without considering the underlying image formation process. Such a method is bound to have generalization problems unless train and test conditions are very similar. Additionally, it can only take a pre-defined number of inputs and does not incorporate any distance measurement.

Other works [2, 30, 5] implicitly use defocus information for monocular depth estimation. [30] proposes to use defocus as a part of the loss function to estimate depth from an all-in-focus image. Nonetheless, they still use an all-in-focus monocular image as input, hence they do not leverage defocus blur during inference. Similarly, [11] uses a differentiable loss layer that uses focus as a cue for depth prediction. [2] uses CNNs for image deblurring and depth estimation from a single out-of-focus image. [5] uses out-of-focus images for direct depth estimation. Their findings indicate that training on defocus images gives better depth measurements than training on sharp in-focus images. These methods use single images as input, therefore failing to leverage the much richer focus information present in a focal stack. As a consequence, they face difficulties when predicting depth in the wild, *i.e.*, for completely different scenes and/or cameras than the ones used at training time. In contrast, we propose to combine the power of data-driven approaches with knowledge of the image formation process, so that depth estimation can be computed by relying on focus differences between images in a focal stack.

Synthetic-to-real. There are several previous works that show domain generalization from synthetic to real data. According to [31], all of them can be divided into three main strategies: domain adaptation, photo-realistic rendering, and domain randomization. Domain adaptation approaches usually convert samples from one domain to another [28, 38], or use model fine-tuning on real data after training on synthetic [10]. Several works show compelling results on photo-realistic synthetic training and real test sets [17, 18]. However, photo-realism consumes a lot of time with physically based rendering (PBR) computation and hand-modelling of the entire environment. Domain randomization, similar to data augmentation, introduces a variety of random attributes to make a model invariant to small

changes and force the network to focus on the main features of the image. It requires simplistic modeling by assuming a random environment while being able to incorporate real data, *e.g.*, in the background. The main issue is to select the extent and type of attribute randomization. Several works show promising results in this direction [31, 1].

We believe there is a fourth strategy for domain generalization, *i.e.*, domain invariance, which involves training a model on features that are invariant for the two domains. One good example is stereo depth estimation. Models trained on synthetic data [9] are successful since they focus on the difference between two input images rather than the appearance and shape of objects. This approach can also be embedded into a network architecture [1], where a permutation invariant architecture is proposed in an image translation context. It combines information across a random unordered number of input images.

In our work, we rely on a domain invariant cue, defocus, as the main strategy for bridging the synthetic and real domain. Additionally, we also use domain randomization, and, to some extent, a photo-realistic rendering approach.

3. Learning depth through defocus

In this section, we detail our model for depth estimation using defocus cues. We show that decomposing the problem into defocus estimation and later depth estimation is critical to close the domain gap between synthetic and real data. We show state-of-the-art results on real data while training our models only with synthetic data.

3.1. Method overview

We show a diagram of our method in Fig. 1, which shows the three main elements of our model:

DefocusNet. Instead of directly estimating depth from a stack of RGB images, we first estimate a defocus map using DefocusNet (Section 3.2). In particular, we estimate the amount of defocus per pixel.

DepthNet. This defocus map is used as input to DepthNet, which estimates the scene depth map (Section 3.3). To have sharper and better structured output depth, skip connections are used between the encoder of DefocusNet and the decoder of DepthNet. Both DefocusNet and DepthNet are trained jointly and in an end-to-end manner.

AiFNet. While obtaining a depth map is our end goal, image post-processing applications, *e.g.*, refocusing, further require an all-in-focus (AiF) image, aside from the predicted depth map. We can easily predict the all-in-focus image with the focal stack and the defocus map, therefore, it is trivial to extend our architecture with a head, AiFNet, to predict all-in-focus images (Section 3.4).

In the following sections, we introduce the necessary concepts related to depth and defocus, and proceed to describe the three modules in detail.

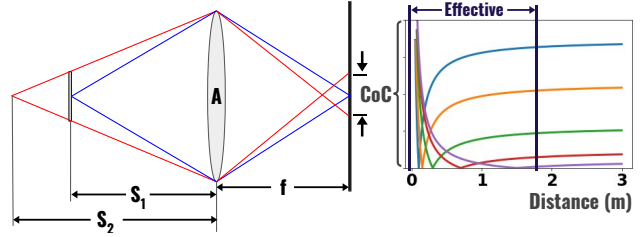


Figure 3. Lens diagram on the left. Circle of confusion plot on the right. Each line in the plot corresponds to a different focus distance.

3.2. Defocus (blur) estimation

Circle of Confusion. In order to compute a defocus map, we first need to establish a measure for sharpness. When a light source passes through the camera lens, the light rays converge to form a focal point, which is found on the image plane of the camera. The circle of confusion (CoC) measures the diameter of the focal point. For a given point in front of the camera, if all rays of light flowing out of it converge into single location in the image plane, then the point will have sharpest projection possible (Fig. 3). Hence, the CoC is a direct translation of the amount of sharpness, equivalently, the amount of defocus.

The CoC can be computed using the following equation:

$$c = \frac{|S_2 - S_1|}{S_2} \frac{f^2}{N(S_1 - f)}, \quad (1)$$

where f is the focal length of the lens, S_1 is the focus distance, S_2 is the distance from the lens to the object, and N is the f-number. The f-number is the ratio of focal length to effective aperture diameter, essentially indicating aperture size. An illustration of a lens system is shown in Fig. 3. The range of acceptable values of CoC, that we consider sharp, depends on the image format and camera model, and it is typically decided based on visual acuity. This range is referred as depth of field (DoF).

On the right of Fig. 3, we show a graph of the evolution of CoC values as the object distance S_2 increases. Each line represents a different value of focus distance S_1 . The more variation lines produces in observation, the easier it is to estimate depth. Once the value surpasses the minimum diameter, the ambiguity of the CoC increases with the distance. After after a certain depth, the CoC no longer changes, indicating we can no longer rely on defocus cues to compute the depth of objects. Thus, the defocus information is useful in a short range which depends on the camera properties.

In this paper, we use the term defocus (blur) and CoC interchangeably. In fact, to construct a defocus map, we compute the CoC values for all pixels in an image, clip all values inside a chosen upper limit, and normalize all values in a range between 0 to 1 (from sharp to blurry).

Depth-from-Defocus limitations. The problem is inherently limited by design, as depth-from-focus works better on short ranges. Nonetheless, ubiquitous depth-from-stereo methods (and by extension, conventional depth cameras with a separate IR projector and an IR camera) are less effective in a short range due to part of the scene being not visible by both cameras. Depth-from-focus can be seen as a solution for short ranges. In our camera settings, the effective range in which we can use defocus to predict depth is within 2 meters, as shown in Fig. 3.

Data-driven defocus estimation. A key design choice of our work is to use defocus estimation as an intermediate step, or supervisory signal, to estimate depth from a focal stack. This allows us to obtain a model that generalizes from synthetic to real images. Therefore, we begin by training a model, DefocusNet, to estimate a defocus map from a set of images. Fig. 4 shows an example of RGB images from our synthetic dataset and their corresponding defocus map, where the pixel values represent their sharpness level.

The focal stack is processed by an autoencoder convolutional neural network (CNN), as shown in Fig. 2. The encoder contains one branch per image in the focal stack, where all branches share the weights of the CNN. Our goal is to encourage the network to perform comparisons between the features extracted at each branch, as to better establish focused and defocused regions. We do this comparison at every layer of the CNN, inspired by [1].

Layer-wise global pooling. The network computes the output of the convolution layer for every input image, then all output feature maps are pooled by a symmetric operation, *i.e.*, we compute the maximum value of each feature map cell across all branches. The globally pooled features, or *global* feature map, are then concatenated to the *local* feature map coming from each branch. The combined output is then passed to the next convolution layer, and the process is repeated. This way, each CNN branch will contain both local as well as global features. Intuitively, this allows the network to compare local features with globally pooled features, finding out the sharpest regions of the image by comparison, and passing those to the next layer of the CNN. The main advantage of using this layer-wise pooling across inputs is that our model can handle an arbitrary number of images as input, making our model extremely flexible.

The CNN is rather shallow with only 4 layers. Since sharpness is a local property, we do not need a large receptive field. The decoder then propagates the estimated focus information from the edges to the center of the objects, where there might not be enough texture to properly estimate sharpness. We also make use of skip connections (by concatenation) to properly recover boundaries in the regressed defocus maps. Global pooling is used only in the encoder, while the decoder has separate branches for each output. The main idea is for the encoder to learn to detect

sharp regions by comparison, and for the decoder to regress a defocus map independently for each input. We use an L2 loss to train DefocusNet without additional regularization.

As we can see in Fig. 1, once defocus is estimated, we can use it as an input to estimate depth.

3.3. Depth estimation.

A depth map can be constructed from defocus maps by using the camera capture settings for each image in the focal stack. However, our network architecture is by design unaware of image order. Therefore, we also include a focus distance map together with the previously predicted defocus map as input to DepthNet, our neural network that is trained for depth regression. The focus distance map is a single channel and single value image, where every pixel takes the value of the focus distance.

We obtain focus distances from our dataset rendering script (Section 4) and then rescale them in the range between 0 and 1. For real images, if we know the order in a focal stack, we can assign focus distances consecutively with computed increments (based on a number of images) in the range 0 to 1. We can also extract the focus distance from EXIF properties (camera settings used to take an image) and rescale the values to the required range. While our architecture needs this additional input, it comes at a minimal cost, does not affect generalization and allows us to have an arbitrary number of input images.

The network architecture for DepthNet is similar to DefocusNet, except we have a single branch also in the decoder to get one depth map as output. Additionally, in the DepthNet decoder we use skip connections from the DefocusNet encoder to combine information from the RGB input image. This helps especially to improve the depth prediction around object boundaries. During training, we use the L2 loss between estimated and ground-truth depth.

The full loss function is shown below:

$$\mathcal{L} = \lambda_a \sum \|I_{def} - E_{def}\|_2 + \lambda_b \sum \|I_{dep} - E_{dep}\|_2$$

where I are ground truth images, E are estimated images, for depth and defocus, $\lambda_{a,b}$ are weight coefficients.

Dynamic stacks. As we mentioned in previous paragraphs, the goal of global pooling across inputs is to encourage the network to compare different input branches. However, such approach has a problem with focal stacks taken with a moving camera or a static camera but a moving scene. In those scenarios, each part of the image will contain drastically different information, and comparison across inputs will not be informative to compute defocus. To handle such scenarios, we propose to use a recurrent autoencoder [6] for DepthNet and DefocusNet, instead of the global pooling autoencoder. Such a recurrent autoencoder concatenates the local features from one branch to the next sequentially, tak-

ing order into account. This allows us to gradually incorporate changes, also changes in the scene, and implicitly compare the amount of defocus in them. Still, such recurrent architecture has its own drawbacks: (i) it has a short memory [3], and (ii) the number and order of images on the focal stack is fixed due to the architecture. For these reasons, we use it only when dealing with dynamic stacks. More architecture details are given in the supplementary material.

3.4. All-in-Focus estimation

In our work, we use a stack of differently focused images to estimate a depth map. However, for image post-processing applications such as refocusing, we additionally need an all-in-focus (AiF) image where all pixels are appropriately sharp. We can estimate such image given a focal stack by combining different image parts according to their sharpness. For this reason, we propose to incorporate such estimation inside our network, reusing the focal stack processing as well as the defocus map. The AiF image is computed by an additional CNN head, the AiFNet. Since AiF prediction is not the main focus of our work, the model description and results are included in the supplementary.

4. Synthetic Training Data

We use only synthetic data to train our full model for depth prediction from focal stacks. As we will show in the experimental section, our network will generalize to depth prediction on real images without the need to fine-tune our model on real data. We create our synthetic dataset using Blender [4] Cycles renderer with reflection turned on, but without shadows. Examples from the dataset are shown in Fig. 4. For training, we render a total of 1000 scenes, each of them with 5 RGB images per focal stack, 5 defocus maps, 1 depth image and 1 all-in-focus image (taken with a wide aperture). Each defocus map was calculated using Equation (1) based on depth and camera parameters. Each image was rendered at a resolution of 256×256 .

Dataset characteristics. We acquired CAD 3D objects from a model repository containing a total of 400 objects [40], and place between 20 to 30 objects per scene, all randomly chosen. Objects are assigned a random size, location and rotation in each scene to have a random spatial arrangement. Locations are limited to the effective range of defocus (Sec. 3.2) and camera field of view. Some objects might not be fully in the camera field of view or can be occluded by other objects. We choose a non-realistic scene composition on purpose due to its simplicity to model, but primarily to avoid overfitting on spatial cues, and instead force the model to focus on defocus cues.

Each object is assigned one random material. Before rendering, we randomize the hue of the diffuse and specular components, glossiness and roughness are chosen within a range that produces a realistic appearance. All materials

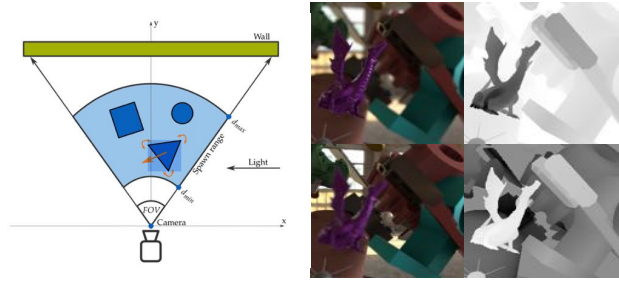


Figure 4. On the left is a schematic for the random scene generation. On the right side are the examples from the synthetic dataset (pairs of RGB images and defocus maps from focal stacks). Sharper regions are darker in defocus maps.

use physically based rendering (PBR) shaders. For illumination, we used 20 different HDR environment maps (EM), both indoor and outdoor. We used fixed camera parameters with fixed focus distances for all scenes. The f-number is set to 1 in order to have a shallow depth-of-field, therefore making depth changes more observable in terms of blur.

Dynamic stacks. To handle camera or scene motion, we additionally create dynamic stacks with 4 images, where the position of the object changes for each image. We assign a random direction and magnitude of translation and rotation for each object at the beginning of sequence rendering. More details are provided in the supplementary material.

5. Evaluation

We present a comprehensive ablation study on a diverse set of synthetic scenes. We further show qualitative and quantitative results for depth prediction on real datasets and also on real images with synthetic blur. We show generalization by training only on synthetic data and obtaining state-of-the-art results on real images with synthetic blur. We provide in the supplementary material additional results on real/synthetic images and all-in-focus image prediction.

5.1. Implementation details

The method is implemented in PyTorch[24]. We train the model using the Adam optimization algorithm [16] with a learning rate of 0.0001. We assign $\lambda_d = 0.02$ and $\lambda_{a,b} = 1$. We provide a detailed description of all network architectures in the supplementary material.

Run-Time Performance. On an Nvidia Titan X, a forward pass of our network takes 70ms on a focal stack with 10 images and 150ms with 20 images. For [33] reported time is 20mins and for [32] - 6.7s.

5.2. Metrics and Datasets

Evaluation metrics. For depth comparison, we use root mean squared error (RMSE) in Table 3 and mean squared

error (MSE) in all other tables in order to compare to existing methods.

Synthetic dataset. Synthetic test data was rendered in the same way as the training data in Section 4, but we use a new set of 10 environment maps, 20 new textures, and 300 new objects. We rendered 4 test sets to evaluate generalization:

(i) *Shape test.* The first test set contains only new objects, while environments and textures are the same as used for training. This is a test on shape generalization.

(ii) *Appearance test.* The second test set contains new objects, textures and environment maps to test appearance generalization.

(iii) *Wide DoF test.* The third test is identical to the second set, but with smaller camera apertures (f-number is from 3 to 10) to show that we rely on defocus cue. Having a larger DoF negatively affects blur estimation because there is less blur information.

(iv) *Medium DoF test.* The fourth test is rendered with the same options as the second set, but with slightly smaller camera apertures (we randomly choose an f-number from 1 to 3 whereas a training set was trained on f-number of 1) to test the generalization to slightly different camera settings.

Real datasets. There are very few datasets that provide focal stacks. We use these datasets for quantitative and qualitative experiments.

(i) *DDFF 12-Scene benchmark [12].* Obtaining focal stacks with standard cameras is a time-consuming task. To speed up the process, [12] uses plenoptic cameras that can capture 4D light-fields. With a single light-field, we can generate a focal stack and an all-in-focus image. This dataset consists of 1200 focal stacks with 10 images each. The dataset is challenging due to the type of scene recorded: many flat and texture-less surfaces such as walls and desks, and other texture-less objects such as monitors, doors and cabinets. Furthermore, their capture settings are not optimal for defocus blur, they shoot with wide DoF and capture scenes with far distances. We use the same training/test split as [12].

(ii) *Mobile Depth [33].* This dataset consists of 13 scenes, each scene has a different number of images in the range between 13 and 32. All images were taken with a mobile phone and aligned using optical flow. Since there are not enough images to train a deep learning approach, we show the results of our model trained only on our synthetic training set.

Synthetically blurred real datasets. Due to lack of datasets with focal stacks for quantitative experiments, we propose to use popular indoor datasets and create focal stacks from RGB images and ground-truth depth. We apply synthetic blur following [11].

(i) *NYU Depth Dataset v2 [23].* This dataset consists of 1449 pairs of aligned RGB and depth frames. We use the regular split between test and train sets.

(ii) *7-Scenes [27].* This dataset consists of around 43000

images of aligned RGB and depth frames scenes. Due to large size of the dataset, we randomly sample total of 890 images from all 7 sequences for our tests.

(iii) *Middlebury stereo dataset [26].* This dataset consists of 46 images of stereo RGB pairs and disparity frames. Based on provided camera calibration parameters, we compute the depth map for the RGB image corresponding to the right camera.

(iv) *SUN RGB-D [39].* This dataset is a combination of NYU depth v2[23], Berkeley B3DO[13] and SUN3D[35] datasets with improved depth maps. It consists of 10,000 images of aligned RGB and depth frames. Similarly, we randomly sample total of 490 images for our tests.

5.3. 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. Table 1 shows estimation results between different models on the depth estimation task. We explain and compare all models below. In the tables, "All" indicates that all N images of the focal stack were used for prediction, while "Random" indicates the use of a randomly chosen number of images $r \leq N$.

Architecture choice. We consider two architecture types: (i) FixedAE, our baseline with single autoencoder without global pooling layers that is trained on a single input with fixed-sized focal stack with N images, and (ii) PoolAE, the autoencoder with global pooling layers, which can take any number of images as input. In Table 1, the first two rows show a direct comparison of the two models. FixedAE was trained on a specific number of inputs and shows good performance only when tested on the same number of images (column All). In contrast, PoolAE shows generalization to the column Random, where the number of input images varies but accuracy is maintained. From these tests, we can summarize that PoolAE architecture is robust, generalizes better than just stacking all inputs in single AE, and has the added value of processing any number of inputs without the need to retrain the model.

Is defocus needed? We compare models that directly estimate depth from the input RGB image and our proposed approach, where we first estimate a defocus map and then depth. In Table 1, with our PoolAE architecture, we achieve a much better result when going over the defocus map (row 3.) compared to a direct estimation (row 2.). On the *Wide DoF test*, PoolAE with defocus map (row 3.) performs worse due to having less defocus blur to rely on. It confirms that our model uses defocus as the primary signal to estimate depth. If that signal is inexistent due to a too wide DoF, performance drops as expected. Overall, we can clearly conclude that using defocus information is beneficial for depth estimation, and is cue that allows us to generalize to a diverse set of object shapes and appearances.

Models	Shape		Appearance		Wide DoF		Medium DoF	
	All	Random	All	Random	All	Random	All	Random
1. FS → Depth (FixedAE)	0.014	0.097	0.012	0.095	0.103	0.111	0.083	0.112
2. FS → Depth (PoolAE)	0.031	0.036	0.034	0.039	0.049	0.050	0.047	0.048
3. FS → Defocus → Depth (PoolAE)	0.008	0.013	0.007	0.012	0.042	0.078	0.014	0.030

Table 1. Results on the synthetic data test sets for depth estimation models with focal stacks (FS) as input. *Row 1.* Direct depth prediction with a fixed-sized AE. *Row 2.* Direct depth prediction with the proposed AE with global pooling. *Row 3.* Depth prediction for our method, using the predicted defocus map as supervisory signal.

Models	Shape	App.	W. DoF	M. DoF
1. RGB → Depth	0.032	0.031	0.134	0.108
2. AiF → Depth	0.049	0.050	0.050	0.054

Table 2. Results of depth estimation on the synthetic test set using only one image as input, either one of the out-of-focus images of the focal stack (Row 1), or the all-in-focus image (Row 2).

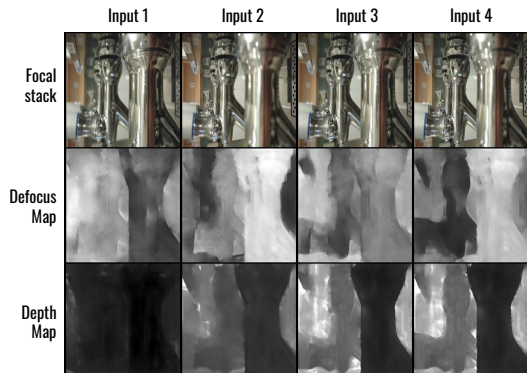


Figure 5. Our sequential estimates on a real focal stack. The first row images are inputs to our pipeline. The other horizontal sequences show our outputs for a growing number of input frames. So the results on the left use only the first input image and on the right use all four inputs. Note how adding more inputs quickly improves the depth.

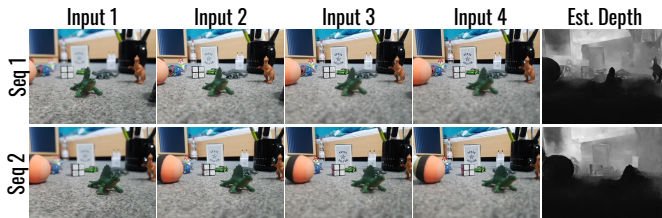


Figure 6. Our estimates on a real focal stack on a dynamic sequence.

Single image vs. Focal stack. We also train our method on a single image input setting. Such network is expected to perform worse when predicting depth, as it is not able to rely on blur comparison between images in the focal stack. Results are presented on the first row of Table 2. We clearly see the error in depth prediction is much larger than all models that use a focal stack (Table 1).

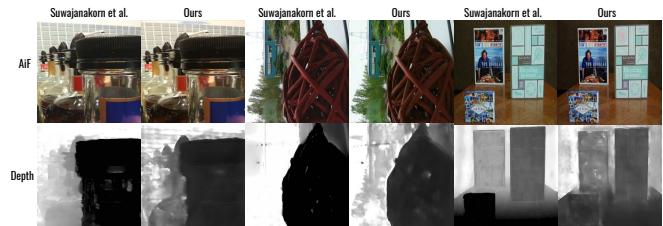


Figure 7. Qualitative results on Mobile Depth dataset. We compare our method with a model-based approach. [33].

All-in-Focus vs. Out-of-Focus. Out-of-focus images give more information related to depth, as shown in [5]. We also perform a similar test in Table 2, where we compare to a model trained on all-in-focus images. We can see from the *Shape* and *Appearance* tests, that defocus gives more information for depth even without explicitly computing the defocus map. The model in Row 1. also performs worse in both camera aperture tests, since they both have wider DoF. Wider DoF has less defocus blur and does not give any information to the model. The Row 2 model does not rely on defocus and therefore shows similar results on all tests.

5.4. Evaluation on Real data

Synthetically blurred NYU, 7 scenes, Middlebury and SUN RGB-D. In this section, we show quantitative results on cross-dataset and domain generalization task. Since defocus blur is only effective on close distances, we conduct experiments on 4 different versions of each dataset: (i) *regular*, no modifications, (ii) *less than 2m*, only depths within 2 meters are taken into account, since this is the functioning range for our method, (iii) *normalized version*, depth is rescaled from 0 to 10 meters to a range from 0 to 1 meters, and (iv) *45 degrees*, the normalized version with images rotated 45 degrees. The last test of rotating input images by 45 degrees is a simple yet effective way to show that current datasets have photographic bias, which leads to overfitting on the training settings.

We compare our method with the state-of-the art single image depth estimation model VNL [36] in Table 3. We trained our method on the regular synthetic dataset, "Ours", and its normalized version, "Ours*". We additionally show a version fine-tuned on the NYU dataset. Due to the difference in datasets, we use the median to rescale estimated

Models	Training data		NYU				7 scenes				Middlebury				SUN RGB-D			
	Synth.	NYU	Norm.*	45deg.*	Regular	<2m	Norm.*	45deg.*	Regular	<2m	Norm.*	45deg.*	Regular	<2m	Norm.*	45deg.*	Regular	<2m
Ours	✓		-	-	1.054	0.272	-	-	0.504	0.282	-	-	0.803	0.384	-	-	0.721	0.259
Ours*	✓		0.056	0.073	-	-	0.030	0.037	-	-	0.052	0.063	-	-	0.037	0.052	-	-
Ours	✓	✓	-	-	0.493	0.181	-	-	0.277	0.189	-	-	0.544	0.351	-	-	0.360	0.196
Ours*	✓	✓	0.013	0.111	-	-	0.010	0.045	-	-	0.025	0.079	-	-	0.014	0.073	-	-
VNL [36]		✓	0.040	0.100	0.395	0.206	0.033	0.050	0.328	0.244	0.064	0.071	0.645	0.400	0.037	0.068	0.370	0.289

Table 3. Regular - no modifications, <2m - same as regular but counting results only for depth less than 2 meters, and normalized version - depth was rescaled to range from 0 to 1. All models with * were trained for normalized sets. 45 degrees set is a version with images rotated 45 degrees. Our models trained first on synthetic dataset then tested with and without finetuning on NYU dataset. All tests show RMSE values.

Models	MSE
FS → Depth (F)	$11.7 * 10^{-4}$
FS → Depth (P)	$13.2 * 10^{-4}$
FS → Defocus → Depth (P)	$9.1 * 10^{-4}$
DDFF [12]	$9.7 * 10^{-4}$
VDFD [22]	$73.0 * 10^{-4}$

Table 4. Results of depth estimation on DDFF-12.

Models	MSE
FS → Depth	0.184
FS → Defocus → Depth	0.045

Table 5. Results of depth estimation on Mobile Depth dataset.

depth for all models to match ground-truth depth as in [8]. VNL was trained on the NYU dataset and performs well on that dataset. However, on other datasets, its performance drops in comparison to our methods that use defocus cues. Our fine-tuned versions, Ours(Synth.+NYU) and Ours*(Synth.+NYU), perform best across all test but fail the 45 degree test. The purely synthetically-trained model shows similar or better performance to the method trained purely on real data, and generalizes much better in the case of the 45 degree experiment. This clearly shows the bias of the real dataset that does not allow the networks to generalize to any scene configuration.

We use focal stacks with 4 images which is to some degree unfair to single image methods. Nonetheless, capturing a focal stack is straightforward: (i) it takes just slightly longer than a single shot, (ii) we do not need additional camera hardware, and (iii) we do not need to move the camera to satisfy stereo requirements. The benefits in depth prediction accuracy come at a very little cost during caption.

DDFF 12-Scene. We compare our approach to a CNN-based method [12] and a classic method (VDFD) [22]. As explained in Section 4, this dataset is not ideal for DefocusNet due its wide DoF. Our synthetically trained models did not directly perform well, but after fine-tuning on the provided training data, we were able to achieve state-of-the-art results, shown in Table 4. As we can see, PoolAE network with DefocusNet shows better performance. We conclude that our approach generalizes well within similar types of real images and is able to handle texture-less surfaces.

Mobile Depth from Focus. We compare our method with traditional methods [33] that take focal stack images as inputs. The dataset does not have ground truth depth, but the authors provide their depth estimations. We compare our models to their depth to test the generalization capabilities of our approach. Table 5 shows that a direct approach does not generalize from synthetic to real images while our method does. We also show qualitative results in Fig. 7 and Fig. 5. Note that our models are *trained on synthetic data only*, and are not fine-tuned on this dataset. Additionally, we show visual results with the increasing number of input images in Fig. 5. We can see that it gradually improves the depth estimates thanks to our pooling architecture.

There are several aspects that work in favor of a better generalization in our work: (i) we use down-scaled images, e.g., original images from Mobile Depth are 360 x 640, which makes the out-of-focus blur details similar for most conventional cameras; (ii) the method is based on the comparison between differently focused inputs rather than analysis of blur shape/size; (iii) [1] showed that with enough randomness in synthetic noise, invariance to various real noise can be achieved.

Dynamic stacks. Since we lack real test data to show our model on dynamic stacks, we implemented a smartphone application to capture focal stacks. Fig. 6 shows qualitative results of the recurrent approach on real data recorded with a moving camera. Note, the models were trained with synthetically moving sequences. We show more qualitative results for all datasets in our supplementary material.

6. Conclusion

We presented a data-driven approach for estimating depth using defocus cues from a focal stack as a supervisory signal. Our key design decision is to use domain invariant defocus information as supervision for the depth prediction. This allows our model to generalize from synthetic to real images. Our permutation-invariant network allows us to correctly estimate depth with any focal stack size, and we further show a simple extension to process stacks with either moving camera or moving scene.

Acknowledgements. This research was funded by the Sofja Kovalevskaja Award of the Humboldt Foundation.

References

- [1] Miika Aittala and Frédo Durand. Burst image deblurring using permutation invariant convolutional neural networks. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VIII*, pages 748–764, 2018.
- [2] Saeed Anwar, Zeeshan Hayder, and Fatih Murat Porikli. Depth estimation and blur removal from a single out-of-focus image. In *BMVC*, 2017.
- [3] Yoshua Bengio, Patrice Y. Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Trans. Neural Networks*, 5(2):157–166, 1994.
- [4] Blender Foundation. *Blender - a 3D modelling and rendering package*, 2018.
- [5] Marcela Carvalho, Bertrand Le Saux, Pauline Trouvéloux, Andrés Almansa, and Frédéric Champagnat. On regression losses for deep depth estimation. *ICIP*, 2018.
- [6] Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, Christoph Schied, Marco Salvi, Aaron E. Lefohn, Derek Nowrouzezahrai, and Timo Aila. Interactive reconstruction of monte carlo image sequences using a recurrent denoising autoencoder. *ACM Trans. Graph.*, 36(4):98:1–98:12, 2017.
- [7] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2002–2011, 2018.
- [8] Ariel Gordon, Hanhan Li, Rico Jonschkowski, and Anelia Angelova. Depth from videos in the wild: Unsupervised monocular depth learning from unknown cameras. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019.
- [9] Xiaoyang Guo, Hongsheng Li, Shuai Yi, Jimmy S. J. Ren, and Xiaogang Wang. Learning monocular depth by distilling cross-domain stereo networks. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XI*, pages 506–523, 2018.
- [10] Xiaoyang Guo, Hongsheng Li, Shuai Yi, Jimmy S. J. Ren, and Xiaogang Wang. Learning monocular depth by distilling cross-domain stereo networks. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XI*, pages 506–523, 2018.
- [11] Shir Gur and Lior Wolf. Single image depth estimation trained via depth from defocus cues. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 7683–7692, 2019.
- [12] 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.
- [13] Allison Janoch, Sergey Karayev, Yangqing Jia, Jonathan T. Barron, Mario Fritz, Kate Saenko, and Trevor Darrell. A category-level 3-d object dataset: Putting the kinect to work. In *IEEE International Conference on Computer Vision Workshops, ICCV 2011 Workshops, Barcelona, Spain, November 6-13, 2011*, pages 1168–1174, 2011.
- [14] Hae-Gon Jeon, Jaesik Park, Gyeongmin Choe, Jinsun Park, Yunsu Bok, Yu-Wing Tai, and In So Kweon. Accurate depth map estimation from a lenslet light field camera. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1547–1555, 2015.
- [15] Hyeongwoo Kim, Christian Richardt, and Christian Theobalt. Video depth-from-defocus. In *International Conference on 3D Vision (3DV)*, pages 370–379, October 2016.
- [16] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.
- [17] Zhengqi Li and Noah Snavely. Cgintrinsics: Better intrinsic image decomposition through physically-based rendering. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III*, pages 381–399, 2018.
- [18] Zhengqi Li, Kalyan Sunkavalli, and Manmohan Chandraker. Materials for masses: SVBRDF acquisition with a single mobile phone image. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III*, pages 74–90, 2018.
- [19] Reza Mahjourian, Martin Wicke, and Anelia Angelova. Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 5667–5675, 2018.
- [20] M. Mancini, G. Costante, P. Valigi, T. A. Ciarfuglia, J. Delmerico, and D. Scaramuzza. Toward domain independence for learning-based monocular depth estimation. *IEEE Robotics and Automation Letters*, 2(3):1778–1785, July 2017.
- [21] Michael Moeller, Martin Benning, Carola Schönlieb, and Daniel Cremers. Variational depth from focus reconstruction. *IEEE Transactions on Image Processing*, 24(12):5369–5378, 2015.
- [22] Michael Moeller, Martin Benning, Carola Schonlieb, and Daniel Cremers. Variational depth from focus reconstruction. *IEEE Transactions on Image Processing*, 24(12):5369–5378, Dec. 2015.
- [23] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgb-d images. In *ECCV*, 2012.
- [24] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.
- [25] Xingchao Peng, Ben Usman, Kuniaki Saito, Neela Kaushik, Judy Hoffman, and Kate Saenko. Syn2real: A new benchmark for synthetic-to-real visual domain adaptation. *CoRR*, abs/1806.09755, 2018.
- [26] Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nescic, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate

- ground truth. In *Pattern Recognition - 36th German Conference, GCPR 2014, Münster, Germany, September 2-5, 2014, Proceedings*, pages 31–42, 2014.
- [27] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew W. Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In *2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013*, pages 2930–2937, 2013.
- [28] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2242–2251, 2017.
- [29] Nikolai Smolyanskiy, Alexey Kamenev, and Stan Birchfield. On the importance of stereo for accurate depth estimation: An efficient semi-supervised deep neural network approach. In *2018 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 1007–1015, 2018.
- [30] Pratul P. Srinivasan, Rahul Garg, Neal Wadhwa, Ren Ng, and Jonathan T. Barron. Aperture supervision for monocular depth estimation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 6393–6401, 2018.
- [31] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3d orientation learning for 6d object detection from RGB images. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VI*, pages 712–729, 2018.
- [32] Jaeheung Surh, Hae-Gon Jeon, Yunwon Park, Sunghoon Im, Hyowon Ha, and In So Kweon. Noise robust depth from focus using a ring difference filter. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jul 2017.
- [33] 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.
- [34] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. Demon: Depth and motion network for learning monocular stereo. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5622–5631, 2017.
- [35] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. SUN3D: A database of big spaces reconstructed using sfm and object labels. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, pages 1625–1632, 2013.
- [36] 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.
- [37] Feihu Zhang, Victor Adrian Prisacariu, Ruigang Yang, and Philip H. S. Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 185–194, 2019.
- [38] Shanshan Zhao, Huan Fu, Mingming Gong, and Dacheng Tao. Geometry-aware symmetric domain adaptation for monocular depth estimation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 9788–9798, 2019.
- [39] Bolei Zhou, Àgata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 487–495, 2014.
- [40] Qingnan Zhou and Alec Jacobson. Thingi10k: A dataset of 10,000 3d-printing models. *arXiv preprint arXiv:1605.04797*, 2016.