Synthesis Depth-of-Field from Depth Estimation using Mobile Devices

Chenhao Yang
Carnegie Mellon University
Pittsburgh, PA
yangchenhao@cmu.edu

Our method takes one in-the-wild all-in-focus image as input and outputs image with synthetic blur, thus creating a shallow depth-of-field effect similar to images taken with DSLR cameras and large aperture. Our method involves applying a well-trained DNN for estimating depth map and fast computing blur for creating *bokeh*. In our GUI, users can freely interact with the system by selecting the focal plane and aperture size.

Abstract

In this project, we investigated some recent advances in depth estimating algorithms using mobile devices for producing realistic shallow depth-of-field as captured with DSLR-style cameras. We re-implemented some algorithms and developed a tool for creating digital synthetic blur. The key idea of our approach is to use well-trained DNN for estimating depth map from monocular camera, and then apply blur on these photos based on depth. We tested the framework using pictures taken with mobile devices which contains a large variety of scenes. Our framework showed impressive results for most of the scenarios, it is made available at https://github.com/afiretony/synthesis-depth-of-field.

1. Introduction

Depth of field refers to the distance between the nearest and farthest objects in a photograph that appear acceptably sharp. In other words, it is the portion of the scene that is in focus. A shallow depth of field means that only a small part of the scene is in focus, while a large depth of field means that more of the scene is in focus. Depth-of-field can greatly affect the appearance and impact of a photograph. While it plays an important role in professional photography, we can hardly see it in photos taken by a mobile device, e.g., a smartphone. This is because a smartphone usually has a tiny camera sensor, often around 25 square mm. Such limited space makes an large aperture unable to be mounted, thus cannot create same level of depth-of-field as DSLR cameras optically.

Recently, smartphone manufacturers have taken computational ways to overcome this issue. For example, some used two cameras that serve as a stereo pair to compute depth map [12] [19]. However, adding a second or even third camera increases manufacturing costs and damages the simplicity of the phone. Similarly, some manufacturers added additional time-of-flight sensors [14] which are expensive and power intensive. More recently, some manufacturers developed dual-pixel cameras that can calculate disparity map [29] [31] [2] [33].

However, all fore-mentioned methods are device specific and unable to process in-the-wild pictures. For example, a user may would like to add blur to an all-in-focus image downloaded from internet. Such limitation may bring negative user experience. To address this issue, we require the system to have following features:

- (1) Realistic synthetic blur and controllable depth-of-field.
- (2) Fast processing.
- (3) Works on a variety of scenes.
- (4) Works for in-the-wild images.

Our system utilized data-driven approach, firstly, we used a deep neural network with encoder-decoder structure that trained on a variety of scenes for estimating depth map. Then, we applied edge-aware filter on the depth map to get smoother output. Lastly, we applied blur on each pixel of the image according to their depth.

2. Related Work

2.1. Depth Estimation

The key to success of applying synthetic blur comes from good depth estimation as they are needed for later mathematical formulation. Depth estimation is a wide area of Computer Vision, in this part, we review some well-known methods commonly used on mobile devices and cheap camera systems.

2.1.1 Depth from Defocus

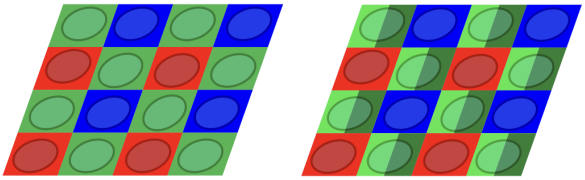
Depth from defous (DFD) refers as estimating depth map from two differently-focused images of a scene [9] [28] [20] [22]. The problem is modeled as given two images with varying circle of confusion differed by blur level σ, estimate blur size at each pixel and relate it with depth. However, such method requires two varying image pairs focusing at different location as input and generally dense surface texture and images with significant defocus blur.

2.1.2 Depth from Focus

The concept of *depth from focus (DfF)* involves recovering depth from a stack of images with varying depth-of-field of the same scene. Attempts have made to apply this technique on mobile devices [27], and additional focal stack alignment are done to compensate for parallax and viewpoint changes produced by a moving, handheld capture. However, such method is again limit by specific requirement at data acquisition and not applicable for in-the-wild scenarios. Moreover, it is limited by having rich-feature and complex scene to differentiate in-the-focus frame from others.

2.1.3 Depth from Duel-Pixel

Dual-Pixel (DP) cameras works by splitting each pixel in half and creating image pairs in a single snapshot. Although DP was developed for auto-focus purpose [3], some works leverage from it for estimating depth by treating DP pairs as stereo pairs which effectively gives a 2-sample light field with a narrow around 1 millimeter baseline.



(a) Traditional Bayer Sensor (b) Dual-Pixel Sensor

Figure 1. A modern Bayer sensor consists of interleaved red, green, and blue pixels underneath a microlens array. (a). In dualpixel sensors, the green pixel under each microlens is split in half (b), resulting in two green images that act as a narrow-baseline stereo camera, much like a reduced light field camera. [10]

Most smartphone manufacturers have adopted DP sensors in rear camera nowadays and they have developed their own algorithms marketed as *Portrait Mode* [29], which make it reasonable and convenient to retrieve depth map from image pairs from DP camera. However, it is still not applicable for in-the-wild pictures and limited by close-range imaging as depth estimated for far away object are often mistaken [10] [21].

2.2. Learning-based Depth Estimation

2.2.1 Dataset

There are a number of datasets for monocular depth estimation, with different types and depth ranges between indoor and outdoor scenes. Samples of these dataset are visualized in figure 2.

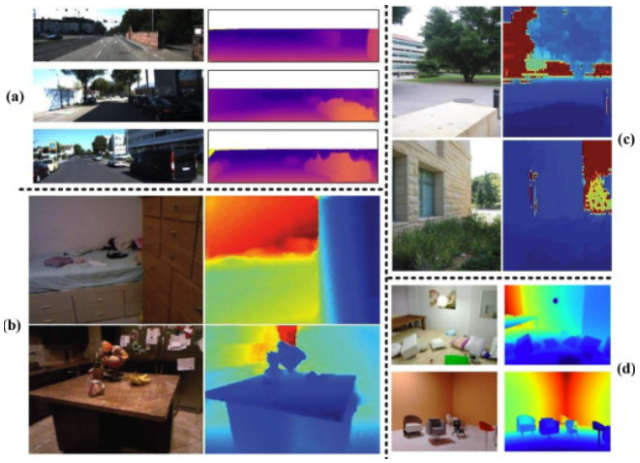


Figure 2. Samples of monocular depth estimation datasets. (a) is KITTI dataset [35], (b) is NYU Depth V2 dataset [130], (c) is Make3D dataset [123], [124], and (d) is SceneNet RGB-D dataset [95] (the left images are RGB images and the right are the ground-truth depth maps).

KITTI dataset [11] is an outdoor dataset for monocular deep estimation and object detection and tracking based on deep learning. KITTI dataset is captured through a car equipped with 2 high-resolution color cameras, 2 gray-scale cameras, laser scanner and global positioning system (GPS), whose maximum measuring distance is 120 m. The dataset contains a total of 93,000 RGB-D training samples, including five categories.

NYU Depth V2 dataset [26] is an indoor dataset for monocular depth estimation based on deep learning, which is provided by Silbereman et al.. NYU Depth V2 dataset contains 407,024 frames of RGB-D image pairs captured by a RGB camera and the Microsoft Kinect depth camera to simultaneously collect the RGB and depth information of 464 different indoor scenes. The authors select 1,449 images from the dataset and use the coloring algorithm to fill and obtain dense depth maps, which are manually labeled with the semantic information. Some samples of NYU Depth V2 dataset are shown in figure 2.

Make3D dataset [24], [25] is another outdoor dataset for monocular depth estimation based on deep learning, which is constructed by Saxena et al.. Make3D dataset includes daytime city and natural scenery, with depth maps being collected by a laser scanner.

The above datasets, KITTI, NYU Depth V2, and Make3D, are all collected from real scenes. There are some virtual datasets generated by computers, such as SceneNet RGB-D dataset [17], and SYNTHIA dataset [23]. These virtual datasets include various scene types under different weather, environment, and lighting conditions.

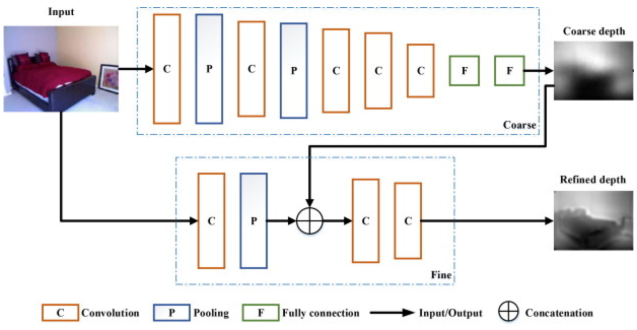


Figure 3. The architecture of multi-scale network for monocular depth estimation proposed by Eigen et al.[7]. The top module is the coarse network for coarse estimation and the bottom module is the fine network for refined depth map.

2.2.2 Overview

Deep neural networks have played an important role in various areas with their powerful feature learning ability. Monocular depth estimation based deep learning is a task of learning depth maps from a single 2D color image through a deep neural network, which was firstly proposed by Eigen et al. [7] in 2014. It was a coarse-to-fine framework, where the coarse network learned the global depth on the entire image to obtain a rough depth map and the fine network learned the local features to refine the depth map, as shown in figure 3. Since then, many researchers have carried out deep learning methods for monocular depth estimation [6], [8], [15], [32].

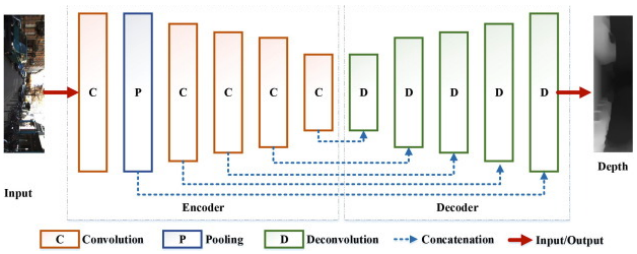


Figure 4. The general pipeline of deep learning for monocular depth estimation. The left module is encoder network learning depth features layer-by-layer, and the decoder network in the right module recovers the depth map.

2.2.3 General model

The framework of monocular depth estimation based on deep learning is an encoder-decoder network, with the RGB image input and depth map output, as shown in figure 4. The encoder network consists of convolution and pooling layers to capture the depth features, and the decoder network includes deconvolution layers to regress the estimated pixel-level depth map, with the same size as the input. Additionally, in order to preserve the features of each scale, the corresponding layers of encoder and decoder are concatenated with skip-connections. The entire network is constrained and trained by the depth loss functions and converges when the desired depth map is generated.

3. Methods

In this section, we present our method for creating synthetic depth-of-field effects for in-the-wild photos.

3.1. DNN-based monocular depth map estimation

To address fore-mentioned device-specific issue brought by classical depth estimation approaches, we adopted a well-trained deep learning based depth map estimator purposed by Alhashim *et al*. [4].

Architecture. Figure 5 shows an overview of the encoder-decoder network for depth estimation purposed by Alhashim *et al*. For the encoder, the input RGB image is encoded into a feature vector using the DenseNet-169 [13] network pretrained on ImageNet [5]. This vector is then

fed to a successive series of up-sampling layers [16], in order to construct the final depth map at half the input resolution. These upsampling layers and their associated skip-connections form the decoder.

Loss function To train the network, loss L between ground truth depth map y and prediction $\hat{y}$ as the weighted sum of three loss functions:

$$L(y, \hat{y}) = \lambda L_{depth}(y, \hat{y}) + L_{grad}(y, \hat{y}) + L_{SSIM}(y, \hat{y})$$

The first loss term $L_{depth}(y, \hat{y})$ is the point-wise L1 loss defined on the depth values:

$$L_{depth}(y, \hat{y}) = \frac{1}{n} \sum_{p}^{n} |y_p - \hat{y_p}|$$

The second loss term $L_{grad}(y, \hat{y})$ is the L1 loss defined over the image gradient g of the depth image:

$$L_{grad}(y, \hat{y}) = \frac{1}{n} \sum_{p}^{n} (|g_x(y_p, \hat{y_p})| + |g_y(y_p, \hat{y_p})|)$$

where g_x and g_y, respectively, compute the differences in the x and y components for the depth image gradients of y and $\hat{y}$.

The third term $L_{SSIM}(y, \hat{y})$ uses the Structural Similarity (SSIM):

$$SSIM(y, \hat{y}) = \frac{(2\mu_y\mu_{\hat{y}} + c_1)(2\sigma_{y\hat{y}} + c_2)}{(\mu_y^2 + \mu_{\hat{y}}^2 + c_1)(\sigma_y^2 + \sigma_{\hat{y}}^2 + c_2)}$$

where $\mu_y = \sum_i^N y_i$ is the averaging over all pixels in a scene, it corresponds to the luminance of a scene. $\sigma_y = \left(\frac{1}{N-1}\sum_{i=1}^{N}(y_i - \mu_y)\right)^{0.5}$ is the standard deviation of the scene for measuring contrast. The loss is defined as

$$L_{SSIM} = \frac{1 - SSIM(y, \hat{y})}{2}$$

Implementation Details Alhashim *et al.* implemented their proposed depth estimation network using TensorFlow and trained on four NVIDIA TITAN Xp GPUs with 12GB memory. The weights for the decoder are randomly initialized. In all experiments, they used the ADAM optimizer with learning rate 0.0001 and parameter values β_1 = 0.9, β_2 = 0.999. The batch size is set to 8. The total number of trainable parameters for the entire network is approximately 42.6M parameters. Among all datasets mentioned, we found NYU Depth v2 most similar to our task for depth estimation for everyday scenes. The training is performed for 1M iterations for NYU Depth v2, needing 20 hours.

3.2. Edge aware filtering of depth map

The depth map estimated by encoder-decoder architecture may contain noises that will bring discontinuity to the final output. To address this issue, we used bilateral filter [18] for improving results.

3.3. Computing blur parameters

We have a somewhat correct and smooth depth map, now we need to pre-compute defocus blur parameters. If we assume the paraxial and thin-lens approximations, there is a staight-forward relationship between circle of confusion and depth O. It implies that:

$$c = mD\frac{|O - S|}{O}$$

where O is actual object distance, estimated from depth map, S is in-focus object distance and D is aperture diameter. The above equation showed that there is an linear relationship between blur radius and inverse depth. It is computed from similar triangle, as illustrated in figure 6. To make it easier for novices to make good shallow-depth-of-field pictures, we artificially keep depth within δ units of S sharp by mapping them to zero blur radius, i.e., we leave pixels at depth [S - δ, S + δ] unblurred. Combining we have:

$$c = mD\frac{max(0, |O - S| - \delta)}{O}$$

3.4. Applying the blur

As suggested by Wadhwa *et al.* [29], when it comes to performing the blur, there are endless debate in photography community concerning the shape and visual quality of the defocused portions of the scene, known as *bokeh*. When produced optically, bokeh is primarily determined by the shape of the aperture in the camera's lens, e.g., a six-bladed aperture would produce a hexagonal bokeh. In our project, we choose to simulate a circle aperture with disk blur. To compute blur for each pixel would be expensive, here we used Opencv's filter2D method which computes correlation between defined blur kernel and source image. In order to get benefited from this, we computed blur in several passes covering different depth range. Namely, we divided depth into several ranges o_j and compute their corresponding blur kernel to form a stack of synthetic blur at different level. In practice, we normalized depth to 0 and 1 and set 25 depth ranges. Then we apply a mask on each image of the stack to form the final rendered image.

$$I_j = \alpha_j \cdot corr(K(o_j), src)$$

where K is blur kernel that depended on depth and src is the original image. In order to get a even smoother transition, we followed Wadhwa *et al.* [29] to set α_j as a truncated tent function on the depths in that range:

$$\alpha_j = (1 + \frac{1}{\eta} min(o - o_{j-1}, o_j - o))|_{[0,1]}$$

where $(\cdot)|_{[0,1]}$ implies clamping a value within $[0, 1]$. This function enforces α_j for 1 at pixels within depth range

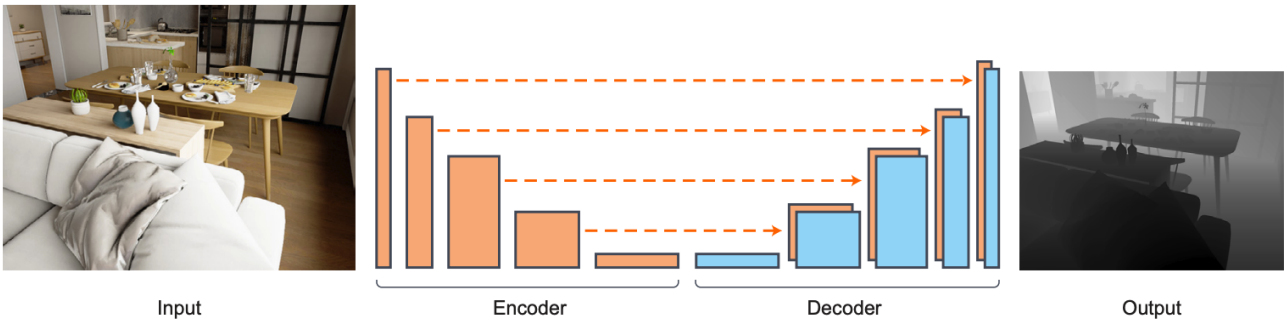


Figure 5. Overview of the network architecture purposed by Alhashim *et al*. [4]. They employed a straightforward encoder-decoder architecture with skip connections. The encoder part is a pre-trained truncated DenseNet-169 [13] with no additional modifications. The decoder is composed of basic blocks of convolutional layers applied on the concatenation of the 2× bilinear upsampling of the previous block with the block in the encoder with the same spatial size after upsampling.

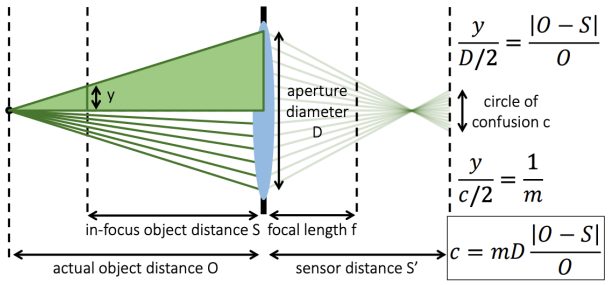


Figure 6. Circle of confusion. [1]

$[o_{j-1}, o_j]$ and tapers fown linearly outside the range, hitting 0 after η depth units from the bounds. We set η so that adjacent bands overlap 25%, i.e., $\eta = 0.25 * (o_j - o_{j-1})$ to get a smoother transition between them. Figure 7 shows the described truncate mask function α_j.

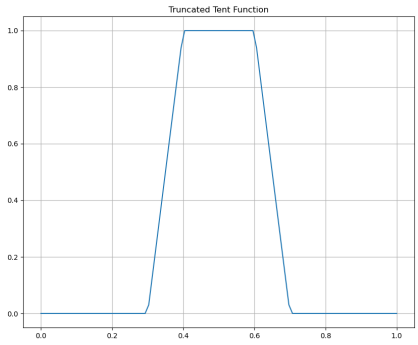


Figure 7. Truncated α_j [29]

4. Experiments

We tested our framework using multiple images containing a wide range of scenes.

4.1. Depth map inference

In our framework, we used a pre-trained model with encoder-decoder structure purposed by Alhashim *et al*. [4], the model is trained on NYU Depth v2 [26] which contain indoor everyday scenes. Although the dataset contains only indoor scene, it generalizes well for our collected outdoor scenes as well. Samples of inference results on our own data are visualized in figure 8.

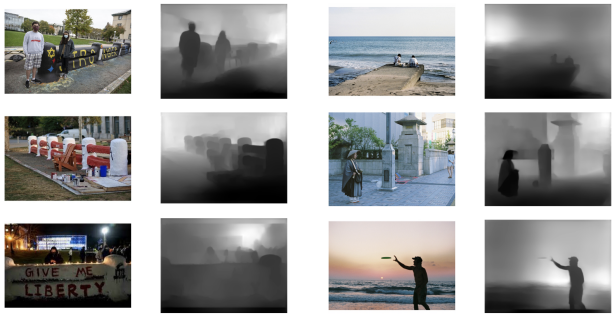


Figure 8. Depth estimated by Alhasim *et al*. [4]

4.2. Applying blur

4.2.1 Different focal plane

Our framework enables user selecting different focal plane by clicking pixel location of the image. The in-focus are indicated in red box. In figure 10, we present sample with defocus blur applied at different focal planes. This process assembles selecting focus point in a DSLR camera.

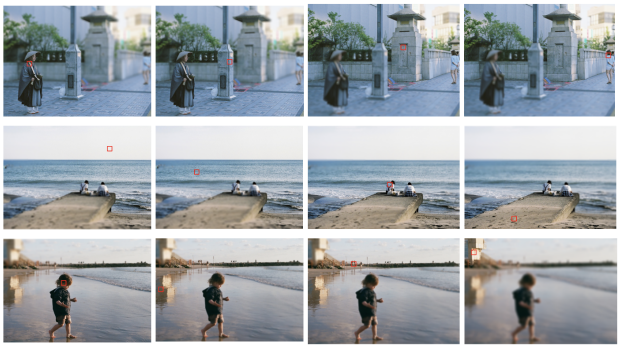


Figure 9. Defocus blur applied with different focal plane. Focal point are indicated in red box, image are sharp in regions near focus point and blurry far away from these points.

4.2.2 Different aperture

Our framework enables user adjusting synthetic aperture level by adjusting bar at the top of our GUI.

Figure 10. Defocus blur applied with synthetic aperture. From left to right, it indicates larger aperture and narrower depth-of-field. The focal plane are the same across the samples.

4.3. Speed

One of the critical requirements for our framework is the speed, as we don't expect user wait too long for getting a rendered image with synthetic blur applied. In this project, we are testing it on a server with Nvidia Tesla T4 GPU enabled only and not extending to edge computing platforms such as a mobile phone, but we can get a primary understanding of the computation needed.

- **Depth map inference:** Inferencing time for one image [640x480] takes 0.5 seconds, loading model takes 24 seconds.
- **Blur calculation and merging:** 125 mili-seconds.

Most smartphone nowadays are equipped with GPU in processing unit and able to perform paralleled deep neural network computations. We believe the time required for rendering a final output is reasonable and possible for deploying on edge computation devices.

5. Discussion

5.1. Improving depth map estimation

In our project, we used a out-of-shell network purposed by Alhashim *et al*. [4] for predicting depth map using monocular input, the model is trained on NYU Dpeth v2 [26]. It is well-known that deep learning based model have good generalization quality on sample that are similar to the training dataset. Our task requires the model to handle everyday scene that might appear and taken by user with their mobile devices. This implies collection of wider and more diverse content from real-world environments. One promising and feasible way is to use RGB-D data collected by duel-pixel enabled mobile phones as suggested by Wang *et al*. [30]. Such data acquisition and training process are beyond scope of this project, and serve as future work.

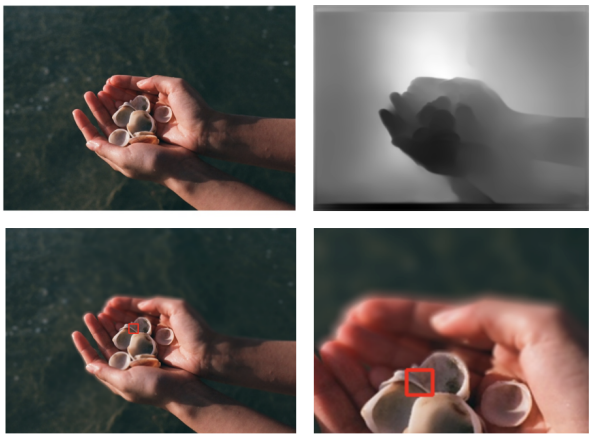


Figure 11. An example of failed depth estimation and unnatural synthetic blur applied. On the upper left is the original image, on the upper right showcase an incorrect depth map estimated, on the bottoms are unnatural blur applied.

Figure 12. An example of before and after applying bi-literal filter on the depth map, the right-most image indicates difference between before and after filtering.

Figure 13. Results of samples, from left to right: original all-in-focus image, depth map estimated, rendered results with synthetic blur applied.

5.1.1 Inference of edge aware filtering

As mentioned in 3.2, we applied edge aware filtering on the depth map prior to estimate blur parameters. The purpose

of applying this filter is to remove random noise in the depth map while remaining sharp transition at edges or other high frequency areas. An example of before and after applying bilateral filter [18] is illustrated in figure 12. The depth map is smoother after applying bilateral filter.

Figure 14. Final rendered output of without and with applying biliteral filter on the depth map.

In figure 14, we present final rendered output of without and with applying bilateral filter on the depth map. We can notice slight improvement on the rendered results.

6. Conclusion

In this project, we developed a framework for applying synthetic blur for all-in-focus image that usually captured by mobile devices with tiny apertures, it is capable of selecting focal plane and configure arbitrary aperture level. In the beginning, we narrowed our solution to be applicable for in-the-wild photos that does not need specific hardware requirement such as duel-pixel sensors. Our method utilized trained encoder-decoder model proposed by Alhashim *et al*. [4] for monocular depth estimation, then we applied edge aware filtering on the estimated depth map and calculate circle of confusion for each pixel. When applying blur, we followed and re-implemented methods purposed by Wadhwa *et al*. [29] to get a smooth transition between blur and achieve fast computation. Our results showed nice bokeh for randomly selected in-th-wild images. Though our project is build and tested with server level computation resources, it is feasible to deploy on edge devices and developed as application on mobile phone.

References

[1] Computational photography: Pinholes and lenses. 0(0), 2022. 5

[2] Abdullah Abuolaim, Mauricio Delbracio, Damien Kelly, Michael S Brown, and Peyman Milanfar. Learning to reduce defocus blur by realistically modeling dual-pixel data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2289–2298, 2021. 1

[3] Abdullah Abuolaim, Abhijith Punnappurath, and Michael S Brown. Revisiting autofocus for smartphone cameras. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 523–537, 2018. 2

[4] Ibraheem Alhashim and Peter Wonka. High quality monocular depth estimation via transfer learning. *arXiv preprint arXiv:1812.11941*, 2018. 3, 5, 6, 8

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 3

[6] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE international conference on computer vision*, pages 2650–2658, 2015. 3

[7] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *Advances in neural information processing systems*, 27, 2014. 3

[8] Jose M Facil, Benjamin Ummenhofer, Huizhong Zhou, Luis Montesano, Thomas Brox, and Javier Civera. Camconvs: Camera-aware multi-scale convolutions for single-view depth. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11826–11835, 2019. 3

[9] Paolo Favaro. Recovering thin structures via nonlocal-means regularization with application to depth from defocus. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1133–1140. IEEE, 2010. 2

[10] Rahul Garg, Neal Wadhwa, Sameer Ansari, and Jonathan T Barron. Learning single camera depth estimation using dual-pixels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7628–7637, 2019. 2

[11] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012. 3

[12] Clemens Holzmann and Matthias Hochgatterer. Measuring distance with mobile phones using single-camera stereo vision. In *2012 32nd International Conference on Distributed Computing Systems Workshops*, pages 88–93, 2012. 1

[13] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 3, 5

[14] Andreas Kolb, Erhardt Barth, Reinhard Koch, and Rasmus Larsen. Time-of-flight cameras in computer graphics. In

Computer Graphics Forum, volume 29, pages 141–159. Wiley Online Library, 2010. 1

[15] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)*, pages 239–248. IEEE, 2016. 3

[16] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2noise: Learning image restoration without clean data. *arXiv preprint arXiv:1803.04189*, 2018. 4

[17] John McCormac, Ankur Handa, Stefan Leutenegger, and Andrew J Davison. Scenenet rgb-d: Can 5m synthetic images beat generic imagenet pre-training on indoor segmentation? In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2678–2687, 2017. 3

[18] Sylvain Paris, Pierre Kornprobst, Jack Tumblin, Frédo Durand, et al. Bilateral filtering: Theory and applications. *Foundations and Trends® in Computer Graphics and Vision*, 4(1):1–73, 2009. 4, 8

[19] Kyeong-Sik Park and Seok-Keun Choi. Evaluation of the quantitative practical use of smart phone stereo cameras. *Journal of Korean Society for Geospatial Information Science*, 20(2):93–100, 2012. 1

[20] Alex Paul Pentland. A new sense for depth of field. *IEEE transactions on pattern analysis and machine intelligence*, (4):523–531, 1987. 2

[21] Abhijith Punnappurath, Abdullah Abuolaim, Mahmoud Afifi, and Michael S Brown. Modeling defocus-disparity in dual-pixel sensors. In *2020 IEEE International Conference on Computational Photography (ICCP)*, pages 1–12. IEEE, 2020. 2

[22] AN Rajagopalan and Subhasis Chaudhuri. Optimal selection of camera parameters for recovery of depth from defocused images. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 219–224. IEEE, 1997. 2

[23] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3234–3243, 2016. 3

[24] Ashutosh Saxena, Sung Chung, and Andrew Ng. Learning depth from single monocular images. *Advances in neural information processing systems*, 18, 2005. 3

[25] Ashutosh Saxena, Jamie Schulte, Andrew Y Ng, et al. Depth estimation using monocular and stereo cues. In *IJCAI*, volume 7, pages 2197–2203, 2007. 3

[26] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European conference on computer vision*, pages 746–760. Springer, 2012. 3, 5, 6

[27] Supasorn Suwajanakorn, Carlos Hernandez, and Steven M Seitz. Depth from focus with your mobile phone. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3497–3506, 2015. 2

[28] Huixuan Tang, Scott Cohen, Brian Price, Stephen Schiller, and Kiriakos N Kutulakos. Depth from defocus in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2740–2748, 2017. 2

[29] Neal Wadhwa, Rahul Garg, David E Jacobs, Bryan E Feldman, Nori Kanazawa, Robert Carroll, Yair Movshovitz-Attias, Jonathan T Barron, Yael Pritch, and Marc Levoy. Synthetic depth-of-field with a single-camera mobile phone. *ACM Transactions on Graphics (ToG)*, 37(4):1–13, 2018. 1, 2, 4, 5, 8

[30] Lijun Wang, Xiaohui Shen, Jianming Zhang, Oliver Wang, Zhe Lin, Chih-Yao Hsieh, Sarah Kong, and Huchuan Lu. Deeplens: shallow depth of field from a single image. *arXiv preprint arXiv:1810.08100*, 2018. 6

[31] Shumian Xin, Neal Wadhwa, Tianfan Xue, Jonathan T Barron, Pratul P Srinivasan, Jiawen Chen, Ioannis Gkioulekas, and Rahul Garg. Defocus map estimation and deblurring from a single dual-pixel image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2228–2238, 2021. 1

[32] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1983–1992, 2018. 3

[33] Yinda Zhang, Neal Wadhwa, Sergio Orts-Escolano, Christian Häne, Sean Fanello, and Rahul Garg. Du 2 net: Learning depth estimation from dual-cameras and dual-pixels. In *European Conference on Computer Vision*, pages 582–598. Springer, 2020. 1