Exploring Positional Characteristics of Dual-Pixel Data for Camera Autofocus
- *Supplementary Document* -

Figure A: Example of lens shading on white reference background and real examples. Note how the background white wall becomes darker near the edge of the image.

A. Notes on Real-world Scenario

Dual-pixel data captured using commodity mobile phones has high resolution (usually >12MP these days) and goes through sophisticated sensor/AP ISP pipeline to restore and enhance the image quality. However, an autofocus module is typically placed very early in the ISP pipeline, and the lens control signal should be computed using the RAW images, making the AF problem prone to spatially-varying errors including lens shading, low-light noise, optical aberration, *etc*. In Fig. A, we show an example of lens shading effects for (dual-pixel) RAW images captured with a smartphone camera, which can be easily seen by capturing a white reference background. We also visualize a real-world sample RAW image captured with a Samsung Galaxy S10 device[1] (different from our training data, which is captured with a Google Pixel 3). We can clearly observe the lens shading effects on the background white wall - the intensity of the pixel values is spatially varying. This is one of the major problems in the camera manufacturing industry, and the capability of managing spatially-varying sources of errors is an important problem in real-world scenarios.

B. Notes on Training Data

In this work, we use the public dataset introduced in Herrmann *et al*. [12]. Let us denote the dataset as L2A (from the name of [12], "Learning to Autofocus"). While the detailed descriptions can be found in [12], here we note some important characteristics about the dataset to eliminate any potential confusion.

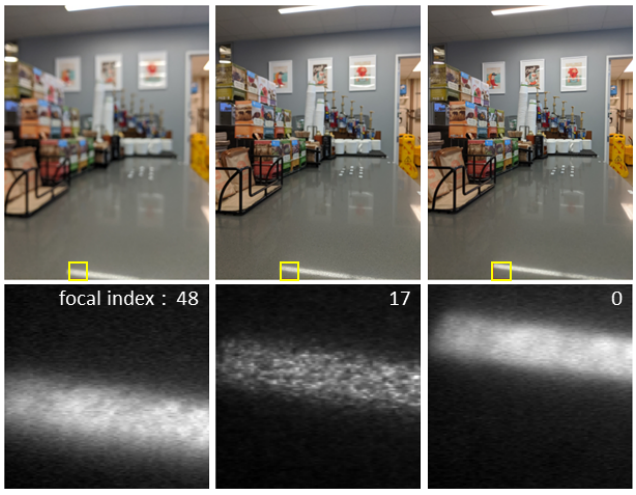
First, L2A dataset is captured vertically, but the left-right dual pixels are split along the direction of the longer edge of the smartphone. In other words, the literal meaning of *left* and *right* pixels is assuming a *landscape* mode of photo capture; if the camera is placed vertically (also called *portrait* orientation), then the dual pixels calculate the disparity along the vertical direction. We acknowledge that all of our figures except Fig. 8(d) use the original photos captured in portrait orientation, so the phase difference is detected along the vertical direction. Figure 8(d), on the other hand, is the only exception where the image (patch) is displayed by rotating 90 degrees counter-clockwise. The reason for this choice is to emphasize a common failure case of left/right dual-pixel images, which is *horizontal* lines (given that a photo is captured in landscape orientation). Note that a line that is parallel to the stereo baseline is fundamentally impossible to calculate the disparity, and Fig. 8(d) is a good example of such a difficult case.

Second, the ground truth label of L2A dataset is calculated with a multi-view stereo algorithm (COLMAP[2] [51, 52]) from synchronized captures from 5 Google Pixel 3 smartphones, so there exist some labeling errors. Figure B illustrates a representative example of such errors. Although we train and evaluate with only the *valid* patches where the confidence of multi-view stereo calculation (COLMAP) is over the threshold, small amount of label errors like Fig. B are inevitable and may affect our final performance. In general, we observed labeling errors for regions with little or no textures, and for regions where focal breathing effect is more evident.

Third, we point out that the focal indices of 1∼49 are not uniformly distributed across the possible focus distance or the lens position, and the values are already accounted for the near/far depth-of-field limits (with some overlaps). We plot the relationship between the focus distance (in millimeters) and the focal index in Fig. C. The distance considered in the L2A dataset ranges from 102mm ∼ 3910mm, where

[1] Following the data capturing process in [12], we obtain the dual-pixel RAW image with the same Android Camera app settings.

[2] https://colmap.github.io/



(a) Input (b) Our Prediction (c) GT (from MVS)

Figure B: A representative example of labeling errors in the L2A dataset. The input focal index of 48 corresponds to 10.2cm focus distance, and the (calculated) ground truth label of 0 corresponds to 3.9m physical depth. However, given that the region of interest is the closest region of the floor (marked with a yellow box), the actual depth is more likely to be closer to our prediction, of which the index 17 corresponds to 27.4cm distance from the camera lens.

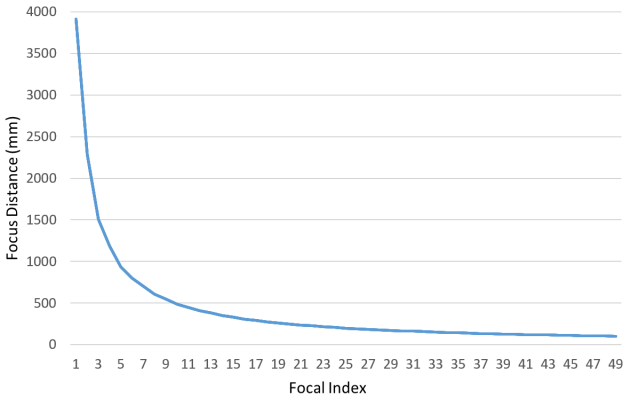


Figure C: Relationship between focal index ($1 \sim 49$) vs. focus distance (in millimeters). The values are inversely proportional to each other.

the focal index of 1 corresponds to the furthest focus distance and the index 49 corresponds to the macro-shot capture of 102mm distance. While we illustrate the focal index vs. distance in Fig. C, this relationship is actually inversely proportional, and we can observe a linear graph if we take a reciprocal of each distance value.

For more specific information on the dataset capturing process and the calibration parameters, we refer the readers to the dataset README[3] of [12].

Here, we note that the choice of 49-index lens position is not directly applicable when our input RAW image is captured with a different device. However, we can easily fix the input preprocessing pipeline to make it applicable:

- If the target device uses the same image sensor (as Google Pixel 3 in case of [12]), then we can perform a simple calibration of the lens position to match our 49-index, which rectifies the fabrication bias.
- If the target device includes a different image sensor, we also have to perform calibration to map the lens position into our 49 focal indices. Since each focal index represents a corresponding focus distance, we can always perform device-dependent calibration to map the lens position to match our 49-index.

C. Baseline Model Architecture

We illustrate the baseline network architectures used in our AF model in Fig. D. We borrow the drawing convention in MCUNet [21]: 1) the inverted residual block from MobileNet-v2 [32] is named as `MB{expansion_ratio}{k}x{k}`, where `k` is the convolution kernel size, 2) the blocks with larger kernel size are colored darker, 3) the striped boundaries denote the downsampling blocks with `stride=2`, and 4) the width of each block corresponds to the expansion ratio of the MB block. Note that our models use 5-channel inputs, while Herrmann *et al*. [12] uses 98-channel inputs (for dual-pixel data), which enabled us to use $\times\frac{1}{4}$ number of channels for our MobileNet-based model. When it comes to our MCUNet-based model, the number of channels are reduced more extremely, and the average expansion ratio for the MB blocks also gets smaller (which also has the same effect of reducing the # of channels, since *e.g*. MB5 with 24 input/output channels includes convolution layers with 1) 24 to 120 (=24$\times$5) channels and 2) 120 to 24 channels).

D. Additional Results

Additional failure cases of our model are illustrated in Fig. E.

References

[51] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[52] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. 1

[3] https://learntoautofocus-google.github.io/#data

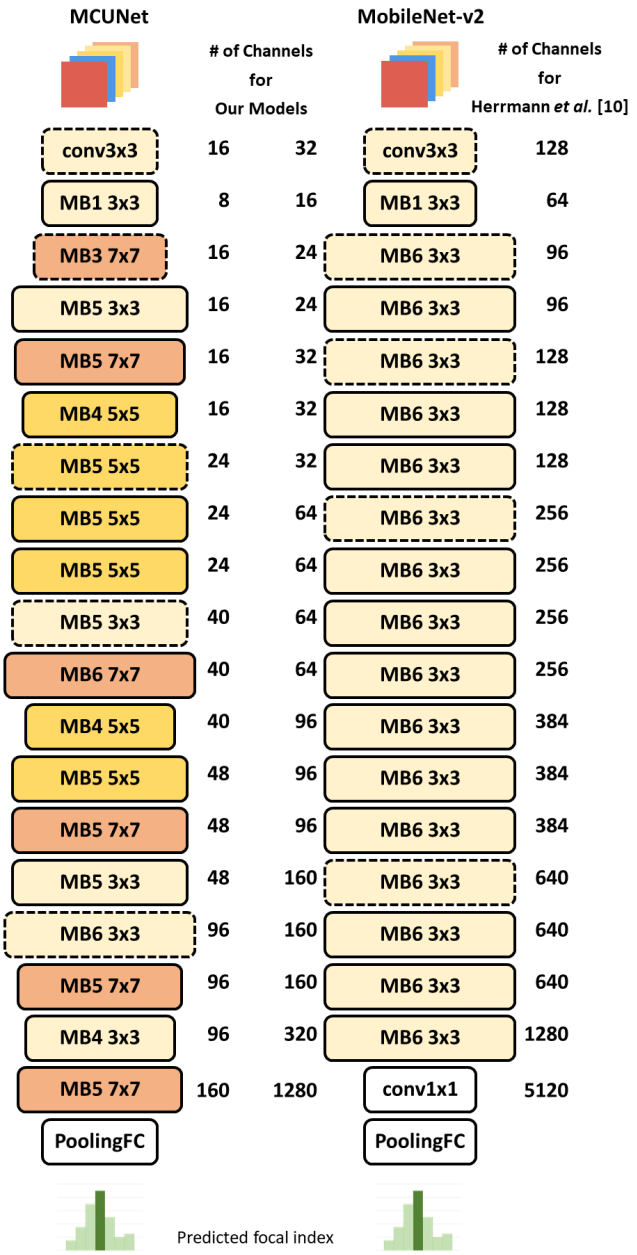


Figure D: Network architecture comparison. Our models (both MobileNet-v2 and MCUNet) significantly reduce the number of channels, resulting in much smaller number of parameters, FLOPs, and peak memory usage. The main building blocks are the inverted residual block from MobileNet-v2, and we express this block as `MB{expansion_ratio} {k}x{k}`, where `k` is the convolution kernel size. The blocks with larger kernel size are colored darker, and the striped boundaries denote the downsampling blocks with `stride=2`.

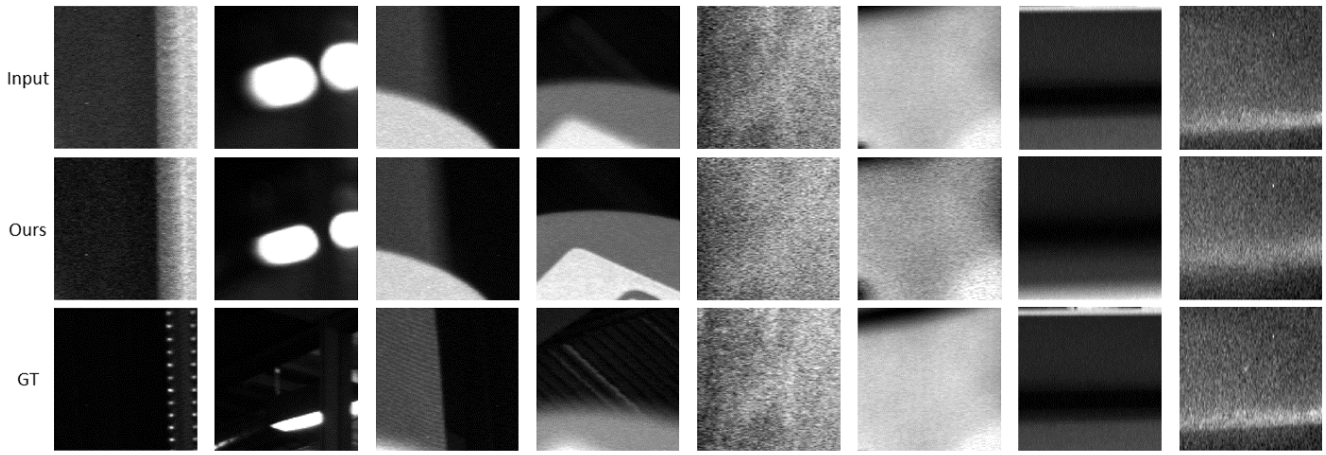


Figure E: Additional failure cases of our AF model. We visualize the *left* image of the dual-pixel images.