Improving Bokeh Simulation on CPUs: Faster Inference and Better Perception

Chia-Lin Chang[1], Hao-Cheng Hsu[1], Cen-En Jian[1], Yu-Hui Huang[2]

Department of Electrical Engineering
Yuan Ze University, 32001, Taoyuan, Taiwan
e-mail : s1103824@mail.yzu.edu.tw

Abstract - We present a set of lightweight optimization techniques to accelerate bokeh simulation on CPUs while improving perceptual quality. By integrating XLA compilation, multiprocessing, edge-aware interpolation, and weight pruning, our method achieves a 7.3% speedup over the baseline and improves both PSNR and SSIM. The results demonstrate practical value for CPU-only deployment scenarios.

I. Introduction

Bokeh effect is a significant visual phenomenon in photography and imaging science. It refers to the quality of the blur and the shape of the out-of-focus areas (usually the background or foreground) when photographing with a shallow depth of field. This effect effectively separates the subject from the background, thereby enhancing the visual focus and narrative tension of the image.

In recent years, the rapid advancement of hardware technology in mobile devices has resulted in an increasing number of users opting for lightweight and convenient mobile phones. With the application of neural network models, mobile phones can also simulate shooting effects close to those of DSLR, and bokeh effect being one of the most popular options. This study focuses on improving the performance and imaging quality of such mobile phone image processing systems running on CPUs. Four common neural network optimization methods are explored and implemented, and the study analyzes and compares two aspects: accelerating the inference process and improving output image quality.

In this study, we selected BGGAN (Bokeh-Glass Generative Adversarial Network) [1] as our baseline for optimization. This model achieved first place in the AIM 2020 (Advanced Image Manipulation) challenge [6]. The architecture of the glass net is an end-to-end network consisting of two components that are similar to U-Net [2].

II. Implementation Details

To improve both the runtime efficiency and output quality of bokeh simulation on CPUs, we incorporated four optimization techniques. Two of these target computational acceleration, while the other two focus on enhancing visual fidelity. Below, we detail each method and its implementation.

A. XLA Compilation for Inference Optimization

We employed XLA (Accelerated Linear Algebra) [3] to optimize the model's computational graph prior to execution. Through just-in-time compilation and operator fusion, XLA generates low-level machine code tailored for the target CPU architecture. This optimization significantly reduces memory access overhead and improves inference throughput. In our implementation, XLA was activated via the JAX backend to accelerate convolution-heavy operations in the bokeh simulation pipeline.

B. Parallel Preprocessing via Multiprocessing

Image preprocessing, including edge detection and interpolation, accounts for a significant portion of CPU load. To mitigate this bottleneck, we adopted Python's multiprocessing.Pool module. Tasks were distributed across multiple subprocesses, each independently loading the model and processing a subset of the input images. This design avoids contention with the Python Global Interpreter Lock (GIL) and achieves speedup proportional to the number of CPU cores.

C. Edge-Aware Mixed Interpolation

To better preserve object contours and texture transitions in the simulation output, we introduced a region-aware interpolation strategy. The input image is first segmented using the Canny edge detector to generate a binary edge mask.

- For edge regions, nearest-neighbor interpolation is applied to maintain structural sharpness.
- For flat regions, bicubic interpolation is used to ensure smooth transitions.

The two interpolated results are then fused on a per-pixel basis using the edge mask. This hybrid interpolation technique helps maintain fidelity in detailed areas while reducing artifacts in homogeneous regions.

D. Lightweight Model via Magnitude-Based Pruning

To reduce the model's parameter count and inference cost, we applied magnitude-based unstructured pruning. We gradually increased the sparsity threshold and removed low-magnitude weights across each layer without subsequent fine-tuning. Although this method did not provide significant speedup—due to CPU-based dense matrix operations—the resulting sparsity contributed to increased robustness and improved visual consistency in the output. This regularization-like effect was especially helpful in suppressing noise and enhancing depth boundaries in the simulated bokeh.

III. Experiments and Results

All experiments were conducted on an Intel i5-12400 processor using a test set of 200 photographs.

Table I summarizes the performance of the proposed "MIX" configuration, which integrates all four optimization techniques. Compared to the original baseline model, MIX achieves a 7.3% reduction in processing time and improvements in both PSNR(↑ 0.25 dB) and SSIM（ ↑ 0.0083）.

TABLE I

The performance of Comprehensive combination

Model	Time(s)	PSNR	SSIM
Baseline	1.5050	23.09	0.7785
MIX	**1.3951**	**23.34**	**0.7868**

TABLE II

Ablation study (each optimization and combination)

Model	Times(s)	PSNR	SSIM
Baseline	1.5050	23.09	0.7785
XLA	1.4541	23.09	0.7785
Multiprocessing	1.2549	23.09	0.7785
Mixed Interpolation	1.5171	23.23	0.7820
Pruning	1.5050	23.20	0.7833
XLA+ Multiprocessing	**1.2496**	23.09	0.7785
Mixed Interpolation+ Pruning	1.6212	23.32	0.7860
MIX	1.3951	**23.34**	**0.7868**

Table II presents an ablation study evaluating the individual and combined effects of each optimization. Clearly, multiprocessing contributes the most to speedup, while mixed interpolation and pruning lead to noticeable gains in image quality. The combination of XLA and multiprocessing offers the best runtime efficiency, whereas combining interpolation and pruning yields the highest perceptual improvements. The full "MIX" configuration demonstrates that integrating these complementary techniques yields the best balance between speed and visual quality.

IV. Summary and Conclusions

This study demonstrates that strategically combining multiple neural network optimization techniques can significantly enhance both the performance and visual quality of bokeh simulation on CPU-based systems. Among all configurations, the proposed MIX approach achieved the highest PSNR and SSIM scores while maintaining efficient processing time. These results highlight the feasibility of delivering high-quality computational photography on resource-constrained platforms, offering a practical reference for CPU-based image processing in mobile devices.

References

[1] Qian, Ming, et al. "Bggan: Bokeh-glass generative adversarial network for rendering realistic bokeh." *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*. Springer International Publishing, 2020.

[2] Ronneberger, Olaf. et al. "U-Net: Convolutional networks for biomedical image segmentation." Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015: 18th International Conference, Munich, Germany, October 5–9, 2015, Proceedings, Part III 18. Springer International Publishing, 2015.

[3] Snider, Daniel, and Ruofan Liang. "Operator fusion in XLA: analysis and evaluation." *arXiv preprint arXiv:2301.13062* (2023).

[4] Lin, Chun-Feng. "A Study on Image Compression Using SVD Combined with Super-Resolution Interpolation Methods." Journal of Science and Engineering Technology, vol. 18, no. 1, 2022, pp. 25–34. Airiti Library, accessed 15 May 2025.

[5] Han, Song, et al. "Learning both weights and connections for efficient neural network." *Advances in neural information processing systems* 28 (2015).

[6] AIM 2020 Challenge. Advanced Image Manipulation Challenge 2020. https://data.vision.ee.ethz.ch/cvl/aim20/ (accessed May 20, 2025)