Single-step beam intensity and profile optimization using a 256×256 micromirror array and reinforcement learning: supplemental document

The PPO implementation is adapted from the Stable-Baselines3 library [1]. The mapping function comprises three components: a shared block, an actor block, and a critic block. The structures of these blocks are detailed in Table S2, Table S3, and Table S4, respectively. The design of the mapping function is inspired by the ConvNeXt-v2 architecture [2], with its basic block structure provided in Table S1. The hyperparameters used for training the models are detailed in Table S5.

The output of the actor has a shape of 2×256×256, to which a pixel-wise Softmax function is applied. The Softmax function is defined as $Softmax(x\_{i})=\frac{e^{x\_{i}}}{\sum\_{j=1}^{n}e^{x\_{j}}}$ for $x\_{i}$ where $i\in Z^{+}$ and $0<i\leq n$ with $n=2$. This function maps the pixel-wise pair of real numbers from the actor output into a pair of values strictly bounded between 0 and 1 (i.e., $Sotfmax\left(∙\right):R^{2}\rightarrow \left(0,1\right)^{2}$). Consequently, both $p$ and $1-p$ are parameterized for a Bernoulli distribution. This approach differs from the common method of using a Sigmoid function (i.e., $Sigmoid\left(x\right)=\frac{1}{1+e^{-x}}$), which requires only a single parameter $p$ to parameterize a Bernoulli distribution. Our empirical study suggests that the Softmax approach leads to more stable training, potentially due to the additional parameterization of $1-p$ which offers enhanced stability against random gradient fluctuations during training.

The training curves for both the “flat” model and the “UoS” model are presented in Fig. S1. In this figure, the reward is normalized by the total number of micromirrors (i.e., 65536), making it a measure of the average pixel-wise absolute difference between $I\_{mod}$ and $I\_{target}$.

Table S1. ConvNeXt Block

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Input Shape | Output Shape | Specifications |
| Depth-wise Conv Layer | $$n×m×m$$ | $$n×m×m$$ | Conv $n×n×7×7$, Stride 1, Groups $n$ |
| Layer Norm | $$n×m×m$$ | $$n×m×m$$ | - |
| Point-wise Conv Layer | $$n×m×m$$ | $$4n×m×m$$ | Conv $4n×n×1×1$, Stride 1, Groups 1 |
| GELU | $$4n×m×m$$ | $$4n×m×m$$ | - |
| GRN | $$4n×m×m$$ | $$4n×m×m$$ | - |
| Point-wise Conv Layer | $$4n×m×m$$ | $$n×m×m$$ | Conv $n×4n×1×1$, Stride 1, Groups 1 |
| Skip Connection | $$n×m×m$$ | $$n×m×m$$ | - |

Table S2. Network Structure of the Shared Layers

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Input Shape | Output Shape | Specifications |
| Conv Layer | 1(2)×512×512 | 16×512×512 | Conv 1(2)×16×3×3, Stride 1, Groups 1 |
| Layer Norm | 16×512×512 | 16×512×512 | - |
| ConvNeXt Block | 16×512×512 | 16×512×512 | - |
| Layer Norm | 16×512×512 | 16×512×512 | - |
| Conv Layer | 16×512×512 | 32×256×256 | Conv 16×32×2×2, Stride 2, Groups 1 |
| ConvNeXt Block | 32×256×256 | 32×256×256 | - |

Table S3. Network Structure of the Actor

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Input Shape | Output Shape | Specifications |
| ConvNeXt Block | 32×256×256 | 32×256×256 | - |
| ConvNeXt Block | 32×256×256 | 32×256×256 | - |
| Actor Output Layer | 32×256×256 | 2×256×256 | Conv 32×2×1×1, Stride 1, Groups 1 |

Table S4. Network Structure of the Critic

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Input Shape | Output Shape | Specifications |
| Layer Norm | 32×256×256 | 32×256×256 | - |
| Conv Layer | 32×256×256 | 64×128×128 | Conv 32×64×2×2, Stride 2, Groups 1 |
| ConvNeXt Block | 64×128×128 | 64×128×128 | - |
| Layer Norm | 64×128×128 | 64×128×128 | - |
| Conv Layer | 64×128×128 | 128×64×64 | Conv 64×128×2×2, Stride 2, Groups 1 |
| ConvNeXt Block | 128×64×64 | 128×64×64 | - |
| Layer Norm | 128×64×64 | 128×64×64 | - |
| Conv Layer | 128×64×64 | 256×32×32 | Conv 256×128×2×2, Stride 2, Groups 1 |
| ConvNeXt Block | 256×32×32 | 256×32×32 | - |
| Layer Norm | 256×32×32 | 256×32×32 | - |
| Conv Layer | 256×32×32 | 512×16×16 | Conv 512×256×2×2, Stride 2, Groups 1 |
| ConvNeXt Block | 512×16×16 | 512×16×16 | - |
| Layer Norm | 512×16×16 | 512×16×16 | - |
| Conv Layer | 512×16×16 | 1024×8×8 | Conv 1024×512×2×2, Stride 2, Groups 1 |
| ConvNeXt Block | 1024×8×8 | 1024×8×8 | - |
| Layer Norm | 1024×8×8 | 1024×8×8 | - |
| Conv Layer | 1024×8×8 | 2048×4×4 | Conv 2048×1024×2×2, Stride 2, Groups 1 |
| ConvNeXt Block | 2048×4×4 | 2048×4×4 | - |
| Layer Norm | 2048×4×4 | 2048×4×4 |  |
| Conv Layer | 2048×4×4 | 2048×1×1 | Conv 1×2048×4×4, Stride 1, Groups 1 |
| Critic Output Layer | 2048×1×1 | 1 | Fully Connected |

Table S5. PPO hyperparameters

|  |  |
| --- | --- |
| Hyperparameters Name | Hyperparameters Value |
| Batch Size | 16 |
| Clip Range | 0.2 |
| Entropy Coefficient | 0.0 |
| GAE λ | 0.0 |
| Discount Factor γ | 0.0 |
| Learning Rate | Anneal from 3e-4 to 1e-5 |
| Maximum Gradient Clipping | 0.5 |
| Value Function Coefficient | 1.0 |
| Total Training Steps | 1e6 |



Fig. S1. Training curves for the “flat” model (red) and for the “UoS” model (blue).

References

1. A. Raffin, et al., “Stable-Baselines3: Reliable Reinforcement Learning Implementations,” J Mach Learn Res **22**, 1-8 (2021).

2. S. Woo, et al., “ConvNeXt V2: Co-designing and Scaling ConvNets with Masked Autoencoders,” 2023 Ieee/Cvf Conference on Computer Vision and Pattern Recognition (Cvpr), 16133-16142 (2023).