Light Field Display Optimizer: Technical Documentation

Enhanced Multi-Ray Sampling Implementation

August 25, 2025

1 Overview

The Light Field Display Optimizer is a PyTorch-based system that optimizes display patterns for light field displays using multi-ray sub-aperture sampling and realistic optical physics simulation. The system optimizes display images to recreate target scenes as viewed through a complete optical system consisting of eye optics, tunable focus lens, and microlens array.

2 Optical System Model

2.1 Eye Optics

The human eye is modeled with the following parameters:

- **Pupil diameter**: 4.0 mm
- **Retina distance**: 24.0 mm (from eye lens)
- **Retina size**: 10.0 mm effective imaging area
- **Focal length range**: 17.0 - 60.0 mm (accommodation)

2.2 Multi-Ray Sub-Aperture Sampling

The system implements realistic depth-of-field through multi-ray sampling:

$$\text{Ray origins} = \text{Pupil samples} \times N_{\text{rays}} \tag{1}$$

$$\text{Ray directions} = \frac{\text{Pupil point} - \text{Retina point}}{|\text{Pupil point} - \text{Retina point}|} \tag{2}$$

Where $N_{\text{rays}} = 16$ rays per pixel for realistic blur simulation.

2.3 Tunable Focus Lens

A tunable lens is positioned 50.0 mm from the eye with:

- **Diameter**: 15.0 mm
- **Focal range**: 10.0 - 100.0 mm
- **Thin lens equation**: $\frac{1}{f} = \frac{1}{s_o} + \frac{1}{s_i}$

2.4 Microlens Array

The microlens array parameters:

- **Distance from eye**: 80.0 mm
- **Array size**: 20.0 mm × 20.0 mm
- **Pitch**: 0.4 mm (center-to-center spacing)
- **Focal length**: 1.0 mm
- **Total microlenses**: 2,500 circular lenses

3 Display System

3.1 Light Field Display

The optimizable display consists of:

$$\mathbf{D} = \{D_1, D_2, \ldots, D_{N_f}\} \tag{3}$$
$$D_i \in \mathbb{R}^{3\times H\times W} \tag{4}$$

Where:

- $N_f = 10$ focal planes
- $H = W = 1536$ pixels (display resolution)
- Each D_i represents RGB display image for focal plane i

3.2 Focal Length Assignment

Fixed focal lengths are assigned linearly:

$$f_i = f_{\min} + \frac{i-1}{N_f - 1}(f_{\max} - f_{\min}) \tag{5}$$

Where $f_{\min} = 10$ mm and $f_{\max} = 100$ mm.

4 Scene Definitions

4.1 Geometric Scenes

Six geometric scenes with parametric object definitions:

$$\text{Object}_j = \{pos_j, size_j, color_j, shape_j\} \tag{6}$$
$$pos_j \in \mathbb{R}^3 \text{ (world coordinates in mm)} \tag{7}$$
$$color_j \in [0,1]^3 \text{ (RGB values)} \tag{8}$$

4.2 Spherical Checkerboard Scene

A MATLAB-compatible spherical checkerboard with:

$$\text{Sphere center} = [0, 0, 200] \text{ mm} \quad (9)$$
$$\text{Radius} = 50 \text{ mm} \quad (10)$$
$$\text{Pattern} = 1000{\times}1000 \text{ grid, 50 pixels per square} \quad (11)$$

The checkerboard pattern is mapped using spherical coordinates:

$$\phi = \arctan 2(Z, X) \quad (12)$$
$$\theta = \arctan 2(Y, \sqrt{X^2 + Z^2}) \quad (13)$$
$$\text{Square}_{i,j} = \lfloor \frac{\theta_{\text{norm}} \times 999}{50} \rfloor + \lfloor \frac{\phi_{\text{norm}} \times 999}{50} \rfloor \bmod 2 \quad (14)$$

5 Ray Tracing Algorithm

5.1 Forward Ray Tracing

The system uses forward ray tracing from retina through the complete optical system:

1. **Retina sampling**: Generate N_{pixels} retina points
2. **Pupil sampling**: Generate N_{rays} pupil samples per retina point
3. **Eye lens refraction**: Apply thin lens equation
4. **Tunable lens refraction**: Secondary lens refraction
5. **Microlens selection**: Find nearest microlens for each ray
6. **Microlens refraction**: Apply microlens optical power
7. **Display sampling**: Sample from appropriate display image

5.2 Ray-Sphere Intersection

For spherical checkerboard scenes, ray-sphere intersection:

$$\mathbf{oc} = \mathbf{ray\_origin} - \mathbf{sphere\_center} \quad (15)$$
$$a = \mathbf{ray\_dir} \cdot \mathbf{ray\_dir} \quad (16)$$
$$b = 2(\mathbf{oc} \cdot \mathbf{ray\_dir}) \quad (17)$$
$$c = \mathbf{oc} \cdot \mathbf{oc} - r^2 \quad (18)$$
$$\text{discriminant} = b^2 - 4ac \quad (19)$$
$$t = \frac{-b + \sqrt{\text{discriminant}}}{2a} \quad (20)$$

6 Optimization Process

6.1 Loss Function

The optimization minimizes mean squared error between target and simulated images:

$$\mathcal{L} = \frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} ||\mathbf{I}_{\text{target}}(i,j) - \mathbf{I}_{\text{simulated}}(i,j)||^2 \tag{21}$$

6.2 Optimization Algorithm

- **Optimizer**: AdamW with learning rate 0.02
- **Weight decay**: 10^{-4}
- **Gradient clipping**: Maximum norm 1.0
- **Mixed precision**: Automatic mixed precision (AMP) for A100 acceleration
- **Iterations**: 100 per scene

6.3 Multi-Scene Training

The system optimizes seven distinct scenes:

1. **Basic**: 3 colored spheres at different depths
2. **Complex**: 4 multi-colored spheres in complex arrangement
3. **Stick Figure**: 6 spheres arranged as humanoid figure
4. **Layered**: 3 spheres at different depth layers
5. **Office**: 4 spheres representing office objects
6. **Nature**: 4 spheres representing outdoor scene
7. **Spherical Checkerboard**: MATLAB-compatible spherical pattern

7 Output Generation

7.1 Training Progress Visualization

For each scene, the system generates:

- **Progress GIF**: 100 frames showing optimization evolution
- **Target image**: Ground truth scene appearance
- **Simulated image**: Optimized display output
- **Loss curve**: Convergence visualization

7.2 Display Analysis

- **Display images**: What each focal plane shows (10 planes per scene)
- **Eye views**: What the eye sees for each display focal length
- **Focal length mapping**: 10-100 mm range across display planes

7.3 Global Analysis

- **Focal length sweep**: 100 frames showing accommodation effects
- **Eye movement sweep**: 60 frames showing parallax effects
- **Focus calculation**: $d_{\text{focus}} = \frac{f \times d_{\text{retina}}}{f - d_{\text{retina}}}$

8 Technical Implementation

8.1 Memory Optimization

- **Batch processing**: 4096 pixels per batch
- **GPU memory management**: Automatic cache clearing
- **Mixed precision**: FP16/FP32 automatic casting
- **Gradient accumulation**: Per-scene optimization

8.2 Computational Complexity

$$\text{Rays per optimization} = N_{\text{pixels}} \times N_{\text{rays}} \times N_{\text{iterations}} \tag{22}$$
$$= 512^2 \times 16 \times 100 = 419,430,400 \text{ rays} \tag{23}$$

8.3 Hardware Requirements

- **GPU**: NVIDIA A100-SXM4-80GB
- **Memory usage**: 2.12 GB peak
- **Compute capability**: CUDA 11.8+
- **Storage**: 15 GB container disk

9 Results and Validation

The optimization system successfully generates:

- **28 individual files**: 4 files per scene × 7 scenes
- **Progress tracking**: Every iteration recorded
- **Multi-ray sampling**: Realistic depth-of-field effects
- **Optical accuracy**: Physical ray tracing through complete system