Optical Simulation Enhancements

Functionality Enhancements

Feature	Difficulty Level
Implement phase shifters for phase gates	Easy
Add support for polarizing beam splitters (PBS)	Easy
Implement CNOT gate using PBS	Intermediate
Add support for multi-qubit gates (Toffoli, Fredkin)	Hard
Simulate Kerr nonlinearity (photon-photon interactions)	Hard
Add support for single-photon sources and detectors	Intermediate
Implement quantum teleportation circuit	Intermediate
Implement quantum error correction (3-qubit bit-flip code)	Hard
Simulate entangled photons	Intermediate
Simulate the Hong-Ou-Mandel effect	Hard

User Experience Improvements

Feature	Difficulty Level
Add snap-to-grid for drag-and-drop UI	Easy
Implement undo/redo functionality	Intermediate
Provide real-time setup validation	Intermediate
Implement interactive simulation (pause, resume, step-through)	Intermediate
Add 3D visualization for optical setup	Hard
Animate the state vector evolution	Hard
Visualize interference patterns	Intermediate
Add built-in interactive tutorials	Intermediate
Implement tooltips and documentation	Easy

Performance Optimizations

Feature	Difficulty Level
Use sparse matrices for state vector calculations	Intermediate
Parallelize matrix operations for faster simulations	Hard
Optimize graph traversal (cycle detection, path mode reduction)	Hard
Implement garbage collection for memory management	Easy
Compress state vector representation	Hard

Future-Proofing

Feature	Difficulty Level
Design a modular plugin system	Hard
Provide API for external tools (Qiskit, Cirq integration)	Hard
Develop a web-based interface (Flask/Django)	Hard
Optimize GUI for mobile compatibility	Intermediate
Integrate real quantum hardware backend	Hard
Support hybrid quantum-classical simulations	Hard

Advanced Features

Feature	Difficulty Level
Implement noise models (photon loss, decoherence)	Hard
Provide error analysis tools	Hard
Simulate the BB84 quantum key distribution protocol	Hard
Implement quantum repeaters for long-distance communication	Hard
Add quantum machine learning algorithms	Hard
Use ML for automated circuit design	Hard

Testing and Debugging

Feature	Difficulty Level
Write unit tests for core functionalities	Easy
Test edge cases for robustness	Intermediate
Allow state inspection during simulation	Intermediate
Add detailed logging for debugging	Easy

Community and Collaboration

Feature	Difficulty Level
Publish project on GitHub	Easy
Write comprehensive documentation	Easy
Implement a user feedback mechanism	Intermediate
Create a community forum or Discord server	Intermediate