PhotonWeave

Simon Sekavčnik [1], **Kareem H. El-Safty** [1], **and Janis Nötzel** [1]

1 Technical University of Munich, Theoretical Quantum System Design, Munich, Germany

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

License
Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License (CC BY 4.0).

Summary

PhotonWeave is a quantum systems simulator designed to offer intuitive abstractions for simulating quantum photonic systems and their interactions in Fock spaces (Fock, 1932) and any custom Hilbert spaces. The simulator focuses on simplifying complex quantum state representations, such as continuous photonic states with polarization using envelopes that can mimic pulses, making it more approachable for specific quantum simulations. While general-purpose quantum simulation libraries such as QuTiP (Johansson et al., 2012) provide robust tools for quantum state manipulations, some require advanced software skills to manipulate complex system interactions. PhotonWeave addresses this by abstracting such details, streamlining the simulation process, and allowing quantum systems to interact naturally as the simulation progresses.

In contracts to other frameworks such as Qiskit (Javadi-Abhari et al., 2024), which are primarily designed for qubit-based computations, PhotonWeave excels at simulating continuous-variable quantum systems, mainly photons, as well as custom quantum states that can interact dynamically. Furthermore, PhotonWeave offers a balance of flexibility and automation by deferring the joining of quantum spaces using State Containers until necessary, enhancing computational efficiency. The simulator supports CPU and GPU execution, ensuring scalability and performance for large-scale simulations. This is achieved by using the JAX (Bradbury et al., 2018) library.

Statement of Need

Tools like QuTiP, Qiskit, Piquasso, StrawberryFields (Killoran et al., 2019; Kolarovszki et al., 2024), and SOQCS (Osca & Vala, 2024) already exist for modeling quantum phenomena. However, many of them either require extensive user control (QuTiP) or enforce rigid circuit structures (StrawberryFields, SOQCS). Researchers in quantum optics and related fields need a tool that simplifies photonic systems simulations and supports dynamic interactions between custom quantum systems. PhotonWeave introduces such features without restricting itself to the circuit model, allowing researchers to focus on component development. Such a tool could generate a library of components and gates that closely model real-world devices, fostering greater collaboration among scientists in these fields. In Fig. 1, a complex scenario of lossy beam splitters is depicted, and in Fig. 2, the performance is compared to Qiskit and QuTiP.

The Journal of Open Source Software

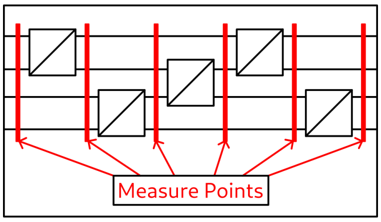

Figure 1: The simulation of lossy Beam Splitters. The simulation tracks the state evolution throughout the experiment. The losses here are photon absorption.

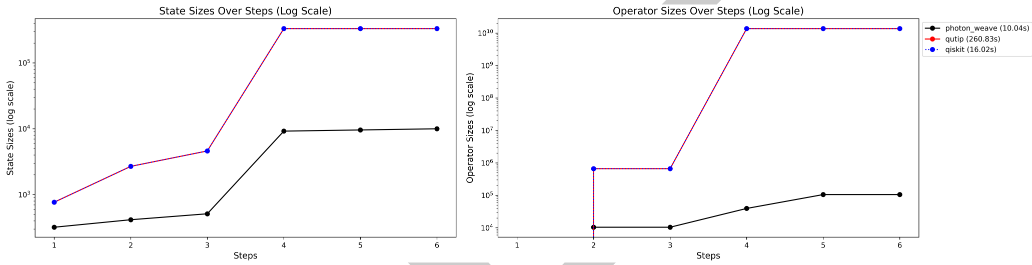

Figure 2: Comparison between PhotonWeave, Qiskit, and QuTip regarding simulation time and the required space to simulate the experiment in Figure 1. The steps are the executed operations.

DRAFT

PhotonWeave Overview

PhotonWeave is a quantum simulation library designed for simulating any system, provided that simulating hardware meets the resource requirements. This simulator allows users to easily create, manipulate, and measure quantum systems.

PhotonWeave Implementation Details

In the following sections, we will describe the main features of PhotonWeave; details about implementations and usage can be found in the documentation.

State Containers

PhotonWeave's core functionality revolves around quantum state containers. States can be represented in three forms: `Label`, `Vector`, or `Matrix`, which progressively require more memory. PhotonWeave automatically manages these representations, reducing representations where applicable to save resources. The framework provides state containers such as `Fock`, `Polarization`, `Envelope`, and `CustomState`. `Fock`, `Polarization`, and `CustomState` are essential state containers that hold the quantum state in any valid representation until the state interacts with other states. When states interact, these containers store references to the `Envelope`, `CompositeEnvelope`, or both. This allows each container to understand its place within a larger product space and how it evolves mathematically.

Envelopes

PhotonWeave places a particular emphasis on the `Envelope` concept. An `Envelope` represents a pulse of light, where all photons are indistinguishable and share the same polarization, representing the $\mathcal{F} \otimes \mathcal{P}$ space where $\mathcal{F}$ represents the Fock space and $\mathcal{P}$ represents the Polarization space. Initially, when the states are separable, they are stored in the respective `Fock` and `Polarization` containers. In addition to the states, an `Envelope` holds essential metadata such as wavelength and temporal profile.

Composite Envelopes

When envelopes interact, for example, using a beam-splitter (Xiang-bin, 2002), their states must be joined. The necessary state data are extracted from their respective containers, and their Hilbert spaces form a product space in these cases. A `CompositeEnvelope` can contain multiple product spaces, which can be accessed from any of the contributing state containers. Additionally, `CompositeEnvelope` instances can be merged, allowing states within both envelopes to interact. `CustomState` instances can also be included in a `CompositeEnvelope` since any custom state can, in principle, interact with any other state.

Operations

PhotonWeave provides several ways to perform operations on quantum states. All operations are created using the Operation type as well as one of the Enums: `FockOperationType`, `PolarizationOperationType`, `CustomStateOperationType`, and `CompositeOperationType` to further define what on which type of a state the operation will operate. Operations can be manually constructed or generated using expressions with a context along with the predefined ones. PhotonWeave supports photonic operators such as `Squeezing`, `Displacement`, `Phase Shift`, `Beam Splitter`, and non-linear operations. It also supports Pauli operators.

PhotonWeave optimizes resource usage by automatically adjusting the dimensionality of the Fock space when necessary, even within product states. This ensures that only the minimal required space is used, dynamically resizing the quantum state representation to avoid unnecessary memory consumption.

Once an operation is defined, it can be applied to an appropriate state at any level. If a state is a part of a product space, PhotonWeave ensures that the operation is applied to the correct subspace. Additionally, Kraus operators can be applied to any desired state space. This allows the user to simulate losses at any level.

Measurements

PhotonWeave offers a robust measurement framework for any state. By default, Fock spaces are measured on a number basis, Polarization spaces are measured on a computational basis, and `CustomState` is measured on a respective basis. PhotonWeave also supports more precise measurement definitions, such as Positive Operator Valued Measurement (POVM).

Conclusion

PhotonWeave is an open-source quantum system simulator under the Apache-2.0 license, targeting researchers and developers who need an easy-to-use yet powerful simulation tool. One of the intended outcomes is to build a library of interoperable quantum device models powered by the PhotonWeave framework.

Acknowledgments

The authors acknowledge the financial support by the Federal Ministry of Education and Research of Germany in the programme of "Souverän. Digital. Vernetzt.". Joint project 6G-life, project identification number: 16KISK002 and via grants 16KIS1598K, 16KISQ039, 16KISQ077, 16KISQ093 and that of the DFG via grant 1129/2-1. This research is part of the Munich Quantum Valley, which is supported by the Bavarian state government with funds from the Hightech Agenda Bayern Plus and received support from the Bavarian Ministry for Economic Affairs (StMWi) via the project 6GQT.

References

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., & Zhang, Q. (2018). *JAX: Composable transformations of Python+NumPy programs* (Version 0.3.13). http://github.com/jax-ml/jax

Fock, V. (1932). Konfigurationsraum und zweite quantelung. *Zeitschrift f r Physik*, *75*(9–10), 622–647. https://doi.org/10.1007/bf01344458

Javadi-Abhari, A., Treinish, M., Krsulich, K., Wood, C. J., Lishman, J., Gacon, J., Martiel, S., Nation, P. D., Bishop, L. S., Cross, A. W., Johnson, B. R., & Gambetta, J. M. (2024). *Quantum computing with qiskit*. https://doi.org/10.48550/arXiv.2405.08810

Johansson, J. R., Nation, P. D., & Nori, F. (2012). QuTiP: An open-source python framework for the dynamics of open quantum systems. *Computer Physics Communications*, *183*(8), 1760–1772. https://doi.org/10.1016/j.cpc.2012.02.021

Killoran, N., Izaac, J., Quesada, N., Bergholm, V., Amy, M., & Weedbrook, C. (2019). Strawberry fields: A software platform for photonic quantum computing. *Quantum*, *3*, 129. https://doi.org/10.22331/q-2019-03-11-129

Kolarovszki, Z., Rybotycki, T., Rakyta, P., Kaposi, Á., Poór, B., Jóczik, S., Nagy, D. T. R., Varga, H., El-Safty, K. H., Morse, G., Oszmaniec, M., Kozsik, T., & Zimborás, Z. (2024). *Piquasso: A photonic quantum computer simulation software platform*. https://doi.org/10.48550/arXiv.2403.04006

Osca, J., & Vala, J. (2024). SOQCS: A stochastic optical quantum circuit simulator. *SoftwareX*, *25*, 101603. https://doi.org/10.1016/j.softx.2023.101603

Xiang-bin, W. (2002). Theorem for the beam-splitter entangler. *Physical Review A*, *66*(2). https://doi.org/10.1103/physreva.66.024303

DRAFT