





PyPO: a Python package for Physical Optics

Arend Moerman ¹, Maikel H. Gafaji², Kenichi Karatsu ^{1,3}, and Akira Endo ¹

¹ Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Mekelweg 4, 2628 CD, Delft, The Netherlands ² The Hague University of Applied Sciences, Johanna Westerdijkplein 75, 2521 EN, The Hague, The Netherlands ³ SRON—Netherlands Institute for Space Research, Niels Bohrweg 4, 2333 CA, Leiden, The Netherlands  Corresponding author

DOI: [10.21105/joss.05478](https://doi.org/10.21105/joss.05478)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Daniel S. Katz](#)  

Reviewers:

- [@MikeHughesKent](#)
- [@brandondube](#)

Submitted: 16 May 2023

Published: 12 August 2023

License

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

Summary

PyPO is a Python interface for end-to-end design, simulation and analysis of (quasi-)optical reflector systems. It can model the forward and backward propagation of electromagnetic field distributions between multiple planar and (off-axis) quadric surfaces, as well as far-field propagation. Simulations are performed using either geometrical optics (GO) or the equivalent surface current approach, belonging to the field of physical optics (PO) ([Balanis, 1989](#)). The GO and PO calculations are performed using libraries written in C++ and CUDA, allowing for multi-threading and GPU acceleration. Common figures of merit, such as aperture efficiency and half-power beamwidth, can be calculated and used for quantitative analysis of the designed system. Input beam patterns can be selected from a range of models, such as Gaussian beams, point sources and uniform current distributions. Custom beam patterns can also be imported to, for example, model the propagation of measured beam patterns through simulated optical systems.

PyPO can be used through either a scripting-based approach, where simulations are defined in Python scripts, or through the graphical user interface (GUI). It only carries core dependencies on NumPy ([Harris & others, 2020](#)), Matplotlib ([Hunter, 2007](#)) and SciPy ([Virtanen & others, 2020](#)). The unittesting framework carries a dependency on nose2. The GUI carries dependencies on PySide6, pyqtdarktheme and attrs.

Statement of need

Development of PyPO started with the need for alignment strategies for the wideband sub-mm spectrometer DESHIMA 2.0 ([Taniguchi & others, 2022](#)). A software package capable of efficient GO and PO calculations through optical systems consisting of off-axis quadric surfaces was necessary for calculating the configuration of the corrective optics. Currently, PyPO is also being used in simulations of measured beam patterns of DESHIMA 2.0 at the ASTE ([Ezawa & others, 2004](#)) telescope for the analysis of instrument and optical system performance.

Commercial software for GO/PO calculations and analysis, such as OpticStudio (Zemax) and GRASP (TICRA), has already been developed. Open-source optical simulation software packages such as POPPy (Perrin & others, 2012) and Prysm (Dube, 2019) are also available. These open-source software packages use Fourier methods to solve the Rayleigh-Sommerfeld integral equation. PyPO aims to contribute to this open-source optical simulation ecosystem by offering a software package working on the principle of the equivalent surface currents.

Availability

PyPO can be found on [Github](#) and is available for Linux, MacOS and Windows. Software documentation and instructions regarding installation, contributing and issue tracking can be found in the [documentation](#). The package comes with several Jupyter Notebook tutorials illustrating the workflow and features, and can be used as building blocks for new reflector systems. Several tutorials for using the GUI features are also included and can be found in the software documentation.

Acknowledgements

We thank Shahab Oddin Dabironezare for useful discussions and proofreading the manuscript. This work is supported by the European Union (ERC Consolidator Grant No. 101043486 TIFUUN). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

References

- Balanis, C. A. (1989). *Advanced engineering electromagnetics*. John Wiley & Sons. ISBN: 0-471-62194-3
- Ezawa, H., & others. (2004). The Atacama Submillimeter Telescope Experiment (ASTE). In J. M. O. Jr. (Ed.), *Ground-based telescopes* (Vol. 5489, pp. 763–772). International Society for Optics; Photonics; SPIE. <https://doi.org/10.1117/12.551391>
- Harris, C. R., & others. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362. <https://doi.org/10.1038/s41586-020-2649-2>
- Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90–95. <https://doi.org/10.1109/MCSE.2007.55>
- Taniguchi, A., & others. (2022). DESHIMA 2.0: Development of an integrated superconducting spectrometer for science-grade astronomical observations. *Journal of Low Temperature Physics*, 209, 278–286. <https://doi.org/10.1007/s10909-022-02888-5>
- Virtanen, P., & others. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17, 261–272. <https://doi.org/10.1038/s41592-019-0686-2>