The Journal of Open Source Software

simshady: A Javascript Package for Photovoltaic Yield Estimation Based on 3D Meshes

Florian Kotthoff [1,2¶], **Konrad Heidler** [1], **Martin Großhauser** [1], and **Korbinian Pöppel** [1]

1 OpenPV GbR, ADRESS 2 OFFIS Institute for Information Technology, Escherweg 2, 26121 Oldenburg, Germany ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

`@openpv/simshady` is a JavaScript package for simulating photovoltaic (PV) energy yields. It can integrate local climate data and 3D object shading. It utilizes three.js meshes for geometric modeling and performs shading analysis using a WebGL-parallelized implementation of the Möller-Trumbore intersection algorithm. The output includes color-coded three.js meshes representing the expected PV yield.

Statement of need

To meet global climate targets, solar photovoltaic (PV) capacity must expand significantly. Tripling renewable energy capacity by 2030 is essential to limit global warming to 1.5°C (International Energy Agency, 2023). The expansion of PV plays a crucial part, and PV systems offer another benefit. Small scale house mounted PV systems enable the public participation and legitimize the energy transition. For calculating the yield of PV systems, various factors are important: the location of the planned installation, the local climate, surrounding objects such as houses or trees, and the terrain. To provide accurate estimates of expected yields, simulation tools are essential in both research and practical PV system planning. For these reasons, a variety of software tools already exist (**?**). The python package pvlib (Anderson et al., 2023)

Package description

The javascript package `simshady` first initializes a scene object. Various adder functions can then be used to add relevant data to this scene:

- The core simulation geometry as the object where the PV yield is of interest. This is usually a geometry of a house. This geomety needs to be a vector mesh. It is not optional.
- The background geometries as geometries that are relevant for shading, but where no PV yield should be calculated. These geometries are usually the houses from the neighborhood or trees. These geometries need to be vector meshes. They are optional.
- A height map can be added to account for the impact of shading from mountains or hills. This heightmap needs to be rasterized.
- Sun irradiance data is needed in the format of sky domes, which are essentially a list of [altitude, azimuth, irradiance] values.

If all relevant data is added to the scene object, the simulation can be started. Since the physical simulation of solar rays can be parallelized, this simulation is implemented in WebGL.

By that, the GPU can be utilized, which results in a much stronger performance.

Conclusion

CRediT Authorship Statement

FK:

Acknowledgements

The authors acknowledge support by ... (how do we name Prototypefund here?)

References

Anderson, K. S., Hansen, C. W., Holmgren, W. F., Jensen, A. R., Mikofski, M. A., & Driesse, A. (2023). Pvlib python: 2023 project update. *Journal of Open Source Software*, *8*(92), 5994.

International Energy Agency. (2023). *Tripling renewable power capacity by 2030 is vital to keep the 1.5°c goal within reach*. https://www.iea.org/commentaries/tripling-renewable-power-capacity-by-2030-is-vital-to-keep-the-150c-goal-within-reach.