

# Easy3D: a lightweight, easy-to-use, and efficient C++ library for processing and rendering 3D data

Liangliang Nan<sup>1</sup>

<sup>1</sup> 3D Geoinformation Research Group, Delft University of Technology, the Netherlands

DOI: [10.21105/joss.03255](https://doi.org/10.21105/joss.03255)

## Software

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

Editor: [Kevin M. Moerman](#) ↗

## Reviewers:

- [@adi3](#)
- [@vissarion](#)
- [@bhajay](#)

Submitted: 02 December 2020

Published: 23 August 2021

## License

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

## Summary

*Easy3D* is an open-source library for 3D modeling, geometry processing, and rendering. It is implemented in C++ and designed with an emphasis on simplicity (i.e., processing and visualizing 3D data can be achieved by few lines of API calls). The contributions of *Easy3D* are threefold: (1) efficient data structures for representing common 3D data (i.e., point clouds, surface meshes, polyhedral meshes, and graphs), (2) commonly used algorithms for processing 3D data, and (3) standard techniques for the visualization of 3D data. The data structures, geometry processing algorithms, and rendering techniques can be seamlessly and easily integrated for the fast development of research prototypes or 3D applications. [Figure 1](#) and [Figure 2](#) show overviews of the data structures and rendering functionalities respectively provided by *Easy3D*.



**Figure 1:** The data structures provided by Easy3D. From left to right: point cloud, surface mesh, polyhedral mesh, and graph.

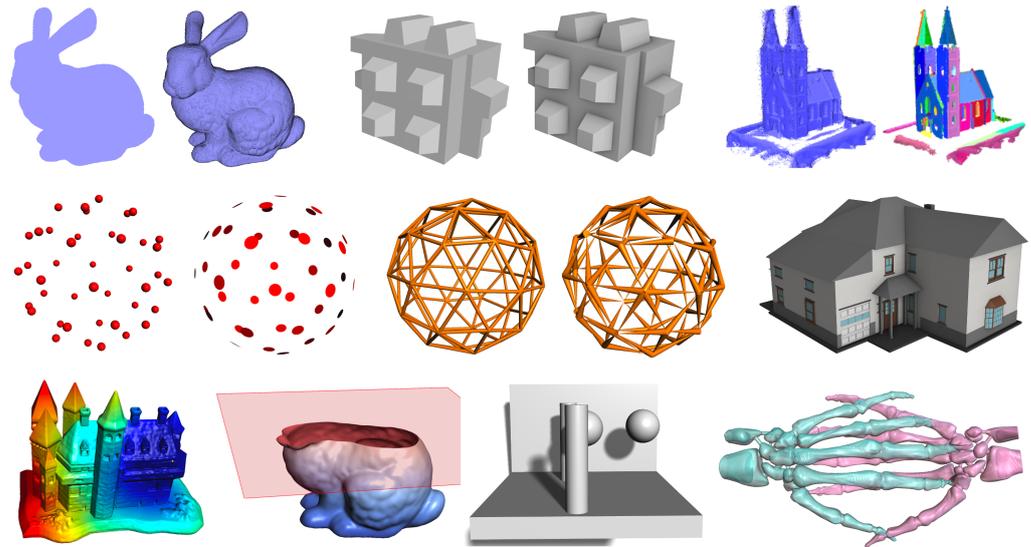
## Statement of need

The amount of 3D data has been explosively increasing in the past decades, driven by the fast development of data acquisition techniques (e.g., laser scanning, photogrammetry, and 3D computer vision) and real-world applications (e.g., urban planning, autonomous driving, surveying, and entertainment). The processing and application of such data require specialized algorithms and software systems, for which great tools and algorithm libraries have been developed, such as MeshLab ([Cignoni et al., 2008](#)), CloudCompare ([Girardeau-Montaut, 2016](#)), CGAL ([Fabri & Pion, 2009](#)), libigl ([Jacobson et al., 2016](#)), and PMP ([Sieger & Botsch, 2020](#)).

The development of end-to-end 3D data processing software typically requires integrating different functionalities such as data structures (for representing and storing 3D data), algorithms (for processing and analysis), and rendering (for visualizing the data and the derived information). However, each of these open-source software has a particular focus and it is not an easy task to integrate the functionalities provided by different software packages. For example, MeshLab ([Cignoni et al., 2008](#)) and CloudCompare ([Girardeau-Montaut, 2016](#)) focus on easy-to-use GUIs (Graphical User Interface) for processing meshes and point clouds,

respectively. The CGAL (Fabri & Pion, 2009), libigl (Jacobson et al., 2016), and PMP (Sieger & Botsch, 2020) have an emphasis on the implementation of sophisticated mesh processing algorithms, lacking rendering and interactive capabilities. As a result, the development process has a high demand for the expertise of a developer in all the related aspects.

To facilitate the fast development of 3D data processing software, *Easy3D* provides not only the data structures, but also commonly used geometry processing algorithms, and standard rendering techniques. All these functionalities can be seamlessly integrated into a software implementation.



**Figure 2:** An overview of the rendering functionalities of Easy3D.

## Overview of the features

The core features of *Easy3D* include:

- Efficient data structures for representing and managing 3D models (i.e., point clouds, surface meshes, polyhedral meshes, and graphs). Easy to add/access arbitrary types of per-element properties. Non-manifoldness is automatically resolved when loading models from files.
- A set of widely used algorithms, e.g., point cloud normal estimation/re-orientation, Poisson Surface Reconstruction, RANSAC, mesh simplification, subdivision, smoothing, parameterization, and remeshing.
- A bunch of rendering techniques, e.g., point/line imposters, ambient occlusion (SSAO), hard shadow (shadow maps), soft shadow (PCSS), eye-dome lighting (for rendering point clouds without normal information), and transparency (average color blending, dual depth peeling).
- High-level encapsulation of OpenGL and GLSL for convenient and efficient rendering (based on modern and faster programmable-shader-style rendering, i.e., no fixed function calls). Client code does not need to touch the low-level APIs of OpenGL.
- Step-by-step tutorials demonstrating various uses of the APIs, to get acquainted with the data structures, rendering techniques, and algorithms for 3D modeling and geometry processing.

- A viewer that can be used directly to visualize 3D scenes in various formats, which can also be easily extended. A handy tool *Mapple* created out of the *Easy3D* library for rendering and processing 3D data.

*Easy3D* was initially designed with an intention for research and educational purposes, while its wide range of features also enables developing sophisticated 3D applications. *Easy3D* has been widely used by graduate students in courses and research projects and has enabled several scientific publications, e.g., [Du et al. \(2019\)](#), [Fan et al. \(2020\)](#), and [Gao et al. \(2021\)](#).

## Acknowledgements

The implementation of *Easy3D* greatly benefited from and was inspired by existing great open-source libraries, such as CGAL ([Fabri & Pion, 2009](#)), libigl ([Jacobson et al., 2016](#)), and PMP ([Sieger & Botsch, 2020](#))]. We would like to thank the original authors of these projects for their permissive license terms. We also thank the users, contributors, and JOSS reviewers for reporting/fixing bugs, testing, and providing valuable feedback and suggestions.

## References

- Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., & Ranzuglia, G. (2008). Meshlab: An open-source mesh processing tool. *Eurographics Italian Chapter Conference, 2008*, 129–136. <https://doi.org/10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136>
- Du, S., Lindenbergh, R., Ledoux, H., Stoter, J., & Nan, L. (2019). AdTree: Accurate, detailed, and automatic modelling of laser-scanned trees. *Remote Sensing*, 11(18), 2074. <https://doi.org/10.3390/rs11182074>
- Fabri, A., & Pion, S. (2009). CGAL: The computational geometry algorithms library. *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 538–539. <https://doi.org/10.1145/1653771.1653865>
- Fan, G., Nan, L., Dong, Y., Su, X., & Chen, F. (2020). AdQSM: A new method for estimating above-ground biomass from TLS point clouds. *Remote Sensing*, 12(18), 3089. <https://doi.org/10.3390/rs12183089>
- Gao, W., Nan, L., Boom, B., & Ledoux, H. (2021). SUM: A benchmark dataset of semantic urban meshes. *ISPRS Journal of Photogrammetry and Remote Sensing*, 179, 108–120. <https://doi.org/10.1016/j.isprsjprs.2021.07.008>
- Girardeau-Montaut, D. (2016). CloudCompare. In *GitHub repository*. GPL Software. <https://github.com/CloudCompare/CloudCompare>
- Jacobson, A., Panozzo, D., Schüller, C., Diamanti, O., Zhou, Q., Pietroni, N., & others. (2016). Libigl: A simple c++ geometry processing library. In *GitHub repository*. GitHub. <https://github.com/libigl/libigl>
- Sieger, D., & Botsch, M. (2020). The polygon mesh processing library. In *GitHub repository*. GitHub. <https://github.com/pmp-library/pmp-library/>