Clipping polygons from GNU Octave[*]

José Luis García Pallero[†]

May 26, 2011

Abstract

This is a small introduction to using the `OctCLIP` package. In this text, you can overview the basic usage of the functions in GNU Octave[1]. If you need a detailed description about the Greiner-Hormann implemented algorithm, please read [2] and visit http://davis.wpi.edu/~matt/courses/clipping/.

1 Overview

The `OctCLIP` package allows you to perform boolean operations (intersection, union and difference) between two polygons in GNU Octave using the Greiner-Hormann algorithm[2].

Greiner-Hormann is an efficient algorithm for clipping arbitrary 2D polygons. The algorithm can handle arbitrary closed polygons, specically where the subject and clip polygons may self-intersect.

2 Installation

As several GNU Octave packages, `OctCLIP` installation consists in compiling the C++ kernel sources, link them against GNU Octave library to generate `*.oct` functions and copy this `*.oct` executables and other `*.m` functions into a working directory.

The automatic procedure can be easily done by running the command:

```
octave:1> pkg install octclip-x.x.x.tar.gz
```

where `x.x.x` is the version number.

After that, the functions and documentation are installed in your machine and you are ready for use the package.

[†]Instituto de Astronomía y Geodesia, Fac. de CC Matemáticas, Universidad Complutense de Madrid. jlgpallero@pdi.ucm.es, jgpallero@gmail.com

[1]http://www.octave.org

[2][2] and http://davis.wpi.edu/~matt/courses/clipping/

3 GNU Octave functions

Two types of functions are programmed for GNU Octave: one `*.oct` function and one `*.m` function.

3.1 `*.oct` function

This function are linked with the C code that actually make the computations. You can use it, but is no recommended because the input arguments are more strict than `*.m` functions and don't check for some errors.

The function is:

- `_oc_polybool`: boolean operation between two polygons.

3.2 `*.m` function

This function makes the computations by calling the `*.oct` function. You must call this function because you can use different number of input arguments and checking of input arguments is performed.

The function is the same as in section 3.1 (without the _ at the beginning of the name):

- `oc_polybool`: calls `_oc_polybool`.

A test script for the package exists too:

- `test_octclip`: plots an example of the `oc_polybool` usage.

3.3 Error handling

`*.oct` and `*.m` functions can emit errors, some due to errors in input arguments and other due to errors in functions from the C[3] code.

Errors due to wrong input arguments (data types, dimensions, etc.) can be only given for `*.m` function and this is the reason because the use of this function is recommended. In this case, the execution is aborted and nothing is stored in output arguments.

The `*.oct` function can emit errors due to wrong number of input arguments, wrong value of the operation identifier and internal errors of memory allocation.

4 Caveats

To do.

5 Examples

To do.

[3]The algorithm is internally implemented in C (C99 standard).

References

[1] Eaton, John W.; Bateman, David, and Hauberg, Søren; *GNU Octave. A high-level interactive language for numerical computations*; Edition 3 for Octave version 3.2.3; July 2007; Permanently updated at http://www.gnu.org/software/octave/docs.html.

[2] Greiner, Günter, and Hormann, Kai; *Efficient clipping of arbitrary polygons*; ACM Transactions on Graphics; Volume 17(2), April 1998; Pages 71–83. There is a web link with some example code at http://davis.wpi.edu/~matt/courses/clipping/.

[3] Kim, Dae Hyun, and Kim, Myoung-Jun; *An Extension of Polygon Clipping To Resolve Degenerate Cases*; Computer-Aided Design & Applications; Vol. 3; Numbers 1–4, 2006; Pages 447–456.