CliReader User-guide

CliReader is a command line utility program to read files of the format common layer interface (CLI). The common layer interface format is used to store slicing and hatching information that is needed by additive manufacturing (AM) or 3D printing machines. CliReader.exe reads both text format and binary format CLI files and displays the total surface area in square mm of each layer to the computer screen. Optionally, it can write an HTML file with an SVG representation of polylines of a given layer.

The following topics are discussed:

- Execution procedure
- Supported CLI keywords
- Limitations
- Example of the execution procedure
- Programming notes
- Future enhancements
- Reference

Execution procedure

You can invoke CliReader from the command line. For example

>CliReader.exe -f <fileName.cli>

Summary of command-line options

Option	Option's values	Remarks
-f	CLI filename	Required option.
-v	A value between 1 and 3	Optional. Verbose output of each CLI commands on the screen. Not recommended to large files. For large files, the user should redirect the standard output to a file.
-html	A layer id between 0 and one less than the number of layers	Optional. Writes an HTML file containing an SVG image of the layer polylines and the total layer area.
-h		Optional. Displays help.

Supported CLI keywords

CliReader supports all valid CLI keywords except $$USERDATA [1].

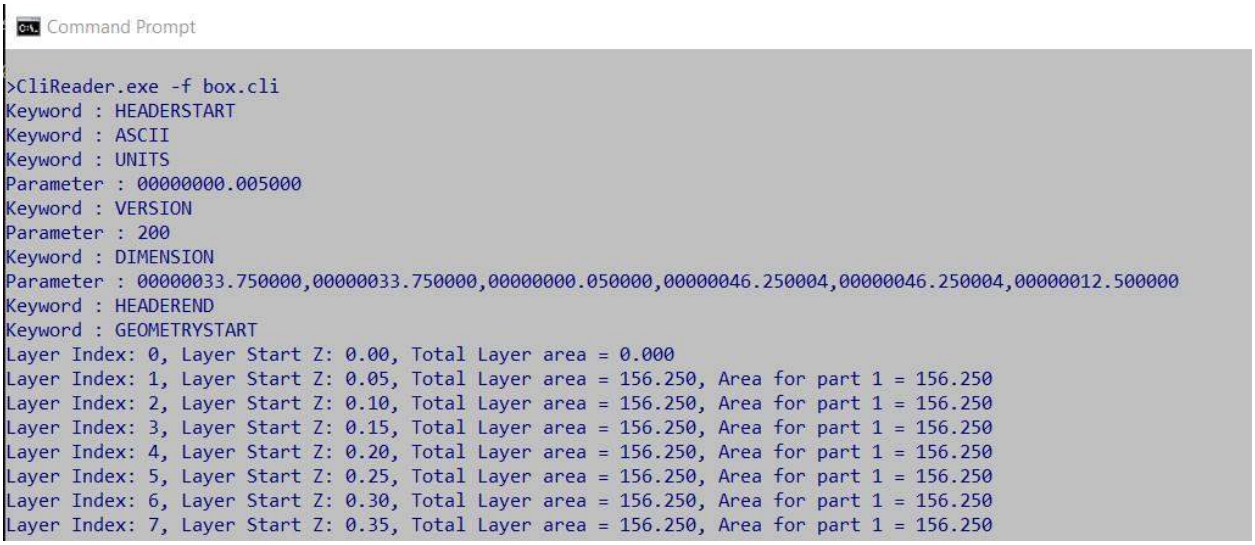
Limitations

Although all CLI keywords are supported, keywords ALIGN, Start Layer Short, Start PolyLine Short, and Start Hatches Short were not tested because of the unavailability of input data for testing.

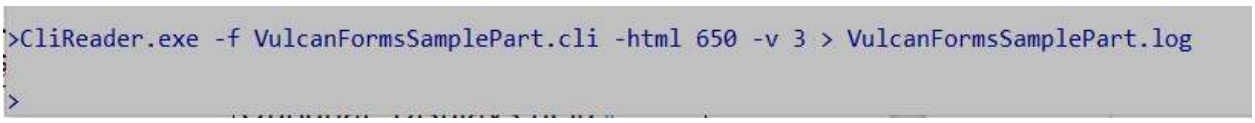
CliReader assumes that the provided input file is a valid CLI file conforming to the CLI standard. CliReader does a minimal validation of the input file. It does not check if the input file contains any invalid keywords. It also does not check if there are conflicting keywords; for example an input file containing both $$BINARY and $$ASCII keywords will cause the program to fail.

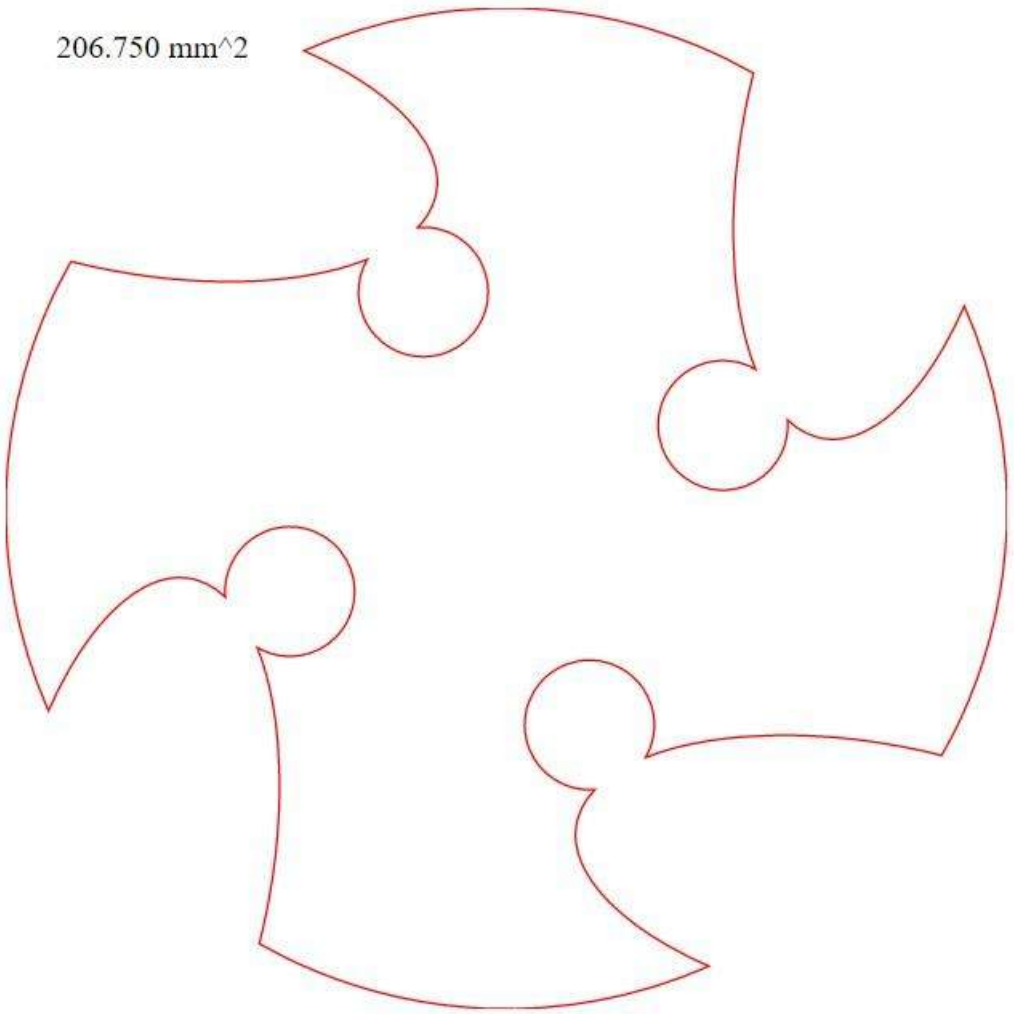
Example of the execution procedure

The figure below shows a screenshot of the essential execution of the CliReader.



The figures below show a screenshot of the execution of CliReader requesting an HTML output and a screenshot of the HTML output file.





Programming Notes

The CliReader was developed using ISO C++17 Standard. One can build the CliReader on Windows from source codes using Visual Studio by setting the option /std:c++17. One can also build CliReader on other OS like Linux or Mac by using any standard C++ compiler.

The organization of the source code follows a Strategy Pattern [2]. The file CliReader.cpp contains the 'main' function. The CliParser class sets up the input file for reading using either a text format interpreter or a binary format interpreter, depending on the format of the data being read. The classes CliASCIIKywInterpreter and CliBinKwyInterpreter implement the ASCII and the binary file interpreters, respectively. The CliData holds header information, and polylines and hatches information for the current layer being read. It also computes the layer area. Utility classes such as CmdLine

and CliHtmlFileWriter are singleton classes that parse command line arguments and write html files, respectively.
Codes are written in a self-documenting way that requires minimal comments in the code for a reader to understand it. This is achieved by choosing appropriate names for classes and variables and following a natural execution flow. This helps code maintenance and future enhancements.

Reading of CLI files for both binary and text format was done in binary mode. The keyword and parameter syntax are simple enough to parse them easily. A more robust solution will be to use a regular expressions library like std::regex from STL.

Future enhancements

There are a number of possible future enhancements:

1. The only computation currently done on the layer data is to compute the layer area, which is not computationally intensive. For computationally intensive tasks, a thread task manager can be set up to delegate computational tasks for each layer to multiple threads on a multi-threaded modern CPU. The task for given layers can be executed asynchronously when other layers' information is being read.
2. Validity checking for invalid and conflicting CLI keywords in the input file.
3. Exception handling can be improved.

References

1. COMMON LAYER INTERFACE (CLI). http://web.archive.org/web/19970617041930/http://www.cranfield.ac.uk/aero/rapid/CLI/cli_v20.html
2. Strategy in C++. https://refactoring.guru/design-patterns/strategy/cpp/example