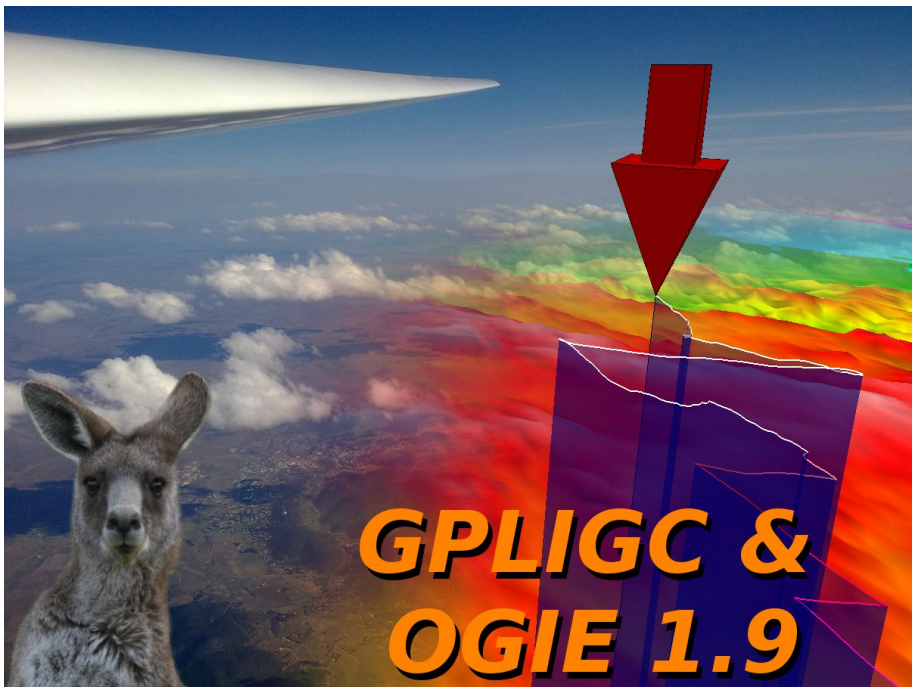
GPLIGC & OGIE
Version 1.9
Manual

Hannes Krüger

December 16, 2010



47.0789°N 11.3060°E

Contents

1 Introduction

GPLIGC is a software package for glider pilots, hang- and paraglider pilots, and for all others, who want to analyse and visualise GPS track logs. *GPLIGC* reads track logs from files in igc-format as specified by the International Gliding Commission [1]. Extracting the data from the GPS devices and conversion to the igc format has to be done with third-party software. (GPS tracks can be downloaded from some Garmin devices using *gpspoint* [2], Nokia/Symbian mobile phones can be used as loggers utilising *GSIL* [3] and another option is to use *gpsbabel* [4]).

The package contains two main programs: (1) *GPLIGC*, analysation and (2) *OGIE*, 3D visualisation (can also be used as a digital elevation data viewer).

The software can be used under the terms of the GNU General Public License (see appendix G), which means that it's free and the source code is available. For details read the license, which is included in appendix G.

The webpage of *GPLIGC* can be found at [5].

1.1 GPLIGC

GPLIGC is a flight data analysing software. Its name is assembled from **GPL** (the GNU General Public License, [6]), **G**nuplot (free plotting software, [7]), **P**erl (the famous scripting and programming language, [8]), **L**ogger (flight data recorder) and **IGC** (the International Gliding Commission and name of the flight data file format, [1]). *GPLIGC* is written in Perl [8], using the Perl/Tk module [9] for the graphical user interface. Track and altitude plots can be visualised in a simple way and some basic statistical information can be calculated. The recorded data can be analysed in detail. Optimisation for the onlinecontest can be performed. Turn-point observation zones can be displayed. *Gnuplot* [7] is used to give out some plots (barogram, GPS-altitude, vertical speed, speed, noise level, etc.) of the data either to the screen or some graphical file format (including png, fig, ps, eps). *GPLIGC* is able to locate coordinates of photos, which have been taken with a digital camera, while logging GPS data. To use this geo-tagging feature a correct timestamp in the JPEGs EXIF header is needed or it should be retained as the files timestamp.

The development of *GPLIGC* started in January 2000.

1.2 OGIE

OGIE is a program written in C++ using openGL and GLUT (or freeglut [10]) libraries. The flight data can be visualised in 3D (even in *real 3D*, using stereoscopic methods). The viewpoint can be controlled in several ways (egocentric, swivel/rotate or coupled with flight). Digital elevation models can be used to display the terrain, digitised maps can be used, and airspaces from OpenAir™-files can be displayed also. Colour scaling can be applied to the terrain data, the digitised maps and to the flight-track itself. The *OGIE* can also be used as a digital elevation model viewer. *OGIE* is able to render offscreen. Images can be generated hardware accelerated, or hardware independent (with Mesa [11]). This can be used to generate nice pictures for contests etc. (server use). *OGIE*s name was assembled from ***o**pen**GL**I**GC**explorer*: **openGL** (the open Graphics Library), **IGC** [1], **explorer**.

The development of *OGIE* started in April 2002, after I bought my first openGL graphics card. Until 2010 the long name *openGLIGCexplorer* was used.

How they work together Basically *GPLIGC* and *OGIE* are independent pieces of software. The *OGIE* was designed to be an independent visualisation-only tool, because Perl is too slow for that task. However, if you start *ogie* from within *GPLIGC* some data (altitude calibration data, marked lifts, etc.) will be put forward to *ogie*.

1.3 Contact, Bug reports, feature requests

Bug reports and feature requests should be submitted via the GPLIGC support page at Sourceforge [5]. I recommend to sign up for the *gpligc-announce* mailing list [5], which I use to inform users of updates or serious bugs, etc. (very low traffic)

2 Requirements

GPLIGC

- Perl 5 with Perl/Tk module [8, 9]
- *gnuplot* $\geq$ 3.7 (earlier versions are not tested). Some basic functionality will work without *gnuplot*
- If no binaries are available for your platform, you need a C/C++ compiler to compile the *optimizer*, which is needed for the task-optimisation feautures.

OGIE

- OpenGL 3D hardware is recommended, but OpenGL software rendering (e.g. Mesa3d) will work too (but slow).
- If binaries are not available for your platform you need a C/C++ compiler in order to build *ogie*.

Platforms, software versions

GPLIGC is used/compiled/developed and/or tested on the following platforms, compilers, hardware:

- Linux: x86_64*, x86
- Mac OS X: 10.4/10.5 (ppc/intel)
- Windows: Vista* (x86), XP (x86)
- OpenBSD: 4.5 (amd86)

- gcc: 4.4*, 4.3, 4.1, 3.4, 3.3, 2.95
- Perl: 5.12*, ActiveState Perl 5.10*
- Perl Tk: 804.029*
- Gnuplot: 4.4*, 4.2, 4.1, 4.0, 3.7
- Mesa: 7.8.2*, 7.7.1, 7.4.1, 7.0.3, 6.5.2, 6.4.2, 5.0.2 for osmesa offline rendering
- gpsd: 2.95*, 2.39 (up to GPLIGC 1.8)

*Systems and software versions used by the author (for development and testing)
If you run *GPLIGC & OGIE* on platforms not listed here please let me know.

3 Installation

3.1 General Linux and Unix installation procedure

1. Extract the archive:
   `tar xvzf GPLIGC-version-src.tar.gz`
   `tar xvzf GPLIGC-version-os-arch.tar.gz`, in the case you're using a package containing binaries.
   Change to the just created GPLIGC-directory:
   `cd GPLIGC-version`
2. Become root if you need root previleges to write in installation directories (recommended).
3. start the installation script: (If you have security concerns, feel free to review the srcipt before)
   `./install.sh`
   If the script fails to run, you might use an incompatible shell. See section E.
   If no binaries are found (or if the binaries don't work for some reason) the script will show an alert and will offer you the automatic compilation of the binaries (and if this fails you will find more details in section 3.1.1).
   The script will ask you for
   a) an installation prefix (/usr/local is recommended)
   If the script is allowed to write, it will install *GPLIGC* and *OGIE*
   Files from older versions will be overwritten.
4. Make sure that everybody who wants to use *GPLIGC* is allowed to write in /tmp (default on most Linux systems)
5. Every user, who wants to use *OGIE* should copy the example configuration file `.ogierc` (from /usr/(local/)share/GPLIGC/) to his HOME directory and edit it for special settings. by default *OGIE* will always read the configuration from the users home directory.
6. Make sure that Gnuplot [7] is installed and in path. *GPLIGC* will work without Gnuplot too, but you will not be able to use some plotting functions.
7. Make sure that the Perl/Tk [9] module is installed
8. Read the documentation to learn how to use GPLIGC & OGIE

3.1.1 Compiling OGIE

These details are not needed by the normal user. If the `install.sh` script fails to produce usable binaries, or you need the special offscreen binary, you'll find some help here.

The C/C++ part of the software is located in the `openGLIGCexplorer` subfolder.

Building from source `make help` will give you an overview of the available targets in the makefile. In general you should try to use one of the *all-** targets, closest matching your platform. If that fails, you need to modify the `Makefile` or to contact the author.

Compiling a statically linked (with Mesa) binary This is only needed if you're going to use ogie on a server for offcreen rendering (like the onlineplotter).

Static linking with Mesa is possible and tested for linux and windows (cygwin) and requires at least *Mesa 5.0.1*, (4.x seems not to work with *OGIE* and offscreen rendering). However, be warned it is a pain to get it linked.

Mesa 7.0.3 works fine, as recently (may 2008) tested. If you need larger than 4096x4096 pictures (e.g. for cool posters) you can change to `#define MAX_WIDTH 32768` and `#define MAX_HEIGHT 32768` in `src/mesa/main/config.h` of the mesa source code, prior to compile mesa.

Mesa 7.4.1: tested in may 2009. It works for me as follows: Download and unpack MesaLib and MesaGLUT-7.4.1. Change the maximal width and height as described above. Use `make linux-x86-64-static` or a similar target. After Mesa compiled, `cp lib64/lib*.a ~/lib/` to put the static mesa-libs in your private ~/lib/.

For further details see the Makefile. `MESAPATH, GLEX_OSMESA` and `GLEX_OSMESA_WIN` are interesting points. The osmesa target in the Makefile requires the static Mesa libraries in ~/lib/

```
make clean
make osmesa
```

Compiling this on windows/cygwin is even more painful. However, I succeeded once, see comments in the Makefile.

```
make clean
make osmesa-win
```

The binary, which will be build is called *openGLIGCexplorer-mesa* and should be installed manually.

3.2 OpenBSD

On OpenBSD platforms, you need to install the following packages: gmake, p5-Tk, gnuplot, jpeg, bash. You'll need the glut library too, which is not available as a packge, but from the ports.

The install.sh script should started with the bash shell:
`sudo bash ./install.sh`

3.3 Gentoo Linux

Gentoo users may use the gpligc ebuild from the *sunrise* overlay.
Adding the sunrise overlay: `layman -a sunrise`
Installing GPLIGC: `emerge gpligc`

3.4 Mac OS X

This paragraph is mainly written by Matthew Hoover. Also he was the one, who did the testing while GPLIGC/ogie was ported to Mac OS X. Owing to his efforts, binaries for ppc- and intel-based Mac OSX machines can be provided. Thanks a lot, Matthew!

Another section was added by Michael Schlotter, who describes how to install GPLIGC/ogie without using fink. For that reason Perl/Tk and Gnuplot have to been build from sources. Thanks Michael

A few notes: Mac OS X is a unix operating system, with a Mac-GUI on top. To use GPLIGC & OGIE we have to do a step behind the scenes and we have to deal with the unixish features. In case of any troubles you can ask for assistance.

3.4.1 General

In order to get GPLIGC running, you need Perl with the Perl/Tk module. This usually requires to have X11 installed, since Perl/Tk doesnt work with the native Mac OSX GUI (Aqua). As you also have to install gnuplot, I recommend the easy way shown in section 3.4.2 using the fink platform. If you're a more experienced developer (not using fink), you may want to compile Perl/Tk and gnuplot by yourself. See section 3.4.3. To compile the binaries of ogie on your machine, you'll need some developer tools (as gcc, make, etc.) So far, I didnt find a way not to use the jpeglib from fink (if you know, please report).

3.4.2 Matthew's howto, using fink

1. Install X11 from the Tools disk included with the OSX package.
   (X11 is the unix X-Window system and is especially needed for the Perl/Tk-stuff (for GPLIGC)).

2. Install fink [12].
   Fink is a packaging system that allows Mac users to install, compile and use a wide range of free software. We need this to install gnuplot and the Perl/Tk-module.

   (a) If the user is unfamiliar with fink then also install fink commander which is a GUI front end for fink.

   (b) Make sure that the fink or fink commander is able to install "unstable" packages. You have to modify /sw/etc/fink.conf (Add main/unstable to the line containing "Trees:") More can be found in the fink documentation.

3. Install and compile the Perl/Tk module. Therefore you have to check which Perl is installed on your system (perl -V). There are different Perl/Tk-module versions (and I am not sure if the newer ones will work with older perls...) According to your installed perl choose one of tk-pm560, tk-pm580 or tk-pm581. This example will continue with tk-pm581.

   install the tk-pm581 package using fink (`fink install tk-pm581`) or fink commander. There are also tk-pm581-bin and tk-pm581-man. These will be installed and compiled and archived when the first is selected. If the GPLIGC will not run try unpacking these also.

4. Install gnuplot using fink (`fink -b install gnuplot`) or fink commander. I used the last stable version with the binary files because it is faster although there is a newer unstable version.

5. Now you can proceed with the instructions for Unix (see 3.1) installation. Be aware to use a binary package for Mac OS X. The installation paths will be different from those used in Unix/Linux instructions. For the installation prefix you can choose by yourself (maybe /sw or /usr).

6. To run, open X11 and then Terminal. GPLIGC must be started from within Terminal and X11 must also be running because it will not be automagically started.

7. You should add
   `test -r /sw/bin/init.sh && . /sw/bin/init.sh`
   to your `.bashrc` file to start GPLIGC in a xterm shell without a terminal shell opened.

3.4.3 Michael Schlotter's howto

1. Install Developer Tools from the Mac OSX Installation CD/DVD
2. Download and install Apples X11-Server
3. Install X11SDK and BSDSDK. This is done by double-clicking on `X11SDK.pkg` and `BSDSDK.pkg` in */Applications/Installers/Developer Tools/Packages*
4. Download, build and install gnuplot 4.0.0. Don't worry if *make-check* produces some errors.
5. Download, build and install Perl/TK. I used `Tk-804.027.tar.gz`. Don't worry if *make-test* produces some errors.
6. Download and install the latest GPLIGC with precompiled binaries for Mac

3.5 Windows NT/2000/2003/XP/Vista/2008/Win7

GPLIGC/OGIE was not yet tested on WinNT, Windows 2000, Windows Home Server, Windows 2003, Windows 2008 and Windows7. If you run it on one of these platforms, please email me a short report. Thanks!

1. Make sure that ActivePerl [13] is installed
   This is needed for *GPLIGC* and the installation script (*OGIE* will not need perl).
2. If your Active State Perl version is later or equal 5.10, you have to open the Perl Package Manager and to install the Tk (804.028) module. Tk is not shown in the default list, you have chose 'all packages' from the view menu.
3. Unzip the GPLIGC-version-win32.zip archive to a temporary location (maybe you have done that already). Open this location in the Explorer and double-click the install-script:
   `install_windows.pl`
   (if *.pl scripts are not associated with the perl-interpreter already, you can try "open with", select browse and find bin/perl.exe in the perl-install directory).
   Attention! Don't run the install script from within the zip file. That method will not work! Unpack the zip archive in any case and run the script from the unpacked directory.
4. The Installation script will ask you for a location to install. Let the script do the following work for you: a) copy all files to the install-location b) set some environment variables by adding them to the registry.
5. If the installation script fails, and tells you to set the environment Variables by yourself:
   Make sure that an environment variable GPLIGCHOME is set, which contains the full absolute path to the GPLIGC-directory:
   For example:
   c:\some\path\GPLIGC
   And add the GPLIGC-directory to your PATH
   How to set an environment variable:
   a) Windows NT, 2000, XP, Vista:
   Start - Settings - Control Panel - System (Advanced) - Environment...

6. You can remove the temporary directory, where the zipfile was extracted.

7. If you do not have Gnuplot installed already, do so. Get it from the gnuplot page [7] (get the win32 zip, e.g. `gp425win32.zip`) Installation is simple: just put the file `bin/wgnuplot.exe` in a location which is in the path (c:\windows\system32\ or the GPLIGC install location for example)
ATTENTION! The executable file of the older gnuplot *3.7.x* was named `wgnupl32.exe`. If you use the older gnuplot, you have to change the setting for the configuration key `gnuplot_win_exec` to `wgnupl32.exe` in the `GPLIGC.ini` file (for details see A). But you should consider updating to gnuplot 4, since there are new nice interactive features like zooming, rotating of 3d-plots etc.

8. Edit the configuration file `opengligcexplorer.ini` if you like to use a "digital elevation model", "digitized maps" and "airspace files". This configuration file can be named as `opengligcexplorerrc` also.

   For details read sections 3.8, 3.9 and 3.10.

9. Make a nice shortcut to `GPLIGC.pl` on your desktop if you like

10. Read the manual

Notes for Windows 95, Windows ME Support for these platforms is discontinued.

3.6 Update installation

If you want to update from an earlier version, you can safely use the install-scripts. Executable files will be replaced. Your data and configuration file will not be overwritten. But be sure to use the same paths for installation and symbolic links as in the previous installation, otherwise you my have conflicting installations.

3.7 Additional Perl modules

3.7.1 Image::ExifTool

To use the photo locator function, you should have the perl-module Image::ExifTool [14] installed.

Attention: *GPLIGC 1.6 and earlier used Image::EXIF, which is a different module. Unfortunately that was not exif V2 compliant*

The most recent version of Image::ExifTool (at the time writing this) is 7.51 There are three options how to install this module.

do it yourself You should go to the CPAN [15] and search for Image::ExifTool, download and install it. After downloading the archive, it takes the usual three commands:
`perl Makefile.PL`
`make`
`make install` (as root)
After that it's done.

Perl Package Manager (Active State Perl on Windows) On Windows platforms you may use the Perl Package Manager (comes with Active Perl) to install the module.

using the CPAN.pm module If the cpan module isn't configured yet, this can be done interactively or even automated during this process.
`perl -MCPAN -e shell`
then enter
`install Image::ExifTool`
at the cpan prompt...

3.8 Digital Elevation Model

There are many digital elevation models on the web, which can be downloaded for free and used with ogie: ETOPO2 [16], GLOBE [17], GTOPO30 [18], SRTM30 Plus (TOPO30) [19], SRTM30 [20], SRTM-3 [20] and SRTM-1 [20]. GTOPO30, SRTM30 (Plus) and GLOBE have a resolution of 30 arc-seconds, 1km. ETOPO2 has a 2-minute grid (4km), but sea-depths too. SRTM-3 is 3 arc-seconds (90m) and SRTM-1 (only available for U.S.) is 1 arc-second (30m).

The "Shuttle Radar Topography Mission" (SRTM) topographic data with resolution 1 arc-second (for USA) and 3 arc-second for almost the rest of the world is available, but due to high resolution not very good for regular flight-analysis. (Graphics hardware won't handle larger areas).

My recommendation is to use either GTOPO30, SRTM30 (Plus). If you have lots of hard disk space and want to analyse flights from many countries, you should consider to build a WORLD.DEM from GTOPO30/SRTM30. If you like to explore the seafloor, you may merge it with the ETOPO2.

SRTM-3 and SRTM-1 should be used for small-scale high resolution application.

Data format

Binary data in 2 byte integer (big endian byte or little endian byte order) format is needed. (You can get these directly from GLOBE, GTOPO30 and ETOPO2 Web sites) Little endian data can be used with the config option: `BIGENDIAN false`.

3.8.1 GTOPO30, SRTM30

The worldwide GTOPO30 elevation model is split up in 33 pieces (tiles). Get the "tile" you need and put the full path to the *.DEM into the configuration-file (see 3.8.8). You also need to set the rows and columns and minima and maxima and grid resolution.

If you have lots of space on your hard-disk and a fast Internet connection you should consider to get all (33) tiles (about 280MB compressed) and use the `create world` program to generate a WORLD.DEM (single file containing worldwide elevation data, really cool!) file:

1. "`tar xvzf`" all tiles to one directory (will need more than 2 GB). You only need to extract the *.DEM files from the *.tar.gz archives downloaded from GTOPO. Use the following cmdline (in the directory with all archives) to extract *.DEM files only:
   `find . -name '*0.tar.gz' -exec tar xvzf {} *.DEM ';'`
2. invoke `createworld` in the same directory (will need another 1.8 GB) Attention: There is not much console-output or error-checking!
3. enjoy the 1.8 GB (!) WORLD.DEM (check WORLD.DEM for its size: should be 1.866.240.000 bytes!)
4. settings for the WORLD.DEM can be found in the default-config file

Now there is a improved SRTM30 model, which is based on the shuttle radar topography mission. So the SRTM30 seems to be a better GTOPO30. The SRTM30 data is available also available for free and can be used to build the WORLD.DEM as described above. SRTM30 data didn't cover the regions south

of 60°S. To obtain a WORLD.DEM file you should take the 6 arctic tiles from GTOPO30, the remaining 27 from SRTM30.

3.8.2 ETOPO2 (and merging it into the GTOPO30)

If you have created a WORLD.DEM (1.8GB) datafile as described above, you can merge it with the bathymetry(sea-depth)-data from etopo20. Get the "etopo20.i2.gz" file from the web. "Gunzipped" it has 116.672.402 bytes. Put the WORLD.DEM and the etopo2.i2 in the same directory and call (in that dir) etopo2merger, which will merge them into a WORLD3.DEM file. Because the ETOPO2 resolution is lower than the resolution from GTOPO30, the additional data-points are obtained by interpolation.

3.8.3 GLOBE

Download the region you need (freely selectable) and make sure that you get the right data format. In the *.hdr file (which you will get too, you can find all needed information to edit the config-file.

These are the options to be selected at GLOBE download page:
FreeForm ND
int16
Mac/Unix Binary

the data file is called *.bin the *.hdr file contains some information you need to edit the OGIE configfile.

3.8.4 SRTM30 Plus (TOPO30)

The SRTM30 Plus elevation model is a merged SRTM30 and GTOPO30, including bathymetry data from several sources. It can be downloaded as a single (1.8GB) data file from [19]. Notice the different settings for DEM_LAT_MAX and DEM_LON_MIN! (differing from what should be used for SRTM30 and GTOPO30 world files). See example configuration file!

3.8.5 SRTM-1 and SRTM-3

SRTM-3 (3 arc-seconds) data is available for free (for north and south-America and for Eurasia). SRTM-1 (1 arc-second) is available for the USA. The data is in .hgt format which is exactly, what OGIE can read. But the data is tiled into 1x1 degree pieces. This might be useful for high resolution analysis of some terrain detail, but is just too much data for normal (glider-)flight analysis (graphics hardware isn't able to handle larger areas). However, you'll find it at [20]. The Documentation folder will give you important information about data-format etc.

3.8.6 SRTM-1 and SRTM-3 finished from seamless server

From the usgs seamless server [21] you can get these data. It can be downloaded in a binary .bil format, which is accompanied by a .blw file, which contains additional information. Attention, there is a half-pixel shift. The actual coordinates for the upp-left corner can be found in the last two lines in the .blw file. It seems that void areas are set to 0, in contrast to the original *research grade SRTM data* which uses -32768. Additionally you will need `BIGENDIAN false`

3.8.7 USGS DEM (30-m and 10-m)

This section is written by Vit Hradecky, *thanks*

Digital elevation data with 30-m and 10-m resolution for the U.S. is now available for free at [22]. The data is broken up into the standard USGS 7.5-min quads. Most of the data is in the newer SDTS

format, while some of it is in the older ASCII DEM format. Fortunately, a utility exists to dump either into a raw binary file, which is readable by ogie. Compile the C source from [23] and execute
`read_dem berlin10m.DEM.SDTS.TAR berlin10m.BIN berlin10m.HDR 0`
This will convert data for the Berlin USGS quad from the SDTS format to the 16-bit binary format, output to berlin10m.BIN, dump the headers into berlin10m.HDR, and set bad data to zero elevation. The output will be in little-endian byte order. Use the `BIGENDIAN false` option in the configuration file to read the data correctly. The DEM latitude and longitude limits can be found in berlin10m.HDR. The 10-m and 30-m data is in units feet rather than meters. Use `DEM_INPUT_FACTOR 0.30488` in the configuration file.

3.8.8 Config file

For details on the configuration see D. This section will only describe the settings for the digital elevation model setup.

In the configuration file you need to specify following lines:
The full path to the used digital elevation data file:

```
DEM_FILE /full/path/to/demfile/W020N90.DEM
```

The number of Rows and columns of data

```
DEM_ROWS 6000
DEM_COLUMNS 4800
```

The Maxima and Minima of our DEM-File

```
DEM_LAT_MIN 40
DEM_LAT_MAX 90
DEM_LON_MIN -20
DEM_LON_MAX 20
```

And the Resolution (0.00833333 for GTOPO30 and GLOBE)

```
DEM_GRID_LAT 0.008333333333
DEM_GRID_LON 0.008333333333
```

Divide by 10 for SRTM-3.

If you are using OGIE for European Terrain (W020N90.DEM) only, you can use the example-values above (you only need to change the path) For other config-file options see D.

3.9 Digitized maps

Since Version 1.2 the digitized maps can be in jpeg format. The file extension should be .jpg (not .JPG or .jpeg etc). The old rgb-format texture maps can be used too, but jpg maps should be preferred (they do not need that much diskspace). Since Version 1.3 the `NUMBER_OF_MAPS` is not necessary anymore.

How to prepare maps First of all you have to use a scanner or digital camera to get your maps to the computer. To avoid differences due to different projections, the map should not be one big tile, but many small pieces. The smaller the better. For a 1:500.000 map (like ICAO) pieces of 40' x 40' are a good choice. For the further processing of the digitized maps a good image manipulation software is needed such as Gimp (the GNU image manipulation program). The pieces have to be cut out from the raw-scanned image(s). Then the pieces have to be straighten out, to avoid any distortions. The latitude or longitude should be constant for each border. I use the transform tool (which can be used to straighten out perspective distortions etc) to define a (distorted) box along the gridlines of the 40'x40' box, as exact as possible. The transform tool will straighten this out to a perfect rectangular box: the map-tile, which should be scaled to some power-of-2 width and height (128x256 or 256x512 or 512x512 or 512x1024 or

or...) otherwise this have to be done internaly in the OGIE, which will take some more startup-time... And you need to know the geographic coordinates of each border. (wgs84) Then save the map tile as jpg image.

Set up the `.ogierc` file For each map-tile the full path to the image-file and the (wgs84) coordinates of the top, bottom, left and right border have to be given in the configuration file. You need to have a section (as follows) for *each* map-tile:

```
MAP_FILE /usr/local/GPLIGC/maps/bremen.jpg

MAP_TOP 53.5
MAP_RIGHT 9.3333333333
MAP_LEFT 8.6666666667
MAP_BOTTOM 52.8333333333
```

The maps can be grouped in sets. You may want to have a map-set for each airfield you fly from (one for your home-location another for you favourite holiday location etc...) Another way to use this feature would be, if you have not enough videomemory to display all maps at one time. In this case you can split up your maps in at least to sets.

Exampls: You have 20 sections for 20 map-tiles in your configuration file. Now you can put a `MAP_CUT` between the first 10 and the second 10 map-tile sections to split into two map-sets. In the OGIE you can toggle these defined map-sets by using "c" and "x" keys. You may specify more than two map sets by using `MAP_CUT` multiple times.

Every map-set can be named with `MAP_SET_NAME name` to select it at startup with `--map-set-name name`.

For Using large textured maps it would be better to have a hardware accelerated videoboard with lots of videomemory.

shifting indevidual map tiles
`MAP_SHIFT_LAT degrees`
`MAP_SHIFT_LON degrees`
if your maps doesnt fit exactly, a shift in latitude and/or longitude may be defined. If `MAP_SHIFT_...` is given, all following map tiles will be shifted (the shift is applied to the border-coordinates), unless the shift is set to zero or to an other value.

rgb-format maps (outdated ... use jpg) This shouldnt be used anymore... except you will use your old maps, or you didnt like lossy compression Every Map-tile should be in headerless .rgb (3 byte per pixel) data-format (I use Image Magick's "convert" to create that format). The size has to be 2^n x 2^n. That means you have to scale the image before. Width and Height should be a power of 2 (pixels). Because the rgb-format is headerless it cannot contain the information about the size of the image. You need to specify MAP_WIDTH and MAP_HEIGHT for each map-tile in your configuration file. There is a Limit for the maximum pixels for each dimension (width and height). You can query this limit by executing "ogie -q". Look for GL_MAX_TEXTURE_SIZE (both values (width and height) have to be less or equal to GL_MAX_TEXTURE_SIZE).

3.10 Airspace

If you want airspace information to be displayed, you should get an OpenAir™ airspace file (the format used by Winpilot) for your region and set up your `.ogierc` file. One keyword declares the filename of the airspace-file, another one sets the default, whether airspaces should be viewed or not.
`OPEN_AIR_FILE /path/to/OpenAir/file`

`AIRSPACE true`
An alternative way would be the command-line options
`--airspace-file=/path/to/OpenAir/file` and `--airspace` or `--no-airspace` to turn them on or off at start-time.

At runtime, airspaces can be switched on or off via the menu or F9. Shift-F9 toggles the wire frame and transparent mode.

3.10.1 How and where to get OpenAir files

On the GPLIGC web-site you may find an Airspace folder in the download area. Some OpenAir formatted files can be found there. Another option is the page of J. Leibacher [24].

I dont know whether the following paragraph is still valid!
Another option is to use the program of `Carl Ekdahl`, which can be found on the *Soaring Server*. That program works on M$-Windows only, but can create OpenAir formatted files from recent *DAFIF* sources. To do that the dafift.zip is needed and the following files need to be copied to the right location inside the airspace-program (from `Carl Ekdahl`) folders: `BDRY.TXT, BDRY_PAR.TXT, SUAS.TXT, SUAS_PAR.TXT`.

3.11 Waypoints

Waypoints can be displayed by *ogie*. The file containing the waypoints can be declared in the ogie-configuration file by the keyword `WAYPOINTS_FILE` or by a commandline argument `--waypoints-file`. Some more keywords and commandline arguments are available to change the default behaviour.

To switch them on or off use the F12 key. Page-up and page-down can be used to change the size of the spheres, the text-size can be changed with shift-page-up/down. Using shift-pos1 or shift-end changes the displayed text (waypoint-long name, waypoint short-name, waypoint-altitude, waypoint-symbol name)

3.11.1 Format of the waypoint file

As there are probably hundreds of waypoint file formats available, I chose a simple one, which I use with my handheld Germin GPS and *gpsbabel* [4]. It has six columns of data: latitude (degrees), longitude (degrees), altitude (metres), short name (max six letters), long name, symbol name. Columns are seperated by whitespaces. Using *gpsbabel* it should be easy to convert any formats to this. You'll find a gpsbabel-style (gpligcwpt.gpsbabelstyle) file in the /share/gpligc folder. Here is an example how to convert a cambridge waypointsfile to the needed format:

```
gpsbabel -i cambridge -f cambridgefile -o xcsv,style=gpligcwpt.gpsbabelstyle -F
mywpts.gwpt
```

The important part is the output format option "xcsv,style=" using the provided style file.
However, *gpsbabel* is cool, you should have a look at it anyway. You can even download your waypoints from a Garmin device like this:
`gpsbabel -i garmin -f /dev/ttyS1 -o xcvs,style=gpligcwpt.gpsbabelstyle -F outfile.gwpt`

4 GPLIGC

4.1 Starting

- Linux,Unix GPLIGC can be started from the command line by typing
  `GPLIGC.pl`
  It's possible to use an igc-file as argument to `GPLIGC.pl`, which will cause the program to open it and pop up the *flight view window*.

- Windows From the command line (MS-DOS prompt): `perl GPLIGC.pl` or double-click `GPLIGC.pl` from windows explorer, if *.pl is associated with the perl interpreter (../bin/perl.exe).

4.2 Menus

4.2.1 File

open You can select an IGC file to open.

reload reloads the opened IGC-file.

download track (gpsbabel) Uses *gpsbabel* [4] to download trackdata from a GPS-device. The used command string can be defined in the configuration keyword `gpsbabel_tdownload`. For details see section A.

download garmin (linux only) Download GPS tracks from a garmin device, using gpspoint. The gpspoint command has to be defined by the `garmin_download` configure option (see section A). The track is then converted to the IGC format by gpsp2igcfile.pl (see section 7.2.1).

download media (linux only) Use this option to download media files (audio recordings, videos, photos) from your mobile phone to see them next to the GPS track. You should specify your mountpoint and folders via `mm_mountpoint` and `mm_download_dirs` (see section A).

export kml Exports the currently opened track to the kml format.

export gpx Exports the currently opened track to the gpx format (useful for openstreetmap).

open photo/multimedia directory If your photos/multimedia files are located not in the same folder as your GPS track, you can chose the folder here.

4.2.2 Options

Draw Options Chose between *lines*, *dots* and *linespoints*. These are Gnuplot modes. Use grid lines or not (in Gnuplot)

Gnuplot This menu is available only on Unix-machines. With windows Gnuplot 4.x will be used by default, for Gnuplot 3.x you have to follow the instructions given in section 3.5. The Gnuplot-shell is not available on windows, sorry. To use the new interactive features of Gnuplot 4.x you have to select the *Gnuplot 4.x* option here. Highlights of these features are interactive zooming (right mouse-button, with history) and interactive rotating of 3d plots. The option *open Gnuplot-shell* will open a Gnuplot-shell for each plot, where you can do some more work on the plot. The terminal application to be used can be changed by the configuration-key `gnuplot_terminal_app` which default is *xterm -e* (see A).

Set terminal Select gnuplot-term (output-option) The x11 option will put the plot on your screen. All the others will write to a file. You will be asked for a filename (for every plot). Specify the file name and extension in the save file dialog.

Optimizer method different methods for the task optimisation can be chosen here. For details see section 4.3.4.

WP cylinder Here you can select the type of way-point observation zones, cylinders or FAI sectors or both. For cylinders the radius can be chosen. Another option is to turn the way-point names in *flight view window* on or off.

WP-Plot side-length Select plot range for way-point-plots (in the gnuplots the ranges used in x and y - nevertheless all gnuplots (track, 3d, or way-point) will be distorted) or zoom in *flight view window*: 1,3,5 oder 10km. (In the *flight view window* the side length is used in y-direction (latitude), x-direction is scaled automatically to prevent distortion. If you use "set range" in the *flight view window*, this value will be used to set the gnuplot range.

Set Noise Level Limit All recorded position points with a noise level above that limit will be plotted in green! (in *flight view window*)

Coordinate Format Here you can select the display format for the coordinates. Choices are DD MM SS or DD MM.MMM

Speed, vertical speed, altitude and distance units Select your preferred units (km/h, m/s, m, ft, knots, ft/min)

Save configuration All Configuration (chosen in the options menu) will be saved to `.gpligcrc` in your home directory. This file is read at program-start.

Reread configuration This will reread the `.gpligcrc` configuration file. This option allows one to change some configuration at runtime...

Photo locator Enables the photo locator feauture. This feature may be used to geo-tag photographs, which were taken while the GPS-track was recorded.

debugging output This option will dump tons of mostly useless data to your terminal. Use this if you like mystic numbers running down your console. You should use this only when requested by the developers to track some bugs...

4.2.3 About/Info

Copyright information and a link to the GPLIGC web-site

4.3 Tools

4.3.1 Flight Info (IGC)

Informations about FDR, pilot, plane and task will be displayed. These comes from the IGC-file-header. The defined task from IGC-file will be displayed too, but doubled way-points in task definition will be removed (e.g. if take-off and start or finish and landing are equal)

4.3.2 Flight Info (additional)

Additional information (which is not stored within the igc-file) can be viewed/edited here. For example the data, which is not contained in the IGC file (it might be encoded in the filename, but maybe not unambiguously). The data entered here can be stored in a `filename.gpi` file, which is automatically loaded, if found.

4.3.3 Flight Statistics

Time of takeoff, landing, and flight. The time of begin and end of un-powered flight is shown also. The un powered flight can be defined in flight-view-window with the keys `s` and `f`, or will be determined automatically (won't be right in any case!).

Oxygen debriefing : This section shows calculations on oxygen, which should have been used according to `FAR 91.211`. FAA requires 1l/min per 10.000ft (regular canulla and mask). Up to FL180 Oxymizer canullas may be used (they use 1/3, values given in brackets). Four altitude bands are distinguished:

FL100–FL125 recommended use of oxygen

FL125–FL140 FAA requires oxygen in access of 30 minutes. Recommended: always

FL140–FL180 up to FL180 Oxymizer canulla is officially allowed

FL180–very high only with mask

The sum is given for recommended oxygen use (strictly from FL100) and FAA conform from FL125 (in access of 30 minutes) or FL140.

Dont forget to do the elevation calibration before calculating the statistics and to set the QNH. For details of the calculation see `GPLIGCfunctions::OxygenStatistics`.

Task The task (which is given in the recent task definition; defined by task-editor, via optimisation, or from igc-file) is shown regardless whether the way-points are reached or not. The task speed is calculated from the total task distance and the unpowered flight-time, which may be adjusted as described above.

The third section will show the flown task, only way-points which you have reached are taken into account. A way-point is reached, if one logged data point is closer to the way-point as the cylinder radius (or 3km if only FAI sectors are chosen). The time of reaching the way-point is taken from the first point inside the way-point radius. For exact analysis of speed you should use the F5,F6,F6 measuring function. For each leg the distance, speed, altitude gain/loss and the glide ratio (calculated from distance wp1-wp2 and altitude gain/loss) is displayed.

4.3.4 Task Editor

Select a way-point of the current task and make 2d or 3d plots of it. You can also delete way-points from task, or set last wp equal to the first one (to close a triangular flight etc)

Optimisations For all optimisations it is necessary to check that the begin and the end of the free (unpowered) flight is set correctly. Otherwise way-points my be set at a part of your flight, where you have been towed or using a motor. To avoid that you need to set (or at least check the automatic detected) the "begin unpowered time" to the beginning of the free flight (release point or engine-off point). The "end unpowered time" needs to be set if you used an engine before landing.

The optimisation will only take the data between "begin of unpowered flight" and "end of unpowered flight" into account. GPLIGCs optimisation routines are based on *Metropolis Monte Carlo* (MMC) and/or *simulated annealing* (SA) methods. One of them can be chosen in the menu (the default can be set using the config key `optimizer_method` to either "mmc" or "sa").

Several configuration keys can influence the algorithms. `optimizer_cycles_mmc` and `optimizer_cycles_sa` define the number of optimizer runs for MMC and SA, respectively (each run is represented by one step of the progress bar).

`optimizer_mmc` sets the commandline parameters for the optimizer run, if MMC is used. `optimizer_sa` sets the commandline parameters for the optimizer run, if SA is used.

If you want to play with that, check the source code of `optimizer.cpp` In general, the default settings for both methods should find the optimal task. I'm looking for feedback on their speed and reliability E.g. if they sometimes stuck at local maxima, missing the global one.

`optimizer_verbose` and `optimizer_debug` may enable the corresponding output of the optimizer at the console.

For brute-force calculation computers are still too slow, since a typical flight with about 5000 data records and a task of 7 waypoints, gives about 10^{22} solutions to check. The `optimizer` c++ code (see optimizer.cpp in the source) implements several experimental methods to find the best task.

Optimisation OLC-classic (rules oct/2007) This will find the best task for OLC-classic (maximum points) and set it as the task. The OLC-classic optimisation will find the probably best task with 7 waypoints (6 legs). The value to be optimised is the raw scoring: 4 legs with 1 point per kilometre, leg number 5 with 0.8 points per kilometre and the last leg with 0.6 points per kilometre. The altitude limit of 1000 m between the lowest point between "begin of unpowered flight" and the starting point and the highest point between the end point and the "end of unpowered flight" will be accounted for.

Optimisation DMST 2005 This optimisation will find the best task according to the german DMSt 2005 rules. FAI tasks will be found, if possible.

This will not check for pre-flight declared tasks. It's still up to you, to check that. But if you've finished a pre-flight declared task, you probably will know that.

HOLC 2005 This optimisation will find the best task (maximum points) for the hang-gliding/para-gliding online contest (rules of 2005). Triangular tasks, or FAI tasks will b used, if they'll have more points. Every logged data point is used (if it is valid).

output of optimisation If the optimisation is finished, a window with some information will appear. Some of them need to be explained. Time of departure: This is the time of the lowest position after begin of un-powered flight and the first way-point (start-point). Finish-time: This is the time of the highest position after the last way-point and the end of un powered flight.

4.3.5 FlightViewWindow (FVW)

The Flight-track will be displayed. Cross-marks will show the actual position, the data of which is displayed above. The displayed data is based on the barometric altitude. In the case, that the igc-file does not contain any barometric altitude, GPLIGC will switch to GPS-Altitude mode. A message will inform you about this, and "GPS-Altitude modus" is displayed in FVW. Move forward, fast forward, backward and fast backward with F1-F4. "t" toggles task-display. "r" sets the Gnuplot-range (with chosen side-length in menu) next to the cross-mark position "c" toggles way-point-cylinders and sectors (on/off). (You should delete doubled way-points from the task before - if departure and start location are equal, or finish and landing, orientations of the correct way-point sectors cant be calculated) "z" zooms in or out! The 'distance to' can be changed by selecting a different way-point in Task Editor Window. You can define a task start and finish point with "s" and "f", these points are used to calculate the task-speed. These points are marked with small black circles. They are used to mark the begin and end of un powered flight also (important if you want to claim the flight for the online contest). GPLIGC will try to detect the begin and end of un powered flight, but this may fail sometimes and should be checked by the user.

Way-points can be set with a, b or p (b adds before actual wp, a after and p replaces the actual wp - actual wp is that one which is shown in TaskEditor Window, if opened)

You can zoom in by selecting an area with right mouse button. return to full view using "z" key.

Select points by clicking in the barograph or the track, cross-marks will move to the selected position. q will close the FlightViewWindow Esc exits GPLIGC

postscript output output of flightviewWindow to postscript is possible by pressing "o" (flight-track) or "i" (barogram)

Resizing of FlightViewWindow After resizing the Window you need to press "y" to redraw the content.

Statistics (thermals/glide) Thermal statistic (F8) will open two windows. One with some statistics and a second one with a list of thermals (double click on a list-entry will jump to that thermal in FlightViewWindow) Glide statistics (F9) will open two windows. One with some statistics and a second one with a list of glide-distances (double-click will jump to that glide-distance)

Statistics (selected range) and wind analysis (F5, F6, F7) can be used to set a first (F5) and a second (F6) point and display some statistics (F7) between these selected points. The selected points will be marked with small green circles and the time-span will be marked in red in the barogram.

Furthermore, a difference plot of airspeed-groundspeed is performed via gnuplot. Even if no airspeed is present (then its assumed as 0), information about the wind can be derived. The difference is plotted vs. the heading, so select a range where many headings are present (e.g. circling). A Sinus function is fitted to the plot. If the gnuplot-shell option is activated, the parameters of the fit can be accessed through gnuplot. The *offset* should be zero, if there is no difference between the recorded airspeed and the groundspeed. Negative offsets will occur if the recorded airspeed is too small. The amplitude of the Sine function will give the speed of the wind, the direction can be read from the position of the maxima.

To obtain a better view of the selected range press F11. The rest of the track will be omitted, and the barogram will be zoomed to the selected range (F11 again will restore the previous view).

Save lifts You may save interesting points (lifts, wave-entry positions and the like) by pressing F10. The actual position including altitude, vertical speed, time etc. is saved to `filename.lif`. This file is being appended, so you have to delete it, in order to start from scratch.

This file can be used in *ogie*, and will be automatically loaded if *ogie* is started from within *GPLIGC*.

Altitude calibration The barometrically recorded altitude is often shifted in respect to the real altitude! In the simplest case, this is a constant shift (constant option). To correct this, you can select a point with known altitude in FlightViewWindow (the airfield elevation (before takeoff) for example) and press "e". Then you have to enter the known elevation at this point, and the data will be shifted. The barometric option will use a barometric model, the shift will decrease with altitude. Chose the model according to your suspected error. Use 'constant', if you have a constant systematic error in your altimeter. Use 'barometric' if the reference pressure is wrong.

QNH and reference Pressure calibration If your recorded track has the altitude referenced to MSL, you can enter an QNH (normalised pressure) to benefit from more accurate pressures and Flight Levels. (Using key "n" in fvw, entering only pressure for QNH, leaving 0 for reference pressure).

If you recorded track is referenced to a known pressure-level (e.g. 1013.25hPa) you can change the reference to MSL by entering the reference-pressure and the QNH via "n" in fvw.

View Photos / Multimedia By pressing "v" in the FlightViewWindow, the closest photo or multimedia file will be displayed, either in the internal viewer, or in an external one.

To set the viewer the configuration key "picture_viewer" can be changed. "internal" would tell GPLIGC to use the internal one (only for pictures), every other value will be used as executable. For example you may use "kuickshow" or "/path/to/any/strange/picture/Viewer/you/like/view". For other multimedia (audio recordings, movies) the configuration key "mm_player" is used (defaults to mplayer).

To show/hide the files at the track, use key h in the flight-view-window to toggle. Pressing m will bring up a list of associated files.

Photo time calibration and geotagging Often, the clock of the digital camera (or cell phone) isn't exactly in sync with the 'official GPS time'. Therefore, we need to synchronise with the GPS time. This can be done, if we have the exact time or position of one photo. (I use to take one photograph of my GPS showing the GPS-time [preferrably UTC]). The procedure in GPLIGC: (1) view the photo in question (using key `v`). GPLIGC will remember it. (2) press "x" and enter the exact time (in UTC). The determined shift will then applied to all photograph. GPLIGC will write a file (.GPLIGC-timeshift) in the directory of the photos, so that this correction will be remembered. To avoid all this the best method would be to have the cameras time in sync with the GPS-time. To make the correction easier it is a good idea to take a photo of your GPS showing the time.

After you have done the calibration you may want to geo-tag your photographs. Pressing `u` will do the job: The GPS coordinates will be written into the Exif headers of the images.

Important! IGC files use UTC. Your camera probably uses local time. Therefore, GPLIGC has to know about your local time. The offset should be set in `.gpligcrc` (key "timezone"). The timezone offset can be set independently for each IGC-file using the *additional flight info dialog* (see section 4.3.2). Once a calibration has been done, the calibration shift and time-zone offset will be saved in `.GPLIGC-timeshift` for that specific folder of pictures. In order to override the timezone from `.gpligcrc` one can create an `.GPLIGC-timeshift` in the directory with the pictures, containing two lines: line 1 should only contain 0 (thats the timeshift without timezone), line 2 should contain the timezone. If your time-zone offset is so large, that the photos cannot be located on your tracklog, you should create `.GPLIGC-timeshift` file manually, containing a suitable time-zone offset and reload the picture-folder.

4.3.6 OGIE – 3d

Starts OGIE with the currently opened flight-data file For details on OGIE read section 5

4.3.7 Logger Read Window (Windows only)

I don't know whether this still works, as I remember some problems with these small DOS programs on XP. Consequently, this feature will be removed in the future.
For each IGC approved logger a so called 'small DOS program' has to be available for free. The small Windows/DOS (they don't even run on any Windows ...) programs can be downloaded at the FAI web-pages.

To use these software from GPLIGC logger read window you should put the `data-xxx.exe`, `vali-xxx.exe` and `conv-xxx.exe` somewhere in your path (or in the GPLIGC installation directory, which should be in path).

Use this with caution! I know that (for example) some versions of data-sdi won't work on XP. There may be problems with the others too.

4.4 Gnuplots

4.4.1 Plot output (2d and 3d plots)

Plots will be written to 'term' (selected under "set Term"). Directly to X11 or after "Save-File-Dialog" to a file.

4.4.2 Ranges

Choose x,y and/or z-ranges for the graphs to be produced with gnuplot. Wrong entries will cause empty plots or gnuplot error-messages. If you have set ranges (by WP-plotting or by hand) you will have to

clear them before doing some other plots!

4.4.3 3D View Control

Set view direction for all 3d plots.

4.5 Photo locator

If the Image::ExifTool module is installed and the configuration key "photos" is active (=1, this is the default), GPLIGC can locate pictures and show them next to the track, which allows you to identify the places where these pictures have been taken. Pictures not featuring an Exif header, may be located by the files timestamp, if thats retained. When opening a igc file, GPLIGC will look for JPEG photos in the same directory, if there are none, GPLIGC will look in the "photo_path" directory (as set in .GPLIGCrc). The third and best method to tell GPLIGC where the pictures are, is to use the "open photo directory" from the file menu. Just select one of the JPEGS there. Since there is no date in igc files, you should point GPLIGC to photos from the same day. For more details about time-zone, and time offsets see 4.3.5.

4.6 The GPLIGC configuration file (.gpligcrc)

This paragraph describes the new configuration file format of GPLIGC, which was introduced with GPLIGC 1.5. Internally, GPLIGC stores all changeable configuration parameters in a 'perl-hash'. This is a data-structure, which is represented by pairs of keys and values. Each key can be assigned to a value. To get a valid .gpligcrc file, you should start GPLIGC and use options/save configuration. This will write a .gpligcrc file, which includes *all* valid keys and their default values. The file has one line for each key-value pair. The key is the first word, the value is enclosed in `"`. The value can be changed with a text editor. If you do this while GPLIGC is running, you need to select *options/reread configuration* to trigger rereading of the changed configuration file.

On Windows systems this configuration file is named `gpligc.ini`

Please refer to the section A to see what values are allowed. If illegal values are used, GPLIGC may behave unpredictable or just crash.

4.7 Remarks for using IGC files with GPLIGC

GPLIGC does not check the integrity of the data. Some calculations may not work as supposed, if there is more then one flight contained in the IGC file (e.g. the times of take-off, landing and flight duration will be wrong (starttime=starttime of first flight, landingtime=time of last landing)

5 OGIE

5.1 Get started

To start *OGIE* press the "OGIE – 3d" button in *GPLIGC*, or use the commandline (xterm, console or MS-DOS prompt):

```
ogie igcfile.igc
```

There are three different modes to use *OGIE*:

- IGC-file mode: You can give an igc-file as a single argument. If you use more than one argument on the commandline, you need to specify the igc-file by giving `--igc-file FILENAME` (or `-i FILENAME`).

- Terrain viewer: Select the centre of your view with `--lat` and `--lon`. The size of the area can be selected with the following options: `--border, --border-lat, --border-lon`
- GPS live view: Use the option `--gpsd`. *ogie* will connect to the local gpsd [25] and obtain positional information for a live display of your location.

5.2 Menus

The pop-up menu is accessible by pressing the right mouse button in the *OGIE* window. Most important options can be changed here.

5.3 Mouse control

The direction of view can be controlled with the mouse, the mouse pointer is invisible and cannot leave the window, unless mouse control is disabled by pressing m.

Moving the mouse while the left button is pressed will result in rotating your position around the centre of the scene or around the position of the marker (if activated). Dragging the mouse up and down, with the middle button pressed, will shift your position towards or away from the centre or the marker position (if marker is activated).

Moving while the middle and left mouse button is pressed will shift the scene.

5.4 Joystick control

A joystick is supported on Windows platforms via GLUT. On Unix/Linux (X11) the joystick can not accessed via GLUT (because GLUT never supported joysticks on X11). If you want to use your joystick on X11, you have to install freeglut [10].

The joysticks x,y and z-axis will move the viewpoint to the side, forward-backward and up-down. How much the viewpoint will be shifted can be set in the configfile (JOYSTICK_FACTOR_X,Y,Z, see D).

5.5 Keyboard control

For information on the Keyboard functions you should read the section B. In the pop-up menu a *help* is present, which will show the most important keys. If you like to change the controls, edit in `KeyPressed` and `specialKeyPressed`-functions in `GLexplorer.cpp` and recompile.

If you need the mouse pointer, it can be made visible by pressing key *m*.

5.6 GPS live mode

The commandline `--gpsd` enables the GPS live mode. *ogie* will connect to the local gpsd and retrieve its position. Subsequently, a track is build up by the GPS information. If the Movie-Mode (see section 5.17.3) is enabled, the marker is always kept at the actual position. Otherwise the marker can be moved freely, as with an IGC-file. The info display (see section 5.16) shows some additional information: Sat/Mode: number of used satellites, GPS mode (2D or 3D). eph/epv: estimated horizontal and vertical errors. Interruptions: count of interruption of the GPS signal.

To use this mode *ogie* has to be build with gpsd support.

5.7 Digital elevation model (terrain) viewer

OGIE can be used without IGC-Files (as a Terrainviewer). Give the centre of the area (which you want to watch) in decimal degrees

```
--lat 53.5 --lon 8.5
```

as commandline parameters to `ogie`, negative values for southern and western hemisphere. With the argument `--border km`, the half sidelength of terrain in kilometers can be set. You may specify the borders separately with `--border-lat km` and `--border-lon km`. If you want to watch very large areas, you can use `--downscaling n` where n is an integervalue bigger than 1. This will force the program to use only every n-th datapoint from the elevation model.

5.8 Colourscaling

The following colourmaps are available:

1. red - rainbow - white
2. green - red - white
3. black - white
4. dark green - red
5. magenta - light blue
6. black - rainbow - white
7. white
8. black - red - yellow - white

2 Colourmaps are used for terrain colourscaling. One upper (normal) colourmap and another (lower) colourmap for the terrain beneath sealevel. The value taken as sealevel can be set by the commandline switch `--sealevel m`. The colourmaps to be used can be set by `--colormap-sea n` and `--colormap n`. The upper colourmap can changed interactively by pressing keys 1-6, the lower (sea) colourmap can be changed with F10 and F11. The default colourmaps can be set in the configuration file: `COLORMAP n` and `COLORMAP_SEA`
Yes, the spelling of *colour* in all parameters and cmdline options is *color* [amer.]

Optimising colourmaps By default the colourmaps scale their colour-ranges from minimum to NN, and from NN to maximum elevation (of displayed terrain). If you want some more "aggressive" colourscaling you can specify the minimum and maximum by giving `--colormap-min m` and `--colormap-max m` Arguments are heights in Meters. If the sealevel is outside the range (min, max) only one of the colourmaps will be used.

Example `--colormap-sea 1 --colormap 3 --colormap-min 20 --colormap-max 3500 --sealevel 600` This will cause *OGIE* to use first colourmap between 20m and 600m, the graymap between 600m and 3500m.

Sealevel2 If you prefer a flat blue ocean surface instead of seafloor-terrain: Setting a `--sealevel2 n`, will cause the explorer to set a ocean-like flat blue surface at an elevation of n meters.

Sealevel3 Almost like seavel2, but sealevel3 will be a transparent surface, through which the seafloor can be seen. Setting a `--sealevel3 n`, will cause the explorer to set a ocean-like flat transparent blue surface at an elevation of n meters.

5.9 Maps

If you have defined map-sets (as described in 3.9) and turn them on, (`--map` or `MAP true`, by menu or pressing "b") then only the terrain covered by the defined maps will be displayed. If you're using an elevation model as well, the maps will be put on the surface (if the terrain-mode is active: `--landscape`, `LANDSCAPE true`, or activated by menu or pressing "l"). The modulation mode (or coloured map mode) can modulate the the maps with colourscaling (on/off F8, "coloured maps" from menu. Another way would be to put `MODULATE on` in the configuration file).

5.10 Stereoscopic viewing

Four stereoscopic modes can be used. Three of them are *runtime-options* and can be activated by menu.

Double image `--stereo` (or `STEREO true`) will display 2 stereoscopic images. You can cross the optical axis of your eyes to get a "real" 3D image (squinting). (Left eye sees right image and vice versa). Maybe someone will use the "parallel" method (right eye sees right image, and left one left). Then you should swap the images (press "A").

Anaglyphic modes `--stereo-rg` (or `STEREO_RG true`) `--stereo-rb` (or `STEREO_RB true`) For these stereoscopic modes you will need either red-green or red-blue 3D-glasses, if left eye is red, you need to swap the images (press "A").

Hardware 3D with shutterglasses If you own a quadro-buffered openGL-card (like nVidia Quadro2, Quadro4...) and some shutterglasses (or other professional stereo-equipment) (and the X-server is configured for stereo) you can use the `--stereo-hw` option. *OGIE* will try to get a quad-buffered window. This mode can be initialised at start time only.

Eye distance For adjusting the strength of the 3D-effect you can change the distance between the "virtual" eyes (`--eye-dist km`, "Q","W" or `EYE_DIST value[km]`)

5.11 Projections

The flightdata, digital elevation model etc. have to be mapped from earths wgs84 coordinate system to a flat surface. This can be done by using different map projections. *OGIE* offers you four of them. Which one to be used has to be chosen at start-time of the program. You can use a commandline switch to set the map projection, or you can set a default in the configuration file. The earth is assumed to be a perfect sphere with a radius of 6371km.

5.11.1 Projection 1 - cylindric

`--projection-cyl-platt` is the commandline switch for this projection. In the configuration file `PROJECTION 1` can be used. The spheres surface is projected to a cylinder, which is parallel to the axis of the earth and which has the same radius as the sphere. The equator of the sphere is the standard parallel which touches the cylinder. The projection is orthographic.

5.11.2 Projection 2 - Mercator

`--projection-cyl-mercator` is the commandline switch for this projection. In the configuration file `PROJECTION 2` can be used. This is the well known *Mercator* projection.

5.11.3 Projection 3 - cylindric

`--projection-cyl-no1` is the commandline switch for this projection. In the configuration file `PROJECTION 3` can be used. This projection is a cylindrical projection, but not geometric. The equator is a standard parallel. The longitude conversion is done like a geometric projection. Latitude is converted in a way, that distances along meridians are preserved.

5.11.4 Projection 4 - pseudo cylindric

`--projection-pseudo-cyl-no1` is the commandline switch for this projection. In the configuration file `PROJECTION 4` can be used. This is the default projection, which is best suited for small areas. Distances along parallels and meridians are undistorted.

5.12 Screenshots

Screenshots can be made using the "p" key for a single shot, or "shift-P" for the continuous screenshot-mode. In the continuous mode every rendered frame is saved. The output format can be specified using the `--image-format format` option or the configuration file keyword `IMAGE_FORMAT format`, where `format` is one of the following: `jpg, rgb`. The names of the image files will start with *frame1000* and the numbers increase. A different basename can be specified by either `--basename string` or `BASENAME string`. Also a path can be given, where to save the screenshots (`--save-path string` or `SAVE_PATH string`)

Jpeg

If the output format is jpeg (which is the default), the jpeg-quality can be set with `--jpeg-quality int` or in the configuration file `JPEG_QUALITY int`, where `int` is a number between 0 (lowest quality) and 100 (highest quality).

rgb

While using the rgb format you should keep the information about the image sizes, because this information is not saved within the image. It is a 6-byte per pixel rgb image. You can use ImageMagick's *convert* to convert these into almost every available image format. For example:
`convert -size widhtxheight -depth 16 -endian lsb frame1001.rgb outfile.png`

5.13 Offscreen rendering

The offscreen rendering function via GLX is available with the linux-binary, but requires GLX 1.3. Offscreen on windows and/or via mesa needs special compilation...

OGIE can be used as an offscreen 2D/3D renderer. As in the "onlineplotter", which can be tested on the GPLIGC website. With this function some contest results can be made more visible etc.

Single images can be rendered offscreen. Two modes are available. For image format related options see 5.12.

GLX offscreen (pbuffer) *This doesn't work with any Windows! GLX is available on unix/linux only* Offscreen rendering is done using GLX pbuffers and requires GLX 1.3. Rendering is done hardware accelerated, but requires the X-server running and accessible. Commandlineswitch `--offscreen` is needed and a filename for the output can be given by `--os-outfile filename`

Mesa offscreen (osmesa) *This doesn't work with the precompiled binaries! Special compilation is needed* Commandlineswitch `--os-mesa` is needed and a filename for the output can be given by `--os-outfile filename` In this mode the rendering is done with the mesa library, but it is hardware independent, no X-Server and no graphics hardware is needed. For mesa-support *OGIE* has to be compiled as described in section 3.1.

Viewpoint All other Commandline parameters can be used and the configurationfile will be used. Important are the `--init-...` parameters to set the viewpoint and viewdirection. `--init-pos-N, W, S, E, NE, SE, SW, NW` can be used to set the initial position to one of the borders or corners of the terrain. The view direction will be set to the centre, if not specified separately (can be used to set the initial position for the interactive mode too).

5.14 Performance

Using the option `--verbose` (or "VERBOSE true" in the configfile) will give you the information how many triangles are used to build the surface. (if DEM is used) Check by yourself how many triangles you system can handle at a tolerable speed. The rendering time is also dependent on the quantity of textures used.

In Movie-Mode ("I") with `--verbose` a framerate is displayed...

A hardware accelerated openGL setup is recommended. I'm using a nVidia GeForce2 MX400 (64MB) and that one is quite good for most flights.

5.15 GPS/Baro alt

OGIE can display the flighttrack based on barometric or GPS altitude: default behaviour is set in configuration file (GPSALT true—false), Without configuration file default is barometric. cmdline-switches `--baroalt` or `--gpsalt` can be used.

5.16 Info

F6 can be used to switch on/off the info-mode. In infomode the viewpoint position is displayed at the top left corner of the screen. In markermode some more information is displayed. The units of speed, vertical speed and altitude can be changed by using factors to convert from standard (km/h, m/s, m) to another unit. These factors can be specified in the configuration file (`SPEED_UNIT_FAC, VSPEED_UNIT_FAC, ALT_UNIT_FAC`). The names of the units can be set using `SPEED_UNIT_NAME, VSPEED_UNIT_NAME` and `ALT_UNIT_NAME`. (See D. The timezone can be changed from UTC to localtime using the `TIME_ZONE` and `TIME_ZONE_NAME` keyword in the configuration file.

5.17 Marker

F7 activates the "Marker". A huge red arrow pointing to a position of a logged datapoint. The arrow can be moved forward (F3), backward (F2) and fast forward (F4), fast backward (F1). If the info-mode is active, some data of the marked position is displayed.

5.17.1 Marker-Range

If your flighttrack crosses the same place several times, you may want a part of the flighttrack to be displayed only. With `--marker-ahead n` and `--marker-back n` you specify how many datapoints before and after the marker will be plotted. Default values are 50 back, and 0 ahead. To turn the marker-range-option on, press "shift-U" (*not "u"*), or use cmdline-switch `--marker-range`. This may be turned on by default by using `MARKER_RANGE true` in the configuration file (where the range-defaults can be defined also)

5.17.2 Follow-mode

The viewpoint will follow the marker-position. If this option is turned on by default, can be disabled by cmdline-switch `--no-follow` or `FOLLOW false` (in the configuration file).

5.17.3 Movie-Mode

The Movie-Mode can be switched on by pressing "shift-I", selection from the menu, or with `--movie`. The Marker position is continuously moved forward. Using this mode together with the follow-mode and marker gives a nice movie of the flight.

If it is too fast, you can define a `MOVIE_REPEAT_FACTOR int`, which will render every frame multiple times before shifting the marker. This factor can be changed at runtime with shift-F1 and shift-F2. `MOVIE_REPEAT bool` will switch this on or off. Leaving this off and setting a `MOVIE_REPEAT_FPS_LIMIT float` will automatically enable the repeating-mode if a certain framerate is exceeded.

deprecated slow-down method
A default delay can be set in the configurationfile: `MOVIE_TIMER` the argument is in milliseconds. (compiled-in default is 1 msec). This introduces a "sleep" command, which reduces the responsiveness of the program. Not recommended.

6 Tutorials

I recommend to work step-by-step through some of the examples, to learn about some special features in GPLIGC. The used igc-files can be found at the download section of the GPLIGC webpages.

6.1 GPLIGC – competition flight analysis of 482zc251.igc

6.1.1 482zc251.igc

This flight has been done at the third day of the Klippenenck-Competition 2004. I won this day 0.1km/h faster than the second one in the 15m FAI class. The plane was a ASW20. You can download the igc file of this flight from the GPLIGC web site (download/examples), to practice using GPLIGC.

6.1.2 General information, altitude calibration

Right after opening the igc file in GPLIGC the main window (fig. 1) and the flight-view-window (fig. 3) are present. In the flight view window we will see a lot of confusing way points. What has gone wrong can be seen in the flight information window (fig. 2). The flight information window shows all the header lines which are contained in the igc file. We can check all available information about pilot, plane and logger here. We also can see, that two tasks are defined, the one from the previous competition days is still present in the file. We will open the task editor window (fig. 4) and delete all waypoints from the previous days task. The remaining task should look like `AP 3 TUTTLING - KIRNBERGSEE - FREUDENSTADT - HOHENZOLLERN - HARBURG - ZL 2 KLIPPENE`. The flight view window will show the task like this (fig. 5). The altitude measurement of the logger uses a fixed reference pressure. To get valid MSL altitudes we have to calibrate the data. As we know the elevation of Klippeneck airfield (970m MSL) we can do this easily. We have to find a logged position (by left-clicking in the flight view windows barogram strip or using F1-F4 to move the cursor), where the plane was on the airfield, right before take-off. Logged altitude is about 920m. Then we have to press `e` and enter the real altitude of this position (970m). Now the calibration has been done.

6.1.3 Starting and finishing time, overall task speed, task distance

To determine the overall task speed we have to define the time of crossing the starting line as well as the time of reaching the finishing line. To do this we will zoom (drag a zooming box using the right mouse button) into the starting point/line (fig. 6). We have to find the last position before the plane crossed the starting line. We will find this at 11:02:03 UTC. To define this as starting point we press `s` with the cursor marking this position. The starting position will be marked with a black circle after this procedure. The altitude of the plane while crossing the starting line was 1930m which is below the limit of 2000m, but the groundspeed of 165km/h exceeds the 150km/h limit, but luckily I did not get any penalty points for that.

Now, we zoom out (pressing `z`) and then in again (dragging a box with right mouse button) to magnify the area at the finish. We will find the first point after crossing the finishing line at 15:47:16 (fig. 7) and mark it with `f`. A black circle is shown here also.

After defining the start and finish times we can open the flight statistics window (fig. 8). The first part shows information on the flight (take-off and landing time, total time of flight) and also the defined task start and finishing times and total time of task. The second part of the window gives information of the task, as defined in the task editor. The overall task speed is calculated from the task distance and the total time of task. In this case it was a 434.93km task, done with 91.5km/h. The third part of this information window gives a rough determination of the flown task. A waypoint is considered as reached if there is any position closer than 3km. The first point, which is closer than 3km is used for this statistics. For each leg of the task some information are given. For exact analysis of speeds for some legs you should use the F5,F6,F7 statistics (see 6.1.5).

6.1.4 Thermal and glide statistics

Pressing `F8` or `F9` in the flight view window will open a list of thermals (fig. 9) or a list of glide distances (fig. 10). Additional to each list another small statistics window is opened. The lists and statistics will be evaluated between the set start and finish time. The thermal statistic points out, that we gained 8272m of altitude while circling in 25 thermals. The best one had a lift of 3.44m/s, the worst 0.67m/s. The average lift was 1.64m/s (this is calculated from total altitude climbed and total time spend circling). The time spend circling is given in percent of the overall task time. The best lift was only used to climb 220m, as can be seen in the list of thermals. Double clicking of items in the list will set the cursor to the associated thermal. This way we can find out, that the most altitude (616m) was gained at a lift of 1.7m/s at the last waypoint (`HARBURG`). In this uplift I was circling 11 times to the right which needs 6 minutes.

The list with gliding distances shows, that the longest gliding distance was the final glide of 33km. It took me 12 minutes with an average descending speed of 1.95m/s. The average speed was 165km/h, the average heading was 245°. Average L/D ratio is calculated to 24. Another very nice glide can be found at 12:01:59. It only needs 5.5 minutes at 159km/h to fly 14.6km with an average L/D ratio of 88. The small glide statistics window provides some sums of the gliding list.

6.1.5 F5,F6,F7 statistics and measuring tool

To get some statistics between some positions the `F5,F6,F7` tool can be used. We'll now get some exact information for the second leg (from `KIRNBERGSEE` to `FREUDENSTADT`. We'll find the last point in the FAI sector of `KIRNBERGSEE` at 11:24:27. This can be marked as first point `F5`. A green circle will show up. The second point (reaching FAI sector of `FREUDENSTADT` 11:59:05) will be defined using `F6`. Another green circle will mark this position, and the range between the first and the second point is shown in red (barogram). Now we can get some statistics between these points by pressing `F7`. The result show that the average speed was 103km/h, although I gained 293m.

6.1.6 OLC flight optimisation

We would like to evaluate the olc scoring distance of the flight. Therefore we have to set the start point (pressing `s`) to the release from tow, and the finish time (pressing `f`) to the landing time. I found 10:15:07 for release and 15:47:43 for landing. Now we can start the optimisation from the task editor. The optimisation will find the best olc-task between the release and the landing time. It's 438.14km.

Figure 1: The GPLIGC main window

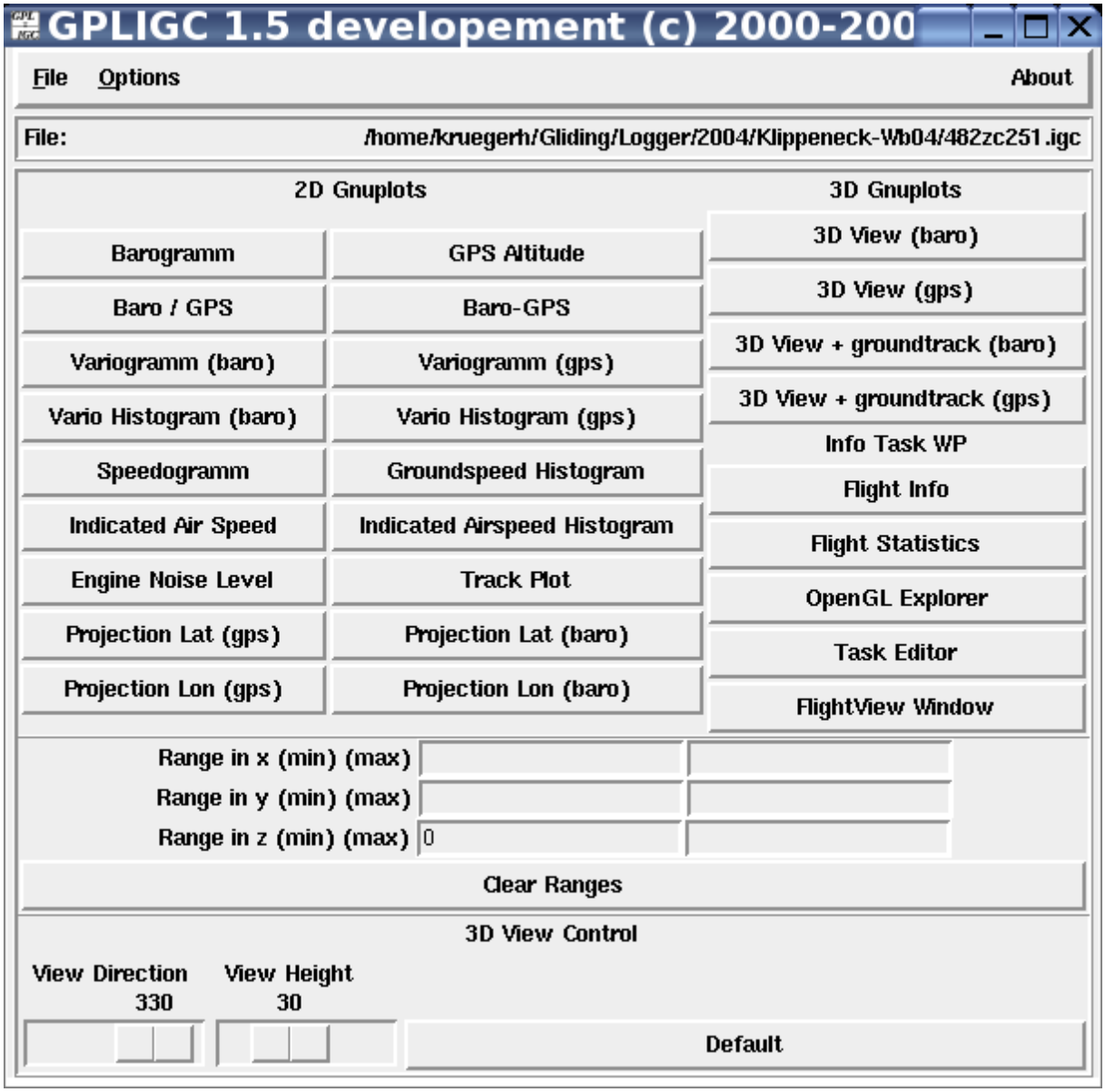


Figure 2: The GPLIGC flight info window

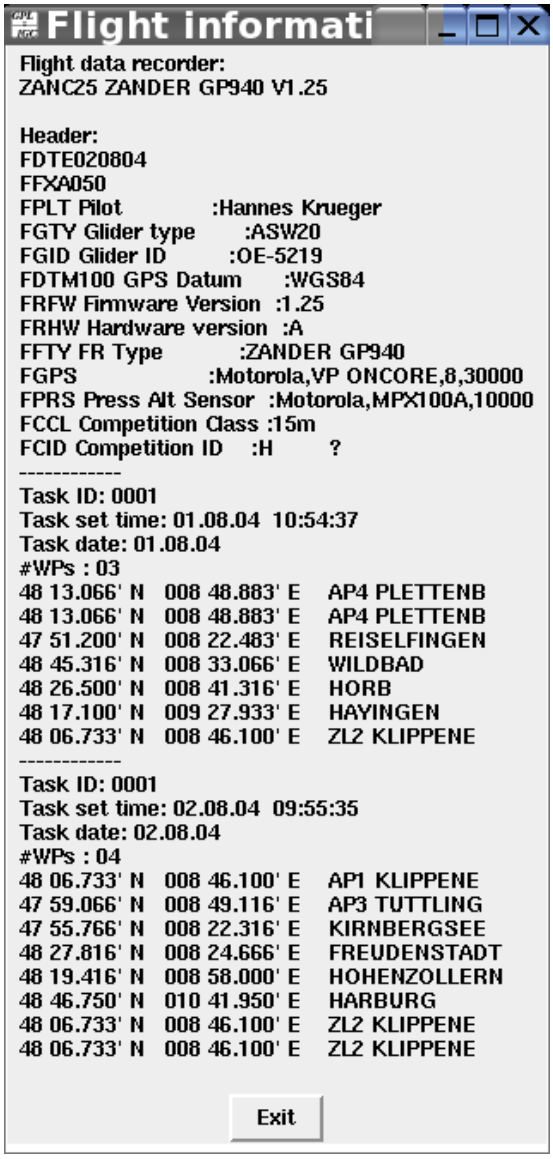


Figure 3: The GPLIGC flight view window showing a flight with two tasks

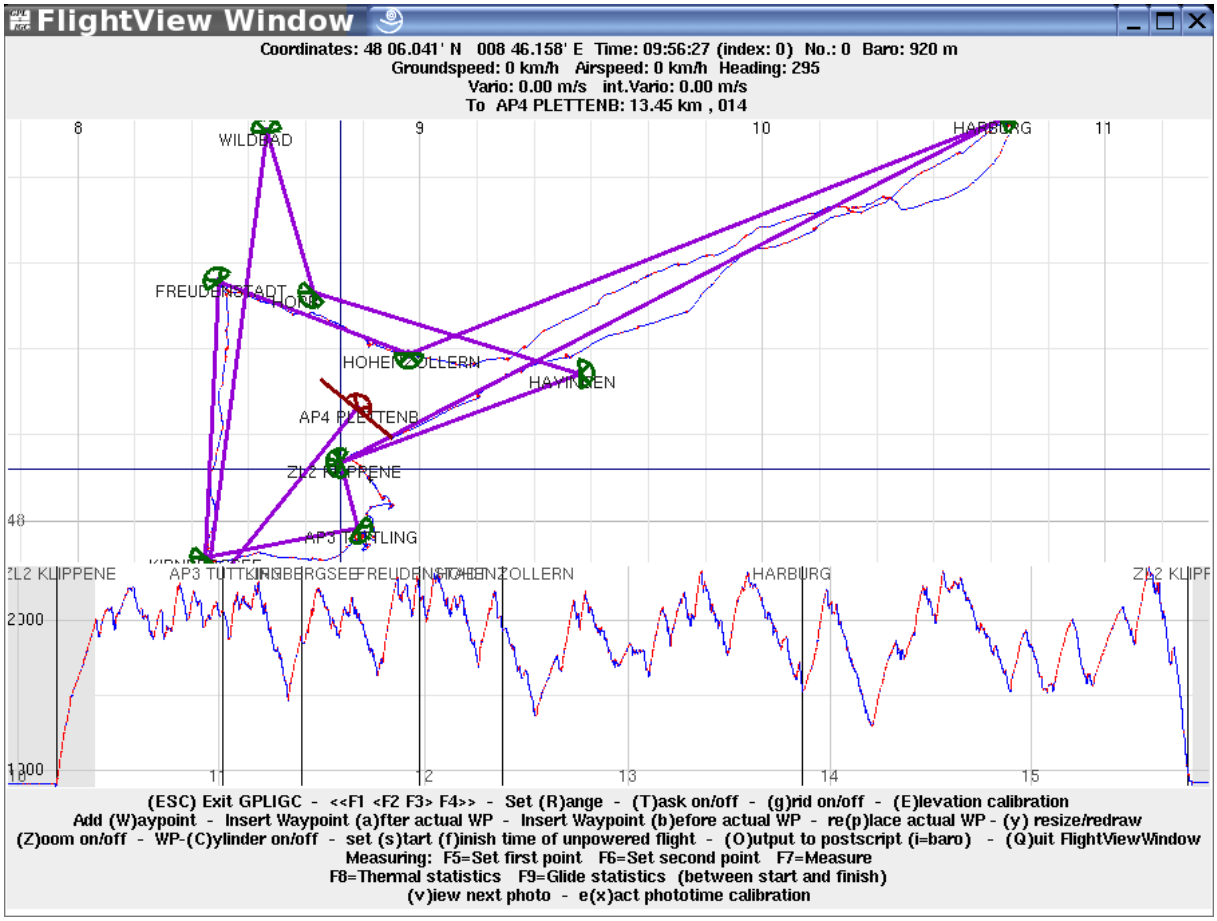


Figure 4: The GPLIGC task editor window

Figure 5: The GPLIGC flight view window, with the valid task

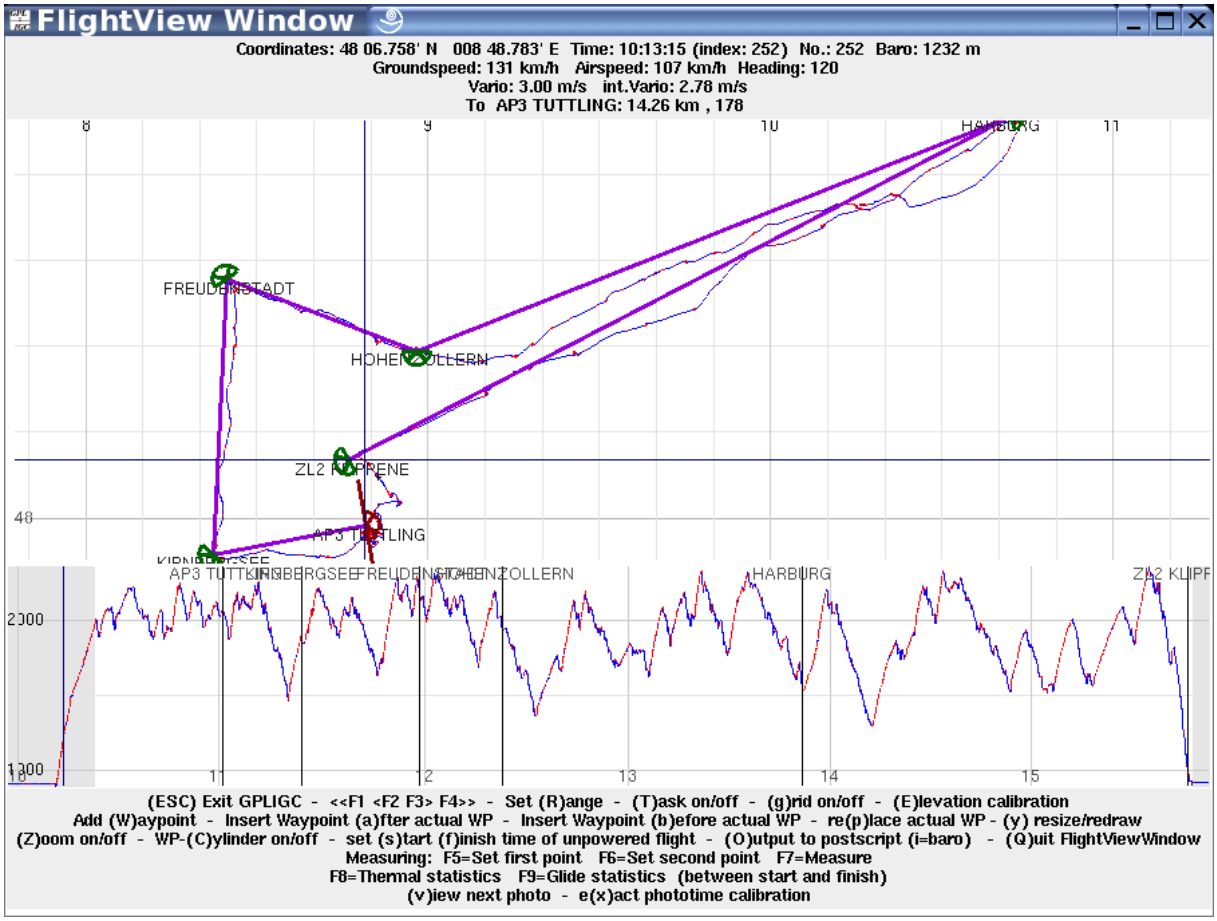


Figure 6: Crossing the starting line

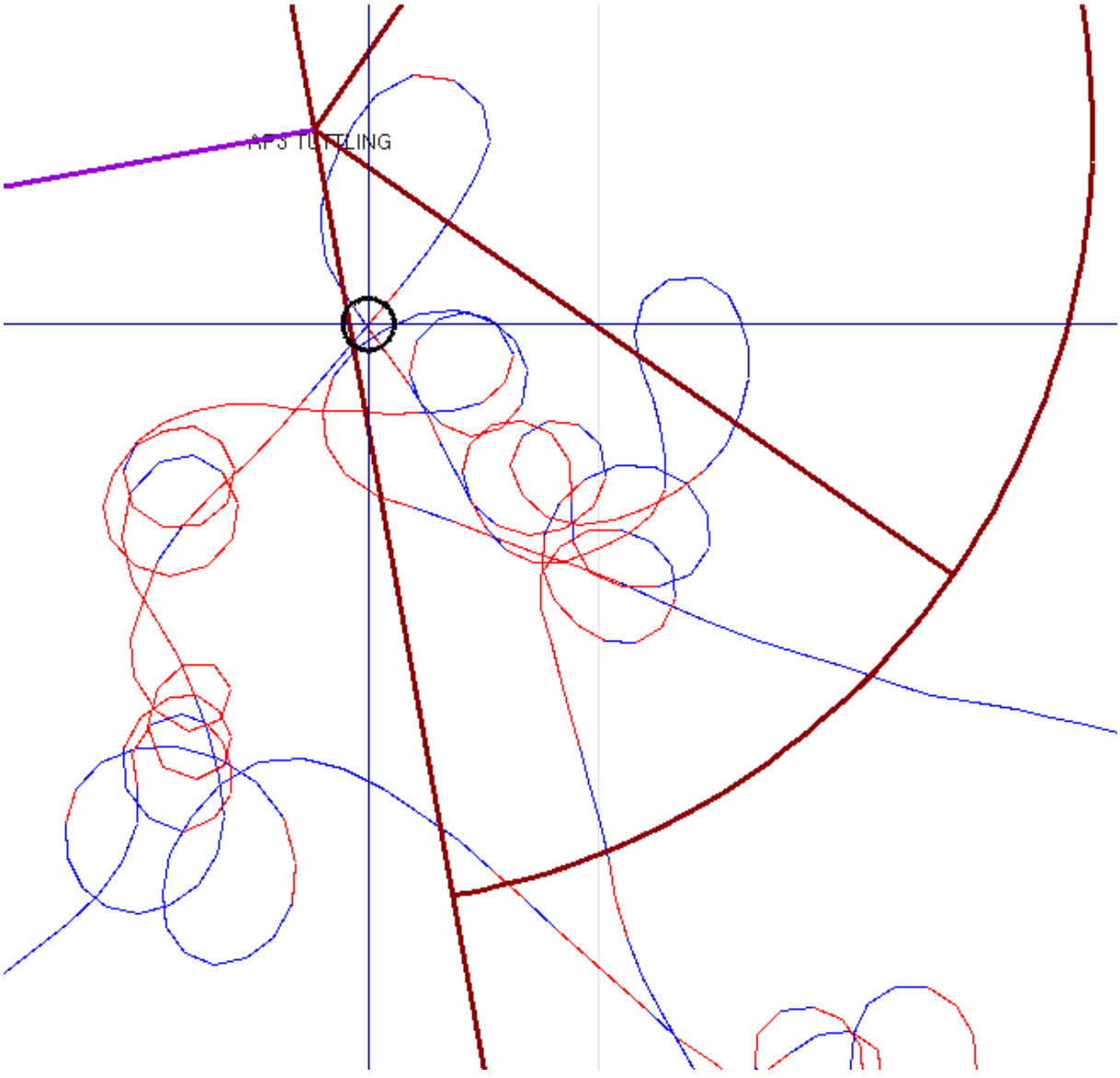


Figure 7: Reaching the finishing line

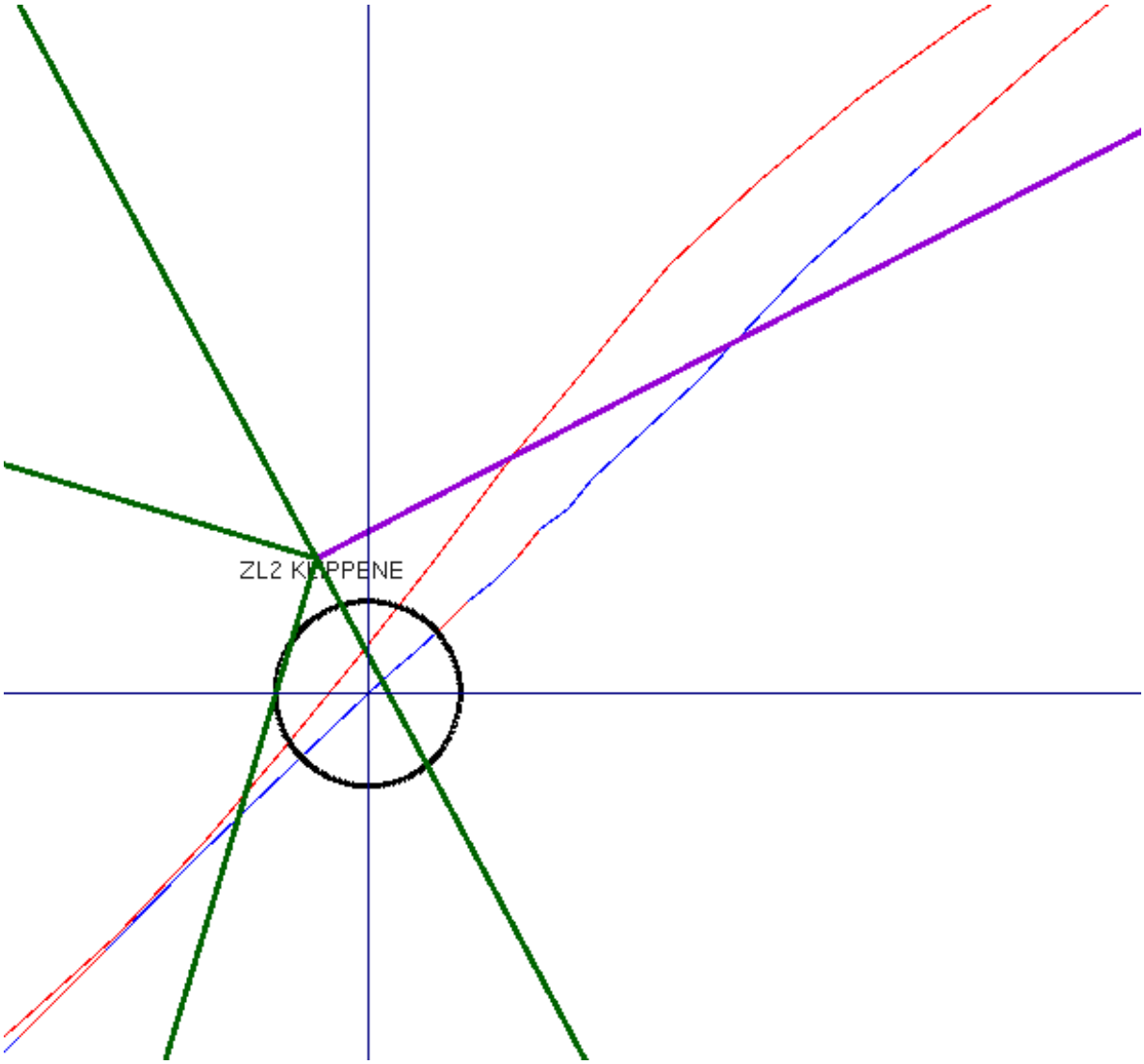


Figure 8: The GPLIGC flight statistic window

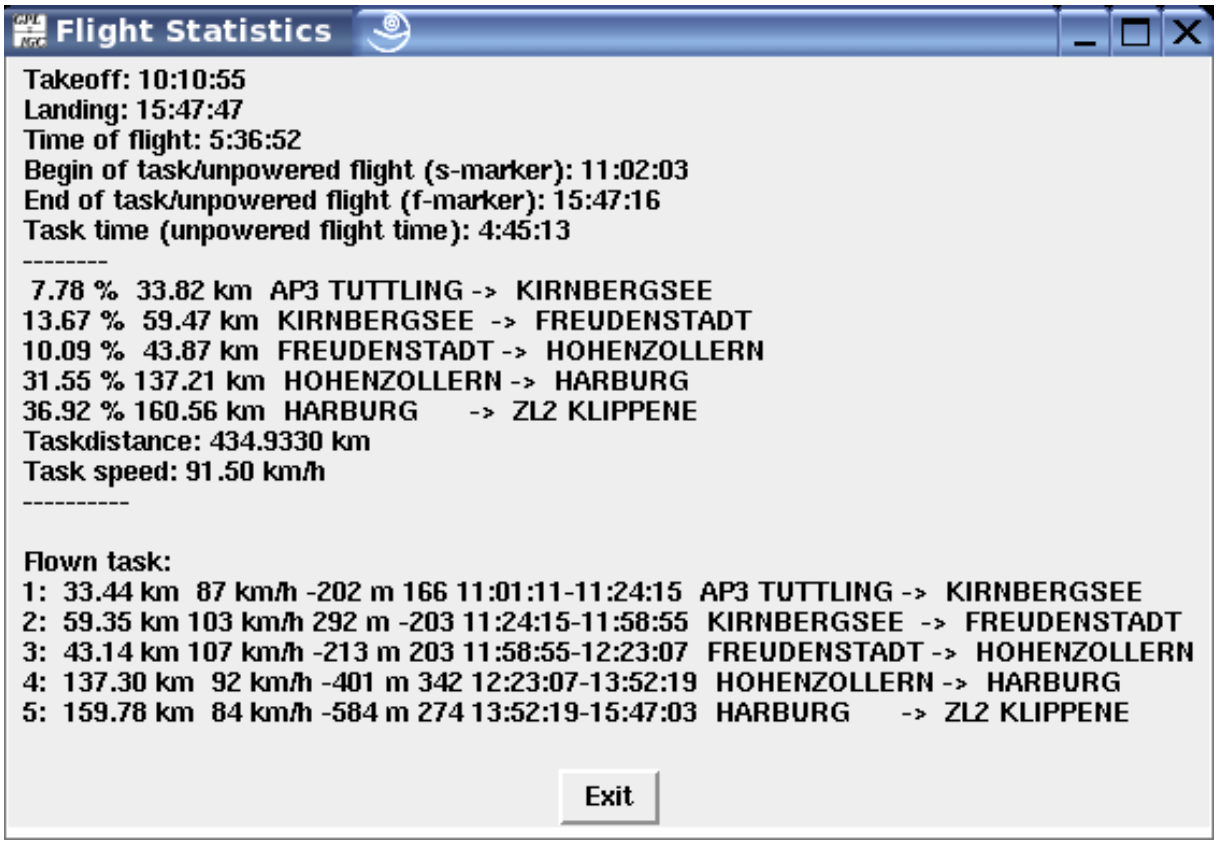


Figure 9: The GPLIGC thermal statistics

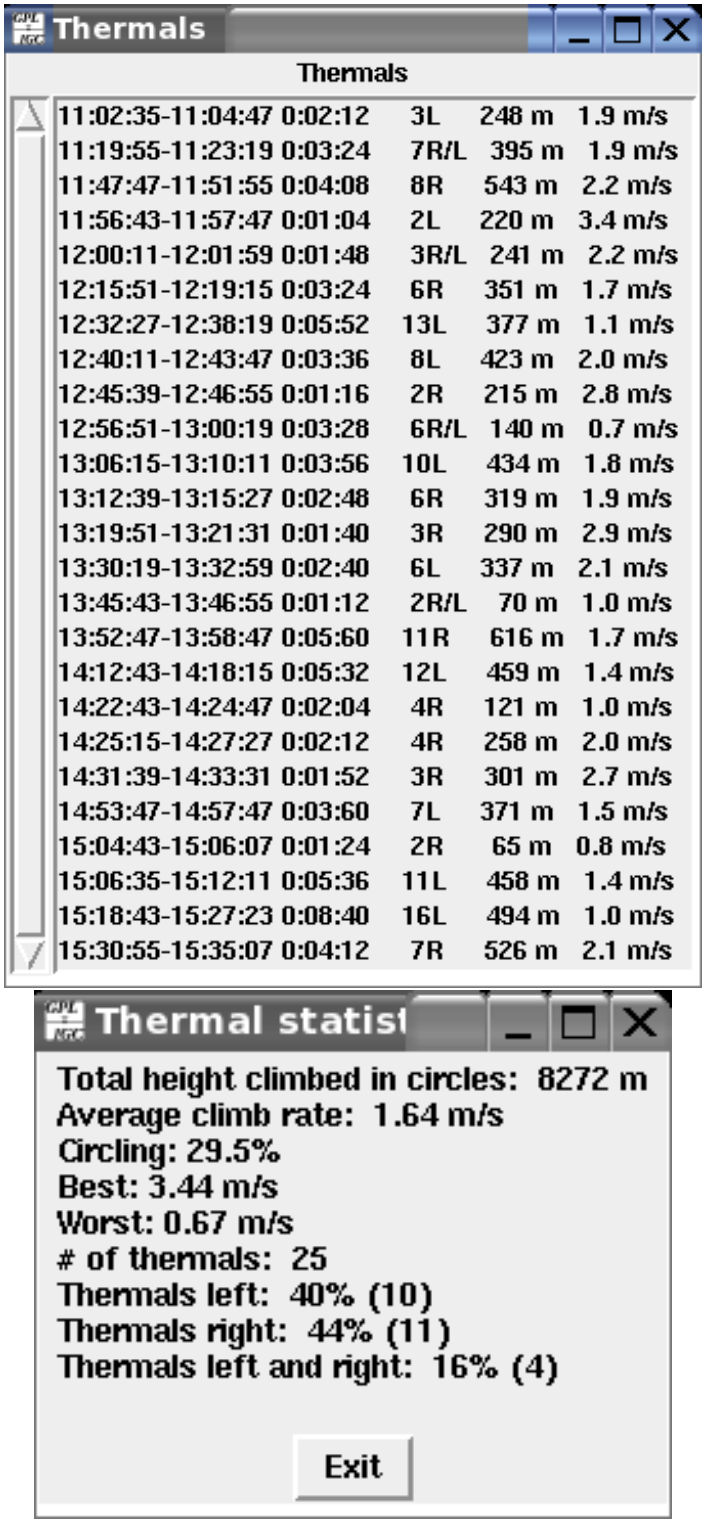
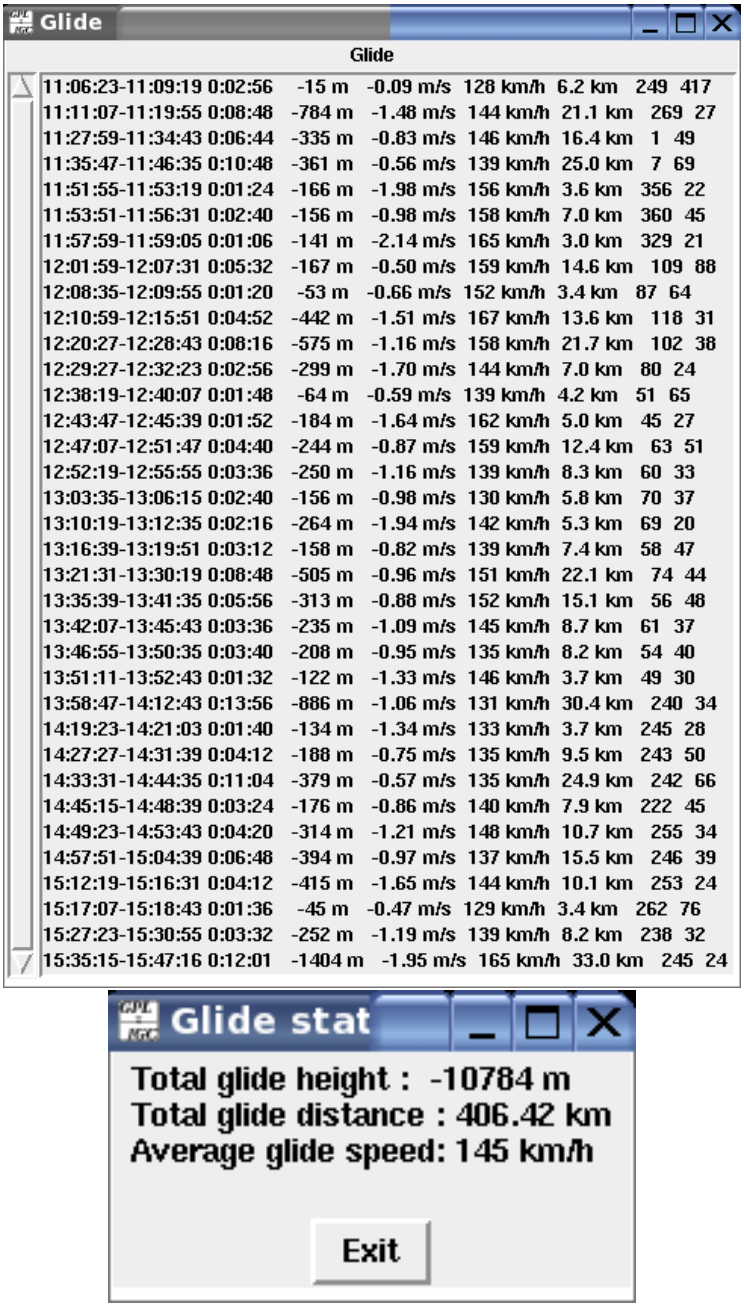


Figure 10: The GPLIGC glide statistics



6.2 Innsbruck Föhn flight 2009-04-16-GAR-000-02.igc

This nice example of a classical Innsbruck Föhn flight was done with our club DuoDiscus T in april 2009.

6.2.1 Flight Information (additional)

This tool allows us to enter/update some additional information. The date is shown correct, but the pilots name is missing. Plane and callsign can be entered too, the QHN can be set to 1006 (if I remember right). The timezone is +2, and the airfield is Innsbruck (LOWI).

6.2.2 Calibration of the altitude

The track was recorded with a Garmin Geko 301. Using the Flight View Window (fvw) the cursor can be set some point right before the take-off (e.g. 13:35). As can be seen here, the recorded altitude is quite close to 580m (which is the elevation of Innsbruck airport). The Geko's auto calibration function did a good job (it corrects the reference pressure with averaged GPS altitudes). Subsequently, the right elevation of Innsbruck LOWI (580 m) can be entered by pressing `e`. If the *save* button of the additional flight information tool is used, the calibration is saved and will be available later (after opening this file again). In this case the calibration is not really needed, since the offset is ca. three or four meters, only.

6.2.3 Winch

Using the fvw and the cursor (move with F1-F4 or with the mouse), the winch take-off can be found between 13:35:26 and 13:36:05. To analyse the winch launch mark the first point with F5, the second with F6. Statistics is then calculated with F7. The launch took 42 seconds, the gain in altitude was 343 m with an average climbing rate of 8.2 m/s. The heading was 81°, which is in good agreement with the runway 08. The speed of 91 km/h looks too small for a DuoDiscus, but the value reflects the projected groundspeed, not the airspeed. Change in altitude and the wind is not taken into account.

6.2.4 Ridge soaring

The lift at the ridge was entered at 13:36:38 and until 13:43:35 (within 7 min) 1700 m were gained in an average lift of 4.1 m/s with a nice maximum of ca. 8 m/s.

6.2.5 Wave

The first climb in the wave was found (after ATC clearence) at 13:56:20. In the following 10 min, ca. 1700m with an average of 2.8 m/s were climbed to the maximum of the ATC clearence. Subsequently airbrakes had to be used to limit the altitude. As you can see in the barogram I really enjoyed beeing on top (for more than one hour). A feet heating device helped to stay warm at about -20°C, and of course oxygen was used. Figure 11 shows the view to the wave clouds and the typical foehn gap above the Inn valley. The pictures are not included in the example, but the photo-locator option would look like Fig. 12, showing the positions of the pictures. Btw. Fig. 12 was created using the postscript output from fvw via the keys `o` and `i`.

6.2.6 Analysis of wind

To analyse the wind, which causes such nice lifts, we have to select parts of the track including many different directions flown. Circling works very well, but some figure-of-eight turns can be used too. 13:36:29–13:39:35 corresponds to the first lift at the ridge up to ca. 1500m. Selecting this range (F5/F6) and pressing F7 will analyse the wind. The opened gnuplot window, shows how the groundspeed depends on the heading (if airspeed is recorded – as e.g. in some Zander loggers – the difference groundspeed-airspeed is used). A sinus function is fitted to the data, which gives the direction and speed of the wind.

Figure 11: Foehn gap and massive wave clouds above the Inn valley. The picture was taken in ca. 5000 m, the top of the Ac Len is probably higher than FL200.

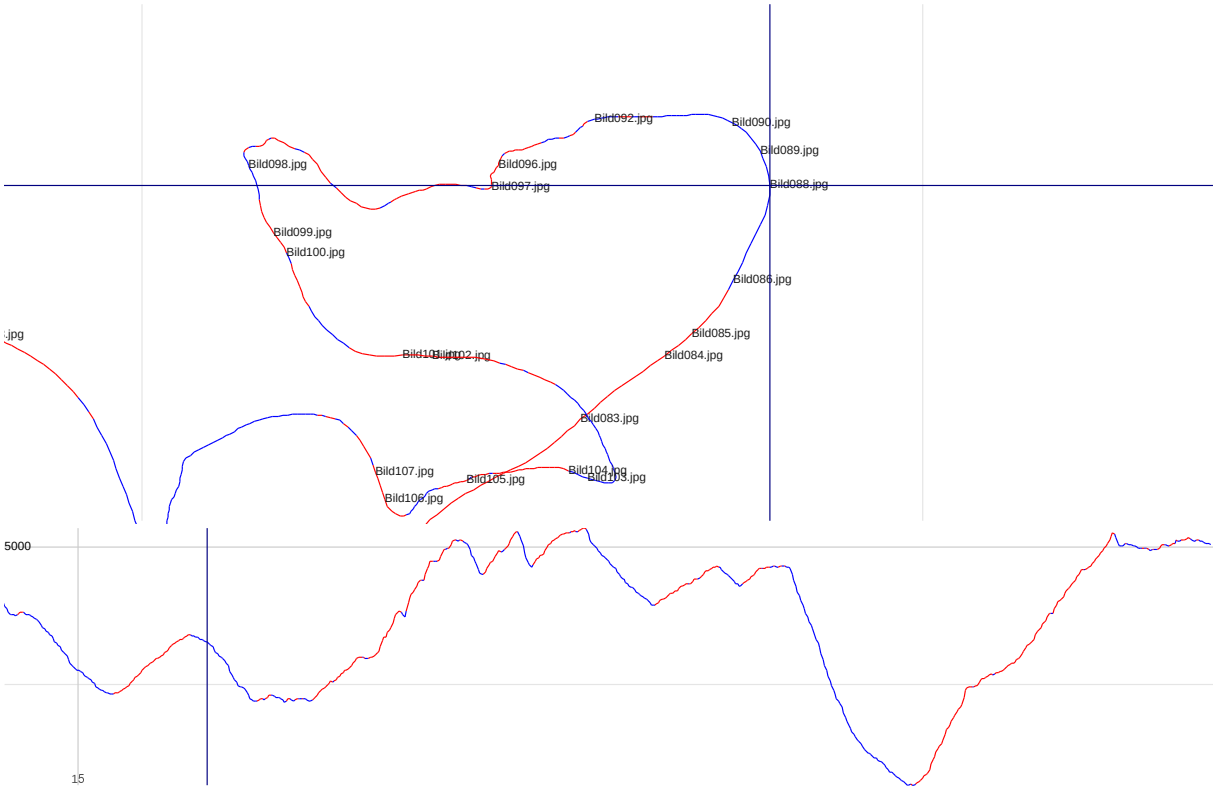
In this case its ca. 40km/h from 137°. The direction is typical for this part of the Inn valley in foehn situations.

Another interesting part is from 14:06:29 to 14:18:53, which includes some circles and figure-of-eight turns in the wave just north of Nockspitze and Axamer Lizum. The analysis shows a wind from 190° with 50km/h only, which is surprisingly low for such a strong wave. The wave north of the Glungezer (14:25:14–14:56:32) shows a bit higher wind speed of 60km/h and ca. 200°. The second ascend to the wave level (16:38:23–16:57:08) exhibits very similar wind conditions.

6.2.7 Oxygen debriefing

The *Flight Statistics* window shows some information on (recommended) oxygen consumption. It is calculated for constant open-flow systems. The values in brackets corresponds to the use of an Oxymizer canula (ca. 1/3 of a regular open-flow system). According to the FAR regulation 91.211 a total of 226 l of oxygen should have been used. Starting to use oxygen at FL100 (as recommended) would have increased the total to 275 l. Using an Oxymizer canula would have reduced these amounts to ca. 1/3. However,

Figure 12: Locating pictures



modern pulse-demand systems save even more oxygen, depending on your breathing rate.

6.3 using loopviewer.pl to make presentations of flights

loopviewer.pl is a simple script, which uses ogie to show flights automatically. This could be used for presentation of flights, e.g. at competitions. loopviewer.pl reads a list, which contains to entries (enclosed in ") per line: the first is the path and name of the igc-file, the second a comment, which should be shown with the flight. So far, this simple script needs manual editing, to obtain a reasonable set of parameters.

7 Tools

This is a small collection of Perl scripts, which can be useful for GPLIGC and OGIE users.

7.1 loopviewer.pl

The `loopviewer.pl` is a small script which can be used as a template to create an automatic show of a list of igc files. It is intended to be used at gliding competitions to have a nice presentation of all flights. All pilots can enjoy every flight of the day, while having a nice cold beer in the briefing hangar (video-beamer!). Therefore a list has to be created containing one line for every flight. Each line should contain the quoted path to the igc-file and a quoted information string:

```
"c:\path\to\file.igc" "1.  - Name - ASW20 - II - 610.34km - 102.3km/h - 1000pts"
"c:\path\to\file2.igc" "2.  - Name - Ventus2 - I2 - 610.34km - 101.7km/h - 980pts"
...
```

The configuration file should be set up to have some nice maps from the contest area, airspaces and whatever is needed. A good initial viewpoint position should be determined in some interactive run, subsequently the corresponding options can be changed in the `loopviewer.pl` script, which can be started with this simple call:

```
loopviewer.pl list
```

where `list` is the file containing the above mentioned list.

7.2 Garmin related tools

Since I bought a Garmin Geko301, a few tools have been developed to use the Garmin's track logs etc.

I recommend to use *gpspoint* by Thomas Schank to do the data transfers to and from your Garmin device. To handle the output from *gpspoint* the following tools can be used.

7.2.1 gpsp2igc.pl and gpsp2igcfile.pl

This tool was developed by my brother Max, and can convert the track log output from *gpspoint* (`gpspoint -dt >tracklog.gpsp`) to something like an igc-file to be opened by GPLIGC & OGIE.

I use gpspoint 2.030521:

```
gpspoint -p /dev/ttyS0 -dt | gpsp2igc.pl >out.igc
```

gpsp2igcfile.pl creates IGC-file(s) with a filename corresponding to the date(s) of the recording.

A The .gpligcrc configuration file

The GPLIGC configuration file. The keys and values in this configurationfile are *case-sensitive!*.

The following list contains the config-keywords with default values and (for some) a short description including valid alternative values.

- `DEBUG "0"`
  This should be set to "1" for debugging purposes.

- `ENL_noise_limit "500"`
- `altitude_unit_factor "1"`
- `altitude_unit_name "m"`
- `baro_grid_large "1000"`
  This changes the spacings of the gridlines in the barograph in the flight-view-window
- `baro_grid_small "500"`
- `baro_histo_intervall "50"`
  Interval for the altitude histograms in m
- `browser "/usr/local/mozilla/mozilla"`
  Select your favourite browser here.
- `coordinate_print_format "igch"`
- `cylinder_linewidth "3"`
- `distance_unit_factor "1"`
- `distance_unit_name "km"`
- `draw_task "1"`
  defines, whether the task is drawn by default in flight-view-window (1) or not (0)
- `draw_wpcyl "1"`
  defines, whether waypoint cylinders/sectors are drawn by default in flight-view-window (1) or not (0)
- `flight_view_window_grid "yes"`
- `garmin_download "sudo gpspoint -dt -p /dev/ttyUSB2"`
  this command is used to download tracks from a GPS device.
- `gnuplot_4_terminal "0"`
  In case of "1" an additional gnuplot shell will be started for each gnu-plot
- `gnuplot_draw_style "with lines"`
- `gnuplot_grid_state "set grid"`
- `gnuplot_major_version "4"`
- `gnuplot_terminal "x11"`
- `gnuplot_terminal_app "xterm -e"`
  The terminal application to be used for the gnuplot shell
- `gnuplot_win_exec "wgnuplot.exe"`
  Contains the filename of the gnuplot-binary on Windows platforms
- `gpsbabel_tdownload "sudo gpsbabel -t -i garmin -f /dev/ttyUSB2 -o igc -F "`
  contains the command string to be used to download trackdata via *gpsbabel* [4]. The last option should be -F, since the output file name will be appended to this string.

- `integrate_over "10"`
  Some of the data plots use integrated values. This will define how many data-points will be used for integration
- `libdir "/usr/local/share/gpligc/"`
  The path which will be used to load the library-files
- `marker_linewidth "3"`
- `mm_download_dirs "Aufnahmen Fotos Videoclips"`
  Directories, from where (multimedia) files will be copied (see mm_mountpoint too)
- `mm_mountpoint "/mnt/sdC"`
  Mountpoint for your multimedia recorder (e.g. mobile phone). Should be user mountable.
- `mm_player "mplayer"`
  Player to be used for multimedia files (audio recordings, movies, etc)
- `new_version_message_shown "0.1"`
- `open_additional_info "0"`
  If set to 1, the “additional info dialog” is opened immediately in cases where no gpi-file is found.
- `open_flight_view_window "1"`
  Whether the flight-view-window will be opened for each new igc file opened.
- `optimizer_cycles_mmc "20"`
  For all of the `optimizer_*` keys see section 4.3.4
- `optimizer_cycles_sa "5"`
- `optimizer_debug "0"`
- `optimizer_method "mmc"`
- `optimizer_mmc " -m 1000 -mmc 25000 -devisor 3 -refine 2 "`
- `optimizer_sa " -sima -m 1000 -sacycles 500 -saexp -sapara 15.0 -saparb 0.03 -devisor 3 -refine 2 "`
- `optimizer_verbose "0"`
- `photo_path "none"`
- `photos "1"`
- `picture_viewer "internal"`
  Whether to use the ”internal” or any other picture viewer. My favourite is ”kuickshow”
- `skip_check "1"`
  With ”1” a skip-check is performed. If the difference between to logged positions is larger than skip_limit_minutes the skip will be marked.
- `skip_del_first_after "1"`
  This circumvents a bug in the Garmin Geko tracklogs. The first position fix after a skip will be discarded.
- `skip_limit_minutes "0.2"`
  Limit to detect skips in the tracklog

- `speed_histo_intervall "5"`
  Interval for the speed histogram in km/h
- `speed_unit_factor "1"`
- `speed_unit_name "km/h"`
- `task_linewidth "3"`
- `terminal "xterm -hold -e"`
  terminal application to be used for some things (copying Multimedia files, downloading from Garmin)
- `te_vario_fallback "0"`
  The total energy compensated vario is usually calculated from the airspeed. Since many loggers dont log airspeed, the total energy compensation can be calculated from groundspeed (errors can be large in case of significant wind).
- `te_warning "1"`
  A warning on groundspeed total energy compensation can be disabled setting this to 0.
- `terminal "xterm -hold -e"`
  Terminal command used for download of garmin-tracks and media
- `timezone "0"`
  Offset to local timezone (e.g. timezone used in your camera) . Used for the photo locator.
- `vario_histo_intervall "0.5"`
  Interval for the vertical speed histograms in m/s
- `vertical_speed_unit_factor "1"`
- `vertical_speed_unit_name "m/s"`
- `viewclick_res "1"`
  If you experience serious delays in moving the crossmarks in the fvw by clicking close to the track, you may increase this number to a larger integer
- `waypoint_linewidth "3"`
- `wind_analysis "1"`
  If set to "1", a airspeed-groundspeed difference is plottet with F5/F5/F7 statistics.
- `working_directory "/home/kruegerh/IGC"`
- `zoom_sidelength "10"`
  Sidelength of the zoom-window in km
- `zylinder_names "1"`
- `zylinder_radius "500m"`
- `zylinder_wp_type "both"`

B OGIE keyboard control

The keys are case sensitive!

Movement

a	fast backward
s	backward
g	forward
space	fast forward
d	left
f	right
t	up
z	down
q,esc	escape
left,right	rotation around marker
up,down	rotation around marker
shift-left, shift-right	rotation around center
shift-up, shift-down	rotation around center

Viewing Modes

l	terrain on/off (build up from DEM data)
L	wireframe mode on/off
b	maps on/off (if texture maps specified in config-file)
x,c	switch to previous/next map-set
w	fullscreen on/off
o	switch 2D-ortho/3D-perspective view mode
h	curtain on/off
B	toggle Grayscale/Color-mode
1	max's special colormap
2	atlas colormap
3	single color (gray-ramp)
4	dark green → yellow → red
5	magenta → dark blue → turquois
6	black → rainbow → white
F10,F11	decrease, increase number of lower colormap
shift-F5, shift-F6	switch flight linestrip mode: up/down, altitude, speed, vario
shift-F7, shift-F8	decrease, increase colormap applied to flight linestrip
shift-F3, shift-F4	decreas, increase Fligh linestrip width (1-5)
O	flat shading on/off
j	fog on/off
9,0	decrease, increase fog-density
7,8	decrease, increase field-of-view-angle
+,-	increase, decrease z-axis scaling
F5	Reset Viewpoint to initial position
F6	Info on/off
M	Lock position relative to marker (follow-mode)

U	Display a part of the flight only on/off
I	Move Marker automatically forward... (Movie-Mode)
J	Turn joystick on/off
shift-F1, shift-F2	decrease, increase framerepeat in moviemode. Use to slow down animation.
F8	Modulate (texture maps with coloured surface elevation) on/off
F9	Airspace on/off
shift-F9	Airspace wireframe mode on/off

Marker

F1	marker fast backward
F2	marker backward
F3	marker forward
F4	marker fast forward
F7	Marker on/off
page-up	increase size of marker
page-down	decrease size of marker

Other Options

m	mouse control on/off
p	screenshot (jpeg's are written as frameNUMBER.jpg) with NUMBER starts at 0000 each session.
P	continous screenshots (each rendered frame will be saved as jpeg).
y	texture map compression on/off (on supported systems)
u,i	increase, decrease elevation offset (shifts flight against terrain)

Lifts

shift-page-up	increases text size
shift-page-down	decreases text size
pos1	switches info up
end	switches info down

Waypoints

F12	showing waypoints on/off
shift-page-up	increases text size
shift-page-down	decreases text size
shift-pos1	switches info up
shift-end	switches info down

Stereoscopic View modes

S	stereoscopic double-image
D	Red-Green stereoscopic
F	Red-Blue stereoscopic

A	swap images (right/left)
Q	decrease eye distance by 50m
W	increase eye distance by 50m

C Commandline options (OGIE)

OGIE has a lot of commandline options. For a full list try:
`ogie --help`
Commandline arguments override configuration file settings, which override compiled-in-defaults. For most options there is an option to turn the feature on, and another option to turn it off, so you are able to override all possible configuration file settings. But most of the features can be changed at runtime too.

Many of the options can be changed at runtime via keyboard input or menus, but some are *start-time-options*, which can be activated or set at the time the program is started, only. These are marked (ST).

Options marked as *default* are active by default (compiled in). This may be overridden by the configuration-file.

Some of the commandline-switches (or commandline-options) require additional parameters (`INT`, `FLOAT`, `FILE` or `STRING`), the meaning of these can be read in section D.

C.1 All available commandline options

- `-h, --help`
  Print help and all available commandline options (ST)
- `-V, --version`
  Print version information (ST)
- `-v, --verbose`
  Be verbose, lots of console output (ST)
- `--quiet`
  Turn off verbosity no output to console. Overrides –verbose and VERBOSE (ST)
- `-q, --query-gl`
  Querying openGL implementation. This will give out some information about your specific OpenGL implementation (ST)
- `--check`
  This returns exitcode 0. Used by GPLIGC to check if OGIE is available (ST)
- `--debug`
  This overrides –verbose and –quiet. Lots of ugly debugging output (ST).
- `--compiler`
  This will give out some information about compiler and building environment (ST)
- `-i FILE, --igc-file=FILE`
  This option specifies the igc-file to be opened (ST)
- `--gpsd`
  Try to connect to local gpsd, retrieve position data, and start in live-mode (ST). Ogie has to be compiled with gpsd support (See section 3.1).

- `-g, --gpsalt`
  The altitude from GPS will be used instead of barometric (ST)
- `-b, --baroalt`
  Barometric altitude will be used (default, ST)
- `--use-all-fixes`
  Use all position fixes. Even those which are flagged invalid. (ST)
- `--lat=FLOAT`
  Latitude of centre. Used for terrain viewing without igc-file (ST)
- `--lon=FLOAT`
  Longitude of centre. Used for terrain viewing without igc-file (ST)
- `--get-elevation`
  To be used with –lat and –lon. Will return the elevation of the given coordinates, if a DEM is configured. To be used with SRTM-3 data. (elevation=0 is the void-flag, at least in the usgs seamless server downloads). INVALIDn is returned, if n neighbouring grid-points are invalid. n=9 is returned, if the requested position is not covered by the configured DEM. The second value which is returned is the max difference in elevation of the four neighbouring grid-points, the maximum of the remaining neighbours for INVALID1-3, 0 for INVALID4 and 9999 for INVALID9. (ST)
- `-c FILE, --config-file=FILE`
  Used to open a non-standard config file (ST)
- `-o, --ortho`
  Forces startup in 2D orthographic viewing mode
- `--perspective`
  Forces startup in 3D viewing mode
- `--aov=INT`
  This sets the angle of view (1-179)
- `-l, --landscape`
  Use digital elevation data to display terrain
- `-f, --flat`
  Don't use terrain from DEM. Use flat surface instead
- `-m, --map`
  Activates displaying of digitised maps. If configured in configuration file.
- `--no-map`
  Don't use digitised maps
- `--map-set-name=STRING`
  Name of map set to use as default
- `--modulate`
  Maps are coloured by DEM altitude colour, if this option is active
- `--no-modulate`
  Use original colour of maps

- `--maps-unlighted`
  If maps used with DEM and `--no-modulate`, this turns off lighting of the maps (ST)
- `--maps-lighted`
  Maps are lighted, if used with DEM and no modulation. (Default behaviour) (ST)
- `--no-lighting`
  Don't use lighting. Use for orthomode with upscaling recommended (ST)
- `--terrain-shading`
  Terrain shading. This implies the `--no-lighting` option (ST)
- `--shading-scale=FLOAT`
  Strength of terrain shading. The smaller the value, the stronger the effect. If not given, the max elevation difference divided by seven is used (ST)
- `--light-direction=INT`
  The direction of the light for terrain shading. 1 corresponds to north, 2 nort-east, 3 east, 4 south-east, 5 south, 6 south-west, 7 west, 8 north-west (ST)
- `-a, --airspace`
  Turn on airspace visualisation
- `--no-airspace`
  Turn off airspace visualisation
- `--airspace-wire`
  Turn on airspaces in wireframe mode
- `--airspace-wire-col-[r|g|b]`
  Defines colours for airspace wireframe lines
- `--airspace-wire-width`
  Sets the linewidth used to draw airspace wireframes.
- `--airspace-transparent`
  Turn on transparent airspace.
- `--airspace-limit=INT`
  Airspaces, which lower boundary is higher that this limit (in FL), will not be shown. (ST)
- `--airspace-file=FILE`
  Use airspaces from file (OpenAir^{TM}-format) (ST)
- `-w, --wire`
  Draw terrain surface as wireframe-model
- `--filled`
  Use filled polygons for terrain (default)
- `--grayscale`
  Use gray scaled image
- `--color`
  Use coloured image (default)
- `--stereo`
  Use double image stereoscopic mode

- `--no-stereo`
  Do not use stereoscopic modes (default)
- `--stereo-rg`
  Use anaglyphic stereoscopic mode (red/green)
- `--no-stereo-rg`
  Do not use anaglyphic stereoscopic mode red/green (default)
- `--stereo-rb`
  Use anaglyphic stereoscopic red/blue mode
- `--no-stereo-rb`
  Do not use anaglyphic stereoscopic red/blue (default)
- `--stereo-hw`
  Use stereoscopic hardware if available (ST)
- `--no-stereo-hw`
  Do not use stereoscopic hardware (default)
- `--inverse-stereo`
  Swap right/left image for stereoscopic modes
- `--no-inverse-stereo`
  Don't swap images right/left (default)
- `--eye-dist=FLOAT`
  Set eye distance for stereoscopic viewing modes (default=0.2km)
- `--flat-shading`
  Do not use gouraud shading (every triangle of the surface will get the same colour)
- `--gouraud-shading`
  Use gouraud-shading (default)
- `--quads`
  Use quadrilaterals to build terrain surface (ST)
- `--curtain`
  Draw curtain (default)
- `--no-curtain`
  Do not draw curtain
- `--haze`
  Enable atmospheric haze
- `--no-haze`
  Do not use atmospheric haze (default)
- `--haze-density=FLOAT`
  haze density (0.0 clear - 0.5 dense fog)
- `--colormap=INT`
  Use colourmap `INT` for terrain surface (see 5.8)

- `--colormap-sea=INT`
  Use colourmap `INT` for seafloor (see 5.8)
- `--colormap-min=INT`
  Minimum altitude for colour scale (ST)
- `--colormap-max=INT`
  Maximum altitude for colour scale (ST)
- `--sealevel=INT`
  Elevation of sealevel. Beneath this elevation seafloor colourmap is used, above terrain colourmap (ST)
- `--sealevel2=INT`
  Elevation of sealevel2. The blue ocean will be drawn at elevation of sealevel2 (ST)
- `--sealevel3=INT`
  Elevation of sealevel3. A transparent blue surface will be drawn at elevation of sealevel3 (ST)
- `--ignore-elev-[min,max]=INT`
  Defines limits or a range, which are not used for determining the extrem values of the topography (ST)
- `-s FLOAT, --scalez=FLOAT`
  Z-axis scaling. A factor of 1.0 represents the *real* relations. A default value of 3.0 is used to emphasise altitude
- `-d INT, --downscaling=INT`
  DEM raster downscaling can be used to reduce resolution of surface (to show larger areas) (ST)
- `--upscaling INT` The resolution of the DEM raster is enhanced by interpolation. Use with care, higher factors increase the number of triangles used dramatically. Good for small area terrain display (ST)
- `--fullscreen`
  Start up in fullscreen mode
- `--window`
  Start windowed (not fullscreen, default)
- `--width=INT`
  Set initial width of window (pixels)
- `--height=INT`
  Set initial height of window (pixels)
- `--border=FLOAT`
  Adds a `FLOAT` km border at top, bottom, right and left margin (ST)
- `--border-lat=FLOAT`
  Adds a `FLOAT` km border at top and bottom margin (ST)
- `--border-lon=FLOAT`
  Adds a `FLOAT` km border at right and left margin (ST)
- `--offset=INT`
  Shifts the flight `INT` meters up (relative to the dem surface)

- `-e INT, --airfield-elevation=INT`
  Sets the elevation of the take-off location (in m). The relative shift of the flight will be calculated automatically
- `--marker-pos=INT`
  Set the position of the marker to datapoint number `INT`
- `--marker-time=string`
  Set the position of the marker to datapoint nearest to HH:MM:SS
- `--marker`
  Activated the marker at start-time
- `--marker-size=FLOAT`
  Size of the Marker (0.01-10)
- `--no-marker`
  Disables marker at start-time (default)
- `--info`
  Activates info text display at start-time
- `--no-info`
  Turns the info text display off (default)
- `--text="STRING"`
  With this option a text string can be specified, which will be displayed in the first line of the info text (ST)
- `--no-position-info`
  This option removes the information about the viewpoint position.
- `--no-marker-pos-info`
  To turn off the information about the marker position use this option.
- `--text-size=FLOAT`
  Size of text for points/lifts (0.001-1.0)
- `--text-width=FLOAT`
  Width of text (1-20)
- `--lifts=STRING`
  GPLIGC liftsfile (ST)
- `--lifts-info-mode=INT`
  which info to display (1= int. vertical speed, 2=verical speed, 3=altitude, 4=time, 5=time, 6=date, 7=file)
- `--waypoints-file=STRING`
  set the waypointsfile (see 3.11). Overrides any names given in the config file
- `--waypoints`
  show waypoints (default=off)
- `--no-waypoints`
  dont show waypoints (default=off)

- `--waypoints-info-mode=INT`
  sets info to display (1-4: 1=description, 2=name, 3=altitude [m], 4=symbol).
- `--flighttrack-mode=INT`
  Sets the mode of track display. The interger value can be one of 0,1,2 or 3. 0: Classic mode, two colours (climbing, descending), colours can be changed with `FLIGHTSTRIPCOL[UP|DOWN]_[R|G|B]`. 1: Colour gradient (see `--flighttrack-colormap`) is used to display altitude (this is the default). 2: Colour gradient is used to show speed. 3: Colour ramp is used to show vertical speed.
- `--flighttrack-colormap=INT`
  Sets the colourmap used to display the flighttrack. Integers from 1 to 7 can be used (see 5.8).
- `--flighttrack-linewidth=FLOAT`
  Sets the linewidth of the flighttrack. Floating point values in the range of 1.0 – 5.0 can be used.
- `--follow`
  The viewpoint will be coupled with marker (default)
- `--no-follow`
  Makes viewpoint independent of marker position
- `--marker-range`
  A range (in time; future-past) around marker is plotted only
- `--no-marker-range`
  Full flight data is displayed (default)
- `--marker-ahead=INT`
  Defines marker range (datapoints in future of marker position) (ST)
- `--marker-back=INT`
  Defines marker range (datapoints in the past of marker position) (ST)
- `--movie`
  This will start up ogie in movie mode.
- `--cycles=INT`
  If given, ogie will perform INT cycles in movie mode before exiting (ST)
- `--spinning=float`
  Whether ogie should do spinning in movie mode. float is an angular value (in degrees), its sign determines the direction of spinning (ST)
- `--smooth-mouse`
  Mouse movement will be damped (ST)
- `--parent-pid=INT`
  PID of parent process. To this PID the signal SIGUSR1 will be send on exit
- `--compression`
  Use texture map compression
- `--no-compression`
  Do not use texture map compression (default)
- `--offscreen`
  Render a single image and output to jpeg. Offscreen rendering with GLX

- `--osmesa`
  Render a single image and output to jpeg. Offscreen with Mesa
- `--os-outfile=FILE`
  Sets filename for offscreen rendered jpeg-image (ST)
- `--jpeg-quality=INT`
  Sets jpeg Quality (compression level, 0-100) of jpeg output (ST)
- `--image-format=STRING`
  Sets output format for screenshots. Available jpg, rgb (ST)
- `--save-path=STRING`
  Via this option location for saving screenshots can be given. (ST)
- `--basename=STRING`
  The basename of screenshots can be defined. A number and file extension will be added automatically (ST)
- `--clipping-far=FLOAT`
  (ST)
- `--clipping-near=FLOAT`
  Distance of the clipping planes in km. Default is 0.2 or 1.0 (near) and 600 or 1000 (far). Change this if you need. This influences depth-buffer accuracy! (ST)
- `--init-lat=FLOAT`
  Sets latitude of initial viewpoint position (ST)
- `--init-lon=FLOAT`
  Sets longitude of initial viewpoint position (ST)
- `--init-alt=INT`
  Sets altitude of initial viewpoint position (ST)
- `--init-heading=INT`
  Sets initial viewing direction (heading in degree) (ST)
- `--init-dive=INT`
  Sets initial viewing dive angle (degrees downwards from horizontal) (ST)
- `--init-pos-N`
  Sets initial position above north border of scene (ST)
- `--init-pos-E`
  Sets initial position above east border of scene (ST)
- `--init-pos-S`
  Sets initial position above south border of scene (default) (ST)
- `--init-pos-W`
  Sets initial position above west border of scene (ST)
- `--init-pos-NE`
  Sets initial position above north-east corner of scene (ST)
- `--init-pos-SE`
  Sets initial position above south-east corner of scene (ST)

- `--init-pos-SW`
  Sets initial position above south-west corner of scene (ST)
- `--init-pos-NW`
  Sets initial position above north-west corner of scene (ST)
- `--init-pos-center`
  Sets initial position in the centre of the scene (ST)
- `--init-ortho-lat=FLOAT`
  Sets initial latitude of orthographic viewing mode centre (ST)
- `--init-ortho-lon=FLOAT`
  Sets initial longitude of orthographic viewing mode centre (ST)
- `--init-ortho-width=FLOAT`
  Sets initial width of orthographic-viewing [km] (ST)
- `--projection-cyl-platt`
  Sets "platt projection" (ST)
- `--projection-cyl-no1`
  Sets cylindric projection 1 (ST)
- `--projection-pseudo-cyl-no1`
  Sets pseudocylindric projection 1 (ST, default)
- `--projection-cyl-mercator`
  Sets Mercator projection (ST)

D Configuration file (.ogierc)

The `.ogierc` has to be used to set up:

1. use of a digital elevation model (see 3.8)
2. use of digital raster maps (see 3.9)
3. use of OpenAir™ airspace file (see 3.10)
4. change the default behaviour of *OGIE*

On startup of the program *OGIE* will load a configuration file `.ogierc` from your `HOME` directory. On Windows platforms the file is called `ogie.ini` and is read from the GPLIGC-installation directory. To use a configuration file from a different location, you can specify the filename by the command line option `--config-file FILENAME`.

The `.ogierc` configuration file may contain keywords-values pairs, with which the default behaviour can be changed. All keywords *are not* case sensitive. In the following list of the allowed keywords, placeholders are used to represent possible values:

- `bool` means that one of `true,on,yes,1,false,off,no,0` (case-insensitive) can be used to turn that option on or off.
- `float` stands for a floating point value (such as 54.734 or 0.3)
- `integer` should be a integer value (such as 1 or 42)

- `file` is to be substituted by a filename with full absolute path (e.g. /usr/share/gpligc/data/dem/demdata.dat or c:
  GPLIGC
  data
  dem
  demdata.dat)
- `string` can be any character string (without whitespaces)

D.1 Keywords

The list of keywords is sorted alphabetically. If you misspell keywords, you will get a warning in the output. Watch the output after changing your configuration file.

- `AIRSPACE bool`
  Used to set whether airspaces should be displayed by default or not. To define the airspace-describtion file see `OPEN_AIR_FILE`
- `AIRSPACE_x bool`
  With x one of D, C, CRT, Q, R, P. Used to set whether airspace type x should be displayed by default or not
- `AIRSPACE_LIMIT integer`
  Upper limit (in FL) for Airspaces. Airspaces with bottom altitude higher than limit, are not shown.
- `AIRSPACE_WIRE bool`
  This changes the default mode for airspace (wireframe or transparent)
- `ALT_UNIT_FAC float`
  Factor to convert from meters to another unit (e.g. 3.28 for feet)
- `ALT_UNIT_NAME string`
  Name of the alternative altitude unit
- `ANGLE_OF_VIEW integer`
  Angle of view can be a value between 1 and 179 (default=80)
- `AUTOREDUCE bool`
  If the resolution of the requested DEM exceeds the `MAXTRIANGLES` limit, the upscaling (`--upscaling`) is reduced and the downscaling (`--downscaling`) is increased until the limit isn't exceeded anymore.
- `BIGENDIAN bool`
  Digital elevation data may be present as big endian (most significant byte first, MSB, motorola bytee order) or little endian (least significant byte first, LSB, intel byte order). The default is yes - big endian, which applies to GTOPO30, SRTM30, SRTM-3 (.HGT), but *not* to GLOBE and SRTM-3 from seamless-server.
- `BASENAME string`
  The basic filename used to save screenshots can be defined here. It defaults to *frame*.
- `BACKGROUND_COLOR_[1|2]_[R|G|B] float`
  Two background colours can be set. The red, green and blue-value can be set seperately for colour 1 and 2. The range of the floating point values are limited from 0 to 1.

- `BACKGROUND_STYLE integer`
  Three background styles are available. 1 correspond to the old style with one solid background (colour 1 is used). A value of 2 will set the background to a vertical gradient from colour 1 (top) to colour 2. The value 3 will switch to a horizontal gradient (colour 1 is left).
- `BORDER float`
  Width of border in km to be added around the terrain (default=5)
- `COLORMAP integer`
  Colourmap to be used for terrain. See 5.8
- `COLORMAP_SEA integer`
  Colourmap to be used for regions below sea-level. See 5.8
- `COMPRESSION bool`
  Whether texture map compression should be used or not (default=no). To use this, the openGL implementation has to support texture map compression (e.g. Mesa does not)
- `CURTAIN bool`
  Whether the blue "curtain" should be drawn or not (default=yes)
- `DEM_COLUMNS integer`
  Number of columns in digital elevation file
- `DEM_FILE file`
  Name and path to digital elevation file
- `DEM_GRID_LAT float`
  Step width in latitude of digital elevation file
- `DEM_GRID_LON float`
  Step width in longitude of digital elevation file
- `DEM_INPUT_FACTOR float`
  Scaling factor for DEM data. Should be set in a way, that the result is in meters (e.g. 0.30488 for feet)
- `DEM_LAT_MAX float`
  Maximum of latitude in digital elevation file
- `DEM_LAT_MIN float`
  Minimum of latitude in digital elevation file
- `DEM_LON_MAX float`
  Maximum of longitude in digital elevation file
- `DEM_LON_MIN float`
  Minimum of longitude in digital elevation file
- `DEM_ROWS intger`
  Number of rows in digital elevation file
- `EYE_DIST float`
  Distance between the "eyes" in stereo viewing modes (in km)

- `FGLUT_CHECK bool`
  If true, a check for the freeglut version is done. Should be enabled if you use freeglut (to use some nice freeglut things in future versions). If you use the classic glut, you should left this to the default (=off) to avoid a harmless warning message.
- `FLIGHTSTRIPCOL[UP|DOWN]_[R|G|B]`
  Sets one of the colour components Red Green or Blue for the classic flightstrip mode (0). If you want to have the flightstrip displayed in a single colour, set the colours for *up* and *down* to the same values.
- `FLIGHTSTRIP_LINEWIDTH 2.0` Set the width of the lines displaying the GPS-track (floating point value in a range of 1.0 – 7.0).
- `FLIGHTSTRIP_MODE` Change the default mode for displaying the GPS-track (default=1, 0=classic, 1=altitude, 2=speed, 3=vertical speed).
- `FLIGHTSTRIP_COLORMAP` Set the default colourmap-type used to display the flight track (integer value, see 5.8)
- `FOLLOW bool`
  While the "follow-mode" is active, the viewpoint will follow the marker
- `FULLSCREEN bool`
  Whether the OGIE should startup in full-screen mode
- `GPSALT bool`
  Whether the GPS-altitude should be used instead of the barometric altitude
- `GRAYSCALE bool`
  Gray-scale (Black/White) mode (default=off)
- `HAZE bool`
  Atmospheric haze
- `HAZE_DENSITY float`
  Density of atmospheric haze (0.0 - 0.5) (default=0.01)
- `IMAGE_FORMAT string`
  This option sets the output format for screenshots. Available options are:
  `jpg` jpeg format
  `rgb` headerless rgb format (without compression, you need to know width and height to use this image later)
- `INFO bool`
  Info display (shows information about viewpoint and marker-position)
- `INFOFONT_SIZE integer` [20-100]
  The default is 40.
- `INFOFONT_LINEWIDTH float` [0.5-3.0]
  The default is 1
- `INFO_STYLE integer` [1—2]
  1= new style (thanks to Antonio Ospite), 2=old style
- `INVERSE_STEREO bool`
  Swap right/left image in stereo-modes

- `JOYSTICK_FACTOR_X float`
  Scaling factor for joystick-input-value. X-Axis (left-right). Negative values will invert movement. (Default=0.01)
- `JOYSTICK_FACTOR_Y float`
  Scaling factor for joystick-input-value. Y-Axis (forward-backward). Negative values will invert movement. (Default=0.01)
- `JOYSTICK_FACTOR_Z float`
  Scaling factor for joystick-input-value. Z-Axis (up-down). Negative values will invert movement. (Default=0.01)
- `JPEG_QUALITY int`
  Sets the quality of the jpeg (0-100, default=75)
- `LANDSCAPE bool`
  Whether terrain should be displayed by default (if digital elevation model is available)
- `LIFTS_COLOR_[R|G|B] float`
  If you don't like the default colour of the lifts, you can change it with these keywords.
- `MAP bool`
  Whether textured maps should be displayed (if available)
- `MAP_BOTTOM float`
  Latitude of lower map border
- `MAP_CUT`
  Used to separate map-sets
- `MAP_FILE file`
  Filename of a map-tile (jpeg)
- `MAP_HEIGHT integer`
  Pixel height of map-tile (not necessary for jpeg)
- `MAP_LEFT float`
  Longitude of left map border
- `MAP_RIGHT float`
  Longitude of right map border
- `MAP_SET_NAME string`
  Name (identifier) for the map-set
- `MAP_SHIFT_LAT float`
  All following map tiles will be shifted in latitude by this value
- `MAP_SHIFT_LON float`
  All following map tiles will be shifted in longitude by this value
- `MAP_TOP float`
  Latitude of upper map border
- `MAP_WIDTH integer`
  Pixel width of map tile (not needed for jpeg)

- `MAPS_UNLIGHTED bool` With this set to true, maps will not be lighted when used with DEM and no modulation. (Default is false)
- `MARKER bool`
  Whether the marker should be active by default
- `MARKER_AHEAD integer`
  How many data points will be displayed (ahead from marker) in marker-range mode
- `MARKER_BACK integer`
  How many data points will be displayed (backwards from marker) in marker-range mode
- `MARKER_RANGE bool`
  Marker range mode default
- `MARKER_SIZE float`
  Marker size (default=1, range=0.01 to 10.0)
- `MARKERCOL_[R|G|B] float`
  If you don't like the default (red) colour of the maker, you can change it using these keywords.
- `MAXTRIANGLES float`
  The value of maximal allowed triangles for the terrain. If this is exceeded, the terrain is turned off. (default=1.5e6). To be used in online-plotter applications to avoid Denial-of-Service attacks. Should be set to a value, which your server can handle in a reasonable time.
- `MODULATE bool`
  Whether the maps should be coloured by digital elevation model elevation colour scaling
- `MOUSE bool`
  Whether the mouse-pointer is visible or not
- `MOVIE bool`
  If set to true, this will startup ogie in movie mode
- `MOVIE_TIMER integer` (deprecated)
  Time delay in milliseconds for movie-mode. This reduces the responsiveness of ogie. Better use the next three options.
- `MOVIE_REPEAT bool`
  Enables multiple rendering of each frame to slow down the marker movement
- `MOVIE_REPEAT_FACTOR int`
  Defines how often a frame should be rendered in `MOVIE_REPEAT` mode
- `MOVIE_REPEAT_FPS_LIMIT float`
  Set a frame rate limit, above which the `MOVIE_REPEAT` mode is activated. When using this option `MOVIE_REPEAT` should be disabled, otherwise it is used at any frame rate.
- `OPEN_AIR_FILE file`
  Filename and path of OpenAir™file
- `PROJECTION integer`
  Which projection should be use. See 5.11
- `QUADS bool`
  Use quadrilaterals instead of triangles to build the terrain surface.

- `SAVE_PATH string`
  The location to store screenshots can be defined by its full path.
- `SCALE_Z float`
  Scaling factor for z-axis (altitude) (default=3.0)
- `SEALEVEL integer`
  Altitude of sea-level (this is the limiting altitude between colourmap and colourmap-sea)
- `SEALEVEL2 integer`
  If sealevel2 is given, the terrain beneath this value will not be displayed, but covered by sea
- `SEALEVEL3 integer`
  If sealevel3 is given, the terrain beneath this value will be covered by (transparent) sea
- `SHADE bool`
  Usage of goraud shading
- `SPEED_UNIT_FAC float`
  Factor applied to the speed. 1.0 is km/h
- `SPEED_UNIT_NAME string`
  Name of the speed units
- `SPINNING float`
  This activates spinning around the marker position in movie mode. In terrain viewer mode this will spin around the centre. float is the spinning step size in degrees, the sign will decide about the direction.
- `STEREO bool`
  This activates double image stereo mode
- `STEREO_HW bool`
  This activates hardware stereo mode (start-time-option)
- `STEREO_RB bool`
  This activates red/blue anaglyphic stereo mode
- `STEREO_RG bool`
  This activates red/green anaglyphic stereo mode
- `TEXT_COLOR_[R|G|B] float`
  If you don't like the deafault colour of the text (for lifts and waypoints) you can change it with these keywords.
- `TEXT_LINEWIDTH float`
  This changes the linewidth of the text for lifts and waypoints
- `TIME_ZONE integer`
  Difference between UTC and your time zone. Do not use the + sign for positive numbers, but − for negative.
- `TIME_ZONE_NAME string`
  Name of your time zone
- `VERBOSE bool`
  If this option is active, some output will be made. This option is read if neither –quiet nor –debug are given. (Default=off)

- `VSPEED_UNIT_FAC float`
  A factor with which the vertical speed is multiplied. 1.0 is m/s
- `VSPEED_UNIT_NAME string`
  Name of the vertical speed units
- `WAYPOINTS_FILE string`
  Sets the filename (and path) for a waypoint file. See section 3.11 for details.
- `WAYPOINTS bool`
  Determines the default behaviour for drawing waypoints.
- `WP_COLOR_[R|G|B] float`
  Can be used to change the colour of the spheres representing the waypoints.
- `WINDOW_HEIGHT integer`
  Initial height of the *OGIE* window in pixels
- `WINDOW_WIDTH integer`
  Initial width of the *OGIE* window in pixels
- `WIRE bool`
  If this option is activated, the surface of the terrain will be drawn as a wire frame model

E Known Bugs

Errors / Syntax errors in install.sh

If you get a message like

```
./install.sh: 40: Syntax error: "do" unexpected (expecting "}")
```

after starting the ./install.sh script, you're using a shell, which is not compatible to the bash shell. `install.sh` needs to be run through a bash compatible shell. Try something like
`sudo /bin/bash ./install.sh`
or
`sudo /bin/zsh ./install.sh`
Shells, known to work: `bash, ksh, zsh`
Shells, known not to work: `csh, dash`

Menu in OGIE (freeglut)

With older freeglut versions (2.4.0) some problems arise due to the positioning and the size of the menu, which sometimes is placed in a way, that parts of it don't fit the screen. In this case it's possible to release the mouse pointer (key m — also see next bug, sorry) and right-click in a different position. That way it's possible to reach all entries from the menu. In freeglut 2.6.0 this seems to be solved.

Crash on activating the mousepointer (freeglut 2.4.0)

Ogie will crash, if the mousepointer is activated by key m. This is a known bug in freeglut. There is a patch for this in freeglut-cvs. On Gentoo systems use freeglut-2.4.0-r1 or later.

ogie zombi process on windows after using exit window button

If you quit ogie by using the exit (x) button from the window decoration, ogie may end up as a zombie process, consuming lots of cpu time. Use the ESC key, or quit from the menu to end ogie.

Weird speedogram plot

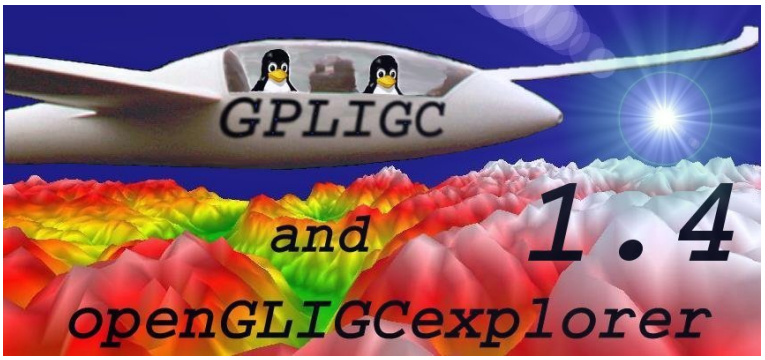
If your (very old) perl-distribution provides the module Math::Complex earlier than version 1.26, some errors will occur. Most likely negative values in speedogram! This is a bug in the old Math::Complex perl module.

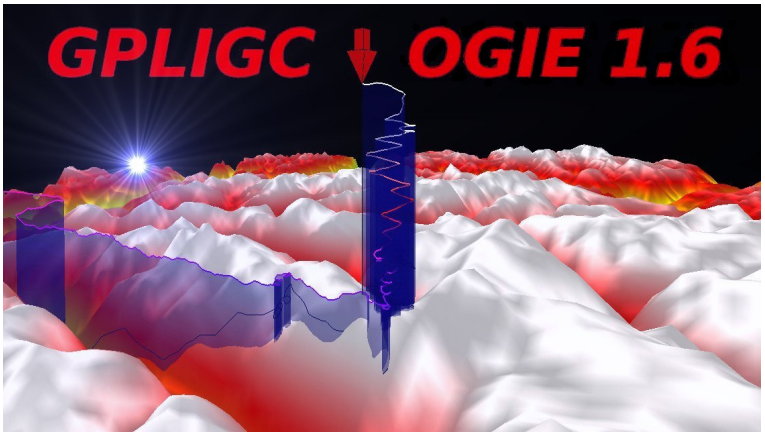
Annoying open dialog

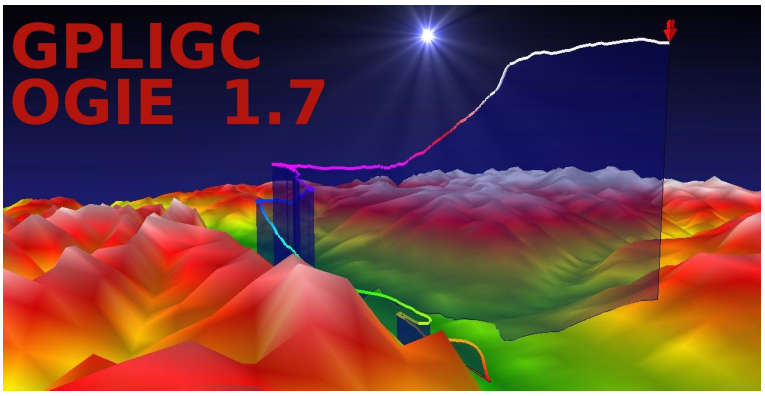
The "open File" dialog starts always in "/". This is a bug in the Tk module. To avoid that, update to Perl/Tk Module "Tk-800.024" or later. This is a bug in the Perl/Tk module before 800.024

F Logo Gallery

You'll find the logos from previous releases here...




GPLIGC OGIE 1.6


GPLIGC
OGIE 1.7


GPLIGC & OGIE 1.8

G The GNU General Public License

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

Terms and Conditions For Copying, Distribution and Modification

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the

Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   (a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   (b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   (c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

   Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

   In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

   (a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

(b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

(c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

No Warranty

11. Because the program is licensed free of charge, there is no warranty for the program, to the extent permitted by applicable law. Except when otherwise stated in writing the copyright holders and/or other parties provide the program "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the quality and performance of the program is with you. Should the program prove defective, you assume the cost of all necessary servicing, repair or correction.

12. In no event unless required by applicable law or agreed to in writing will any copyright holder, or any other party who may modify and/or redistribute the program as permitted above, be liable to you for damages, including any general, special, incidental or consequential damages arising out of the use or inability to use the program (including but not limited to loss of data or data being rendered inaccurate or losses sustained by you or third parties or a failure of the program to operate with any other programs), even if such holder or other party has been advised of the possibility of such damages.

End of Terms and Conditions

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

> one line to give the program's name and a brief idea of what it does.
> Copyright (C) yyyy name of author
>
> This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

> Gnomovision version 69, Copyright (C) yyyy name of author
> Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type 'show w'.
> This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

> Yoyodyne, Inc., hereby disclaims all copyright interest in the program
> 'Gnomovision' (which makes passes at compilers) written by James Hacker.
>
> signature of Ty Coon, 1 April 1989
> Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

References

[1] Fédération Aéronautique Internationale, Lausanne, Switzerland. *International Gliding Commission*. http://www.fai.org/gliding.

[2] Thomas Schank. gpspoint. http://pc12-c714.uibk.ac.at/gpspoint, 2003.

[3] H. Krüger. GSIL: GPL Symbian IGC Logger. http://pc12-c714.uibk.ac.at/GSIL/GSIL.php, 2010.

[4] GPSBabel: Free software for GPS data conversion and transfer. http://www.gpsbabel.org, 2002–2010.

[5] H. Krüger. *GPLIGC & OGIE*. Bremen, Innsbruck, Canberra, 2000–2010. http://gpligc.sf.net. Support: http://sourceforge.net/projects/gpligc/support. Mailing list: https://lists.sourceforge.net/lists/listinfo/gpligc-announce.

[6] Free Software Foundation, Inc., Boston, USA. http://www.fsf.org/.

[7] gnuplot. http://www.gnuplot.info.

[8] Perl.com — the source for perl. http://www.perl.com.

[9] Nick Ing-Simmons. Perl Tk. http://search.cpan.org/~srezic/Tk, 1995–2004.

[10] The freeglut project. http://freeglut.sourceforge.net.

[11] The mesa 3d graphics library. http://www.mesa3d.org.

[12] The fink project. http://www.finkproject.org.

[13] ActiveState Software Inc., Vancouver, Canada. *ActivePerl*. http://www.activestate.com/activeperl.

[14] Phil Harvey. Image::ExifTool. http://search.cpan.org/~exiftool/Image-ExifTool, 2003–2009.

[15] CPAN: Comprehensive Perl Archive Network. http://www.cpan.org.

[16] NGDC, ETOPO2 Global 2' Elevations. http://dss.ucar.edu/datasets/ds759.3/.

[17] The Global Land One-km Base Elevation Project (GLOBE). http://www.ngdc.noaa.gov/mgg/topo/globe.html.

[18] GTOPO30. http://eros.usgs.gov/Find_Data/Products_and_Data_Available/gtopo30_info.

[19] SRTM30 PLUS Global Topography. http://comlmaps.org/how-to/layers-and-resources/physical-environment/srtm30-plus-global-topography.

[20] SRTM version 2. http://dds.cr.usgs.gov/srtm/.

[21] The national map seamless server. http://seamless.usgs.gov.

[22] USGS DEM Resources. http://data.geocomm.com/dem/demdownload.html.

[23] read_dem.c – Convert USGS DEM files to LAS format. ftp://dbftp.essc.psu.edu/pub/code/read_dem.c.

[24] John Leibacher. Worldwide Soaring Turnpoint Exchange — Airspace. http://soaringweb.org/Airspace/HomePage.html.

[25] gpsd — a GPS service daemon. http://gpsd.berlios.de.