GeoELAN v2.1 2023-01-03

This is the manual for GeoELAN, a tool for annotating action-camera GPS logs via the ELAN annotation software.

Please cite this document as

> Larsson, J. 2022. GeoELAN Manual. Lund: Lund University Humanities Lab. https://gitlab.com/rwaai/geoelan

Acknowledgments

> GeoELAN was developed with support from the Bank of Sweden Tercentenary Foundation (Grant nos NHS14-1665:1 and IN17-0183:1).
>
> We would also like to acknowledge the The Language Archive, Max Planck Institute for Psycholinguistics in Nijmegen for their tireless efforts in developing ELAN, and making it available for free.

Introduction

GeoELAN is a command-line tool that geo-references time-aligned text-annotations of observed phenomena in audiovisual recordings, captured with a recent GoPro or Garmin VIRB action camera, see Larsson et al 2021. In other words, GeoELAN is used for annotating action camera GPS logs with the help of ELAN.

By annotating a section representing an on-site utterance, a plant that is in view, or anything else that was captured, the annotation can be automatically linked to the corresponding coordinates. The nature of the workflow also means consultants not physically present at the the time of recording may evaluate observed phenomena to be geo-referenced post-collection. As the name implies, the free ELAN annotation software plays a central role and is required to annotate events. The final output can be points, polylines, or polygons (circles), in the form of annotated KML and GeoJSON-files. Henceforth, "GoPro" refers to a GoPro Hero 5 Black or later, and "VIRB" to the Garmin VIRB Ultra 30. Note that while GeoELAN functionality differs slightly between Garmin and GoPro due to differences in formats and file structure, the general workflow and the final output are the same.

> GeoELAN is multi-functional tool that can
>
> - geo-reference ELAN-annotations of GoPro and VIRB footage (i.e. annotate GPS logs) and output these as annotated points, lines, or circles.
> - inspect the content of your GoPro GPMF data, or Garmin FIT-files.
> - locate and match all relevant files belonging to the same recording session (clips, telemetry-files).
> - automatically concatenate clips for a specific recording, and generate an ELAN-file with linked media.

Requirements

- An action camera with a built-in GPS. Supported devices are:
  - GoPro Hero Black 5 or newer (Max, and Fusion cameras have not been tested)
  - Garmin VIRB Ultra 30 (documentation)
- ELAN (documentation)
- FFmpeg (for concatenating video)
- Rust toolchain (optional, only required for compiling GeoELAN from source)

Installation

- Download the zip-file from https://gitlab.com/rwaai/geoelan or use `git clone https://gitlab.com/rwaai/geoelan.git`.
- See the `bin` directory for pre-compiled executables for Linux, macOS, and Windows.

Compile and install from source

You can also compile GeoELAN from source. Depending on your operating system, this may require installing additional software, and a basic understanding of working in a terminal. The basic steps are:

1. Install the Rust toolchain
2. Get the GeoELAN source from https://gitlab.com/rwaai/geoelan (via `git` or the zip-file)
3. `cd geoelan` (you should be in the folder containing `Cargo.toml`)
4. `cargo build --release`
5. `cargo install --path .` (optional, makes `geoelan` a global command)

Before you start

Practical advice

- For both *Garmin* and *GoPro*: **keep the original files!** These contain unique identifiers used for synchronisation and/or telemetry. Converting files will usually discard the identifiers along with other crucial data, including the GoPro GPS-log.
- **Preserve *GoPro* original filenames!** GoPro MP4-files currently contain no identifier that can be used for determining the chronological order of the clips. GeoELAN uses the original file names as a substitute.
- *Garmin* stores all telemetry, such as the GPS-log, in a separate file in the FIT-format. Make sure you keep all files on the microSD, unless you are absolutely certain which files are relevant.
- *Garmin* MP4-files contain a unique identifier that is used to match FIT-files and MP4-files, but also for synchronisation. **Converting the original MP4-files will discard this identifier**.
- *GoPro* embeds all telemetry inside the MP4-clips. **Converting the original MP4-files will discard all telemetry in the output file**.

Running GeoELAN

- GeoELAN is a command line tool and has no graphical user interface.
- FFmpeg is required to concatenate clips. (commands `virb2eaf`, `gopro2eaf`)
- If you use macOS and GeoELAN does not run, see https://support.apple.com/en-us/HT202491.
- The terminal command is `geoelan` on linux/macOS, and `geoelan.exe` on Windows.

Device compatibility

- Garmin: Only VIRB Ultra 30 has been tested extensively, but earlier VIRB models may still work.
- GoPro: Only "main line" Hero cameras with GPS have been tested, but Max and Fusion cameras may still work.

GPS

- Make sure the GPS is turned on and in reach of a satellite. This make take a couple of minutes or longer, especially if you have not used the camera for a while or have traveled far between uses.
- On the VIRB's screen the GPS-icon should be steady, not blinking (it may log coordinates while the icon is still blinking, but do not rely on this being the norm).

Annotating in ELAN

- It is best to limit each kind of observed phenomena you wish to geo-reference to a single ELAN-tier, so…
- …to keep e.g. place names and plant sightings within the same ELAN-file make a separate tier for each (see the example walkthrough in the next section). Then you can just re-run GeoELAN on the same ELAN-file and select another tier to geo-reference along with other output options.

Example walkthrough

This section describes an example of how GeoELAN can be used to geo-reference ELAN-annotations. Please refer to the detailed sections if you get stuck. Note that all input video clips must be the unprocessed, original MP4 (GoPro + VIRB) and FIT-files (VIRB). The so-called FIT-files mentioned throughout this manual are where the VIRB logs GPS-data and other kinds of telemetry during a recording session. These need to be matched to the corresponding video recording (see *The FIT-format and the Garmin VIRB* for further information). GeoELAN will help with all of this, with the exception of annotating your data.

As noted below, some of the commands differ slightly between GoPro and VIRB due to differences in file and data structures.

The basic steps are:

1. Record video with a recent GoPro or Garmin VIRB action camera.
2. Use GeoELAN to concatenate the video clips and generate an ELAN-file.
3. Annotate spatially interesting sections in ELAN using the pre-generated ELAN-file.
4. Use GeoELAN to geo-reference the annotations, resulting in annotated KML and GeoJSON files.

Input files

VIRB

- `VIRB0001-1.MP4`, any clip in a recording session (remaining clips located automatically)
- FIT-file with corresponding GPS-data (located automatically)

GoPro

- `GH010026.MP4`, any clip in a recording session (remaining clips located automatically)

Output files

VIRB + GoPro

- KML and GeoJSON files with ELAN annotation content synchronised and mapped to the corresponding points as descriptions. See the command *eaf2geo* for other options.

Step 1/3: Generate an ELAN-file with linked media files

In step 1 we will locate all video clips (GoPro + VIRB) and FIT-files (VIRB) that belong to a specific recording session, process these, and generate an ELAN-file with linked media files.

VIRB

Command

```
geoelan virb2eaf --video INDIR/VIRB0001-1.MP4 --indir INDIR/ --outdir OUTDIR/
```

Output files

```
OUTDIR/VIRB0001-1/
├── VIRB0001-1.mp4     High-resolution video (concatenated)
├── VIRB0001-1_LO.mp4  Low-resolution video for ELAN (concatenated)
├── VIRB0001-1.wav     Extracted audio for ELAN (concatenated)
├── VIRB0001-1.eaf     ELAN-file with pre-linked media files
├── VIRB0001-1.kml     Overview KML-file with all points logged during the recording session
└── VIRB0001-1.txt     FFmpeg concatenation file, paths to input clips
```

GoPro

Command

```
geoelan gopro2eaf --video INDIR/GH010026.MP4 --indir INDIR/ --outdir OUTDIR/
```

Output files

```
OUTDIR/GH010026/
├── GH010026.mp4     High-resolution video (concatenated)
├── GH010026_LO.mp4  Low-resolution video for ELAN (concatenated)
├── GH010026.wav     Extracted audio for ELAN (concatenated)
├── GH010026.eaf     ELAN-file with pre-linked media files
├── GH010026.kml     Overview KML-file with all points logged during the recording session
└── GH010026.txt     FFmpeg concatenation file, paths to input clips
```

Explanation of the command

GeoELAN locates and concatenates all clips belonging to the recording session starting with `VIRB0001-1.MP4`/`GH010026.MP4`, then generates an ELAN-file with the resulting audio and video files pre-linked.

The relevant sub-commands are `virb2eaf`/`gopro2eaf`, depending on camera. By specifying any clip in the recording session via `--video`, the remaining clips (GoPro + VIRB), including the corresponding FIT-file (VIRB), will be automatically located as long as these exist somewhere in the specified input directory (`--indir`). Sub-directories will be searched as well. The result, including the corresponding FIT-file for VIRB cameras, will be saved to the specified outout directory (`--outdir`).

If low-resolution clips (`.GLV`/`.LRV`) are located, a concatenated low-resolution video will be linked in the ELAN-file. If not, the concatenated high-resolution video will be linked instead.

GeoELAN defaults to *not* insert a tier with geo-data in the ELAN-file due to the effect this may have on performance. To do so, use the `--geotier` flag (see *Geo-data in ELAN*). The result, including the FIT-file (VIRB), is copied to a folder named after the first clip in the session under the specified output directory (`--outdir`).

> **TIP:** For longer recording sessions, resulting in many video clips, step 1 is usually much faster if `--indir` and `--outdir` is not on the same physical hard drive. Those with an SSD (standard on most modern laptops) should be fine running step 1. on a single drive however.

Step 2/3: Annotate events in ELAN

In step 2 the user annotates events in ELAN (as per normal) they wish to geo-reference using the generated ELAN-file.

GeoELAN will geo-reference annotations from a single tier, selectable in step 3. Thus, if you want

to generate a KML-file with e.g. indigenous place names mentioned on-site during the recording, all place names must be limited to a single tier. When the annotations are geo-referenced in step 3, their textual content will be used as descriptions for the corresponding points in the KML and GeoJSON-files. Points corresponding to unannotated sections of the ELAN-file will either be discarded or have no description, depending on which options you use in step 3.

An annotated event can relate to anything observed in the recording and can be represented as either points or polylines in the output KML-file. If you are unsure which best applies to what you have in mind for your data, or how this may affect how you annotate, here are a few ideas for each kind.

> **Points** could concern documenting:
>
> - **the location of a plant or a geographical feature**, e.g. annotate the the timespan either is visible in the video.
> - **an uttered place name or an animal cry**, e.g. annotate the timespan it is uttered or heard on-site.
>
> For these specific cases, the exact time spans of the annotations are not that important. It should be enough to ensure the annotation lasts for the duration of the place name being uttered, or for as long as the plant is visible. If unsure, add a another second to the annotation timespan. An average coordinate will be calculated for those that were logged within each annotation's time span, so as long as the camera wearer does not stray too far from the observation point, the result should be accurate enough.
>
> **Lines** could concern documenting:
>
> - various **types of movement through the landscape**. To annotate the movement of "walking up-hill" as it is observed visually in the recording set the annotation's start time at the bottom of the hill and its end at the top, or for as long as the motion can be observed.
> - a **narrative reflecting on the immediate surroundings** as they change over time. E.g. comments on visible landscape features, or perhaps the re-construction of an historical event as it unfolded over space and time.

If you wish to geo-reference several categories of phenomena in a single ELAN-file for a specific recording session, create a separate tier for each category. In step 3 you can then re-run GeoELAN

as many times as required, then select a different tier and/or options on each run.

Step 3/3: Generate a KML-file from geo-referenced ELAN annotations

Now that we have a few annotations, we can geo-referenence these by determining which points were logged within each annotation's timespan. Note the different commands between GoPro and VIRB.

This is where you choose the approriate geographical representations for your annotated phenomena. Here are suggestions for the examples in step 2.

> **Points**:
>
> - **the location of a plant or a geographical feature**
> - **an uttered place name or an animal cry**
>
> To get a single, average coordinate for each annotation in the KML and GeoJSON-files, use the `--geoshape point-single` option.
>
> **Lines**:
>
> - **types of movement through the landscape**
> - **narrative reflecting on the immediate surroundings**
>
> Two line options may apply to the above. To get a continuous polyline alternating between marked (annotated) and unmarked (un-annotated) events, use the option `--geoshape line-all`. To get a broken-up polyline representing marked events only, use the option `--geoshape line-multi`.

There are other options, such as *circle* output. It is similar to point output, for which radius and height can be specified (all circles will have the same size). For a more detailed overview of the possibilities, see the `--geoshape` option for the command *eaf2geo*. Experiment! If you realise one representation is not appropriate after all, re-run GeoELAN with a different option.

VIRB

Command

```
geoelan eaf2geo --eaf VIRB0001-1.eaf --fit 2003-01-02-12-00-00.fit --geoshape point-single
```

Output files

```
OUTDIR/VIRB0001-1/
├─ ...                              Existing files
└─ VIRB0001-1_point-single.kml      New KML-file, one point per ELAN-annotation in the selected
tier
└─ VIRB0001-1_point-single.geojson  New GeoJSON-file, one point per ELAN-annotation in the selected
tier
```

GoPro

Command

```
geoelan eaf2geo --eaf GH010026.eaf --gpmf INDIR/GH010026.MP4  --geoshape point-single
```

> **Important:** `GH010026.MP4` must be the original video generated by the camera, indicated by `INDIR` from step 1/3.

Output files

```
OUTDIR/GH010026/
├─ ...                            Existing files
└─ GH010026_point-single.kml      New KML-file, one point per ELAN-annotation in the selected tier
└─ GH010026_point-single.geojson  New GeoJSON-file, one point per ELAN-annotation in the selected
tier
```

Explanation of the command

GeoELAN geo-references all annotations in a single ELAN-tier (selectable from a list) for the specified ELAN-file and generates an annotated KML-file where each point represents a single annotation.

The relevant command is `eaf2geo`. By specifying an ELAN-file (`--eaf`) and the corresponding GoPro MP4-file (`--gpmf`) or VIRB FIT-file (`--fit`), GeoELAN will synchronise the annotations with the coordinates contained within the MP4/FIT-file. This process is usually completely automatic.

`--geoshape point-single` lets GeoELAN know that each, respective annotation should be distilled into a single point, meaning that the generated KML-file will contain as many points as there are

annotations in the selected tier. Each point inherits the corresponding annotation text for the selected tier as its description. The KML-file is named according to the selected `--geoshape` option, in this case `VIRB0001-1_point-single.kml`/`GH010026_point-single.kml`.

If the proces fails for VIRB footage, the user will be presented with a list of recording sessions present in the FIT-file (see *The FIT-format and the Garmin VIRB*). GoPro MP4-files lack the appropriate metadata to display such as list.

Commands

There are six commands:

Command	Alias	Description	Camera Model
`gopro2eaf`	`g2e`	Generate an ELAN-file, and link concatenated media files	GoPro
`virb2eaf`	`v2e`	Generate an ELAN-file, and link concatenated media files	VIRB
`eaf2geo`	`e2g`	Geo-reference ELAN-annotations and generate annotated KML/GeoJSON	GoPro, VIRB
`locate`	`l`	Locate and match video clips and/or FIT-files	GoPro, VIRB
`inspect`	`i`	Inspect the contents of a GoPro MP4-file or any Garmin FIT-file	GoPro, VIRB
`manual`	`m`	View or save this manual to disk	-

To see help for a specific command, run `geoelan <COMMAND> --help`.

The most relevant commands are probably `gopro2eaf`/`virb2eaf` and `eaf2geo`. `locate` is there to help with locating and matching video clips and/or FIT-files that belong to the same recording session, but this functionality partly exists in `gopro2eaf`/`virb2eaf` as well. `inspect` will print the data contents of a GoPro MP4/Garmin FIT-file, but will do so in an unprocessed form. It is intended more as a technical aid for troubleshooting or to verify the contents of MP4/FIT-files. `manual` is for viewing or saving the full manual.

Time adjustment with `--time-offset`

If the action camera has not adjusted for the current time zone, several commands have a `--time-offset` option. It takes a +/- value in hours that will be applied to all timestamps in the output, e.g. `--time-offset 7` will add seven hours to all timestamps.

Reducing the number of coordinates with `--downsample`

For the commands `gopro2eaf`/`virb2eaf`, and `eaf2geo`, the output may contain coordinates in some form (a KML/GeoJSON-file or as annotations in an ELAN-file). Since the VIRB GPS logs at 10Hz, the resulting files may become too heavy to work with in some cases, such as loading a KML-file in Google Earth. GoPro logs at 10 or 18Hz depending on model, but saves points as a cluster each second. Since it is only the cluster that gets timestamped, and not the individual points, GeoELAN averages these to a single, timestamped point at the same interval the cluster was logged originally (roughly once/second).

`--downsample` can be used for both ELAN and KML/GeoJSON output to reduce the number of coordinates that are imported/exported. It is especially advised for KML-files that are to be loaded into Google Earth, since a 2 hour VIRB recording may contain up towards 72 000 logged points. `--downsample` takes a positive numerical value that is effectively a divisor: `--downsample 10` means an average coordinate will be calculated for every cluster of 10 points. For 72 000 logged points, a value of 100 means the output will contain 720 points and so on. If the user sets `--downsample` to a value that exceeds the total number of points logged by the GPS, it will be changed to the largest applicable value (resulting in a single point for the entire recording as opposed to none at all). Extreme values may also affect the result in unexpected ways, depending on gaps in the GPS-data.

If 'gopro2eaf'/'virb2eaf' or 'eaf2geo' return errors

Try the `inspect` command on problematic MP4/FIT-files. This way you can verify whether points were actually logged or not. If the file is corrupt the error message will also be printed.

> **TIP:** GeoELAN will never overwrite existing files without permission. Should you accidentally delete the generated ELAN-file with the output media files intact, just re-run the `gopro2eaf`/`virb2eaf` command. It will automatically skip concatenating videos, but still generate a new ELAN-file.

> **TIP:** In the tables for the respective command sections, arguments listed under 'Flags' do not take a value, whereas those listed under 'Options' do. If a `default` value is listed, it will be automatically set, unless the user specifies otherwise.

FFmpeg

For the commands `gopro2eaf` and `virb2eaf` FFmpeg is required. If you intend to use the *static build* for FFmpeg as described in the appendix under *FFmpeg*, point to it using `--ffmpeg PATH/TO/FFMPEG/ffmpeg` (`ffmpeg.exe` on Windows). If the `--ffmpeg` option is not used, `geoelan` will assume `ffmpeg` is available as a global command and complain accordingly if it is not.

gopro2eaf

- *Camera model:* GoPro
- *Command/alias:* `gopro2eaf` / `g2e`
- *Help:* `geoelan gopro2eaf --help`
- *Basic usage:* `geoelan gopro2eaf --indir INDIR/ --video GH010026.MP4 -i --outdir OUTDIR/`

`gopro2eaf` locates and concatenates all clips for the specified recording session found in the input directory (`--indir`) and exports a concatenated WAV-file. An ELAN-file is then generated with the media files pre-linked. The result is copied to the specified output directory (`--outdir`). By specifying a clip in the relevant session (via `--video`), the remaining files will be automatically located if the `--concatenate` option is used.

Flags

Short	Long	Description
`-c`	`--concat`	Concate clips for recording session starting with 'video' parameter
	`--geotier`	Insert tier with synchronised coordinates in ELAN-file
`-l`	`--low-res-only`	Only concatenate low resolution clips (`.LRV`)
	`--dryrun`	Show results but do not process or copy files

Options

Short	Long	Description	Default	Required
	`--ffmpeg`	Custom path to FFmpeg	`ffmpeg`	
`-g`	`--gpmf`	GoPro MP4-file		unless `-u` or `-v`
`-i`	`--indir`	Input path for locating files		yes
`-o`	`--outdir`	Output path for resulting files	`OUTPUT`	
`-d`	`--downsample`	Downsample factor for coordinates	`1`	
`-t`	`--time-offset`	Time offset in +/- hours	`0`	
`-v`	`--video`	Unaltered GoPro MP4-file		unless `-f` or `-u`

Example 1

`geoelan`	`gopro2eaf`	`-g GH010026.MP4`	`-i INDIR/`	`-o OUTDIR/`	`--geotier`
	command	clip in session	input directory	output directory	insert coordinate tier

Result: Locates all clips for the recording session containing the clip `GH010026.MP4` (`-g`) in the input directory `INDIR/` (`-i`). These will be concatenated, and the audio track exported as a WAV for use in ELAN. The resulting files are then copied to the output directory `OUTDIR/` (`-o`). The generated ELAN-file will also have synchronised coordinates inserted as a tier (`--geotier`).

virb2eaf

> - *Camera model:* VIRB
> - *Command/alias:* `virb2eaf` / `v2e`
> - *Help:* `geoelan virb2eaf --help`
> - *Basic usage:* `geoelan virb2eaf --indir INDIR/ --video VIRB0001-1.MP4 --outdir OUTDIR/`

`virb2eaf` locates and concatenates all clips for the specified recording session found in the input directory (`--indir`) and exports a concatenated WAV-file. An ELAN-file is then generated with the

media files pre-linked. The result is copied together with the corresponding FIT-file to the specified output directory (`--outdir`). By specifying any clip in the relevant session (via `--video`, `--uuid` or `--fit`), the remaining files will be automatically located. Optionally, the high-resolution clips can also be copied to the output directory as-is (`--copy`).

Flags

Short	Long	Description
	`--copy`	Copy, do not concatenate, high resolution clips
	`--geotier`	Insert tier with synchronised coordinates in ELAN-file
`-l`	`--low-res-only`	Only concatenate low resolution clips (`.GLV`)
	`--dryrun`	Show results but do not process or copy files

Options

Short	Long	Description	Default	Required
	`--ffmpeg`	Custom path to FFmpeg	`ffmpeg`	
`-f`	`--fit`	VIRB FIT-file		unless `-u` or `-v`
`-i`	`--indir`	Input path for locating files		yes
`-o`	`--outdir`	Output path for resulting files	`OUTPUT`	
`-d`	`--downsample`	Downsample factor for coordinates	`1`	
`-t`	`--time-offset`	Time offset in +/- hours	`0`	
`-u`	`--uuid`	UUID for a VIRB clip in the relevant session		unless `-f` or `-v`
`-v`	`--video`	VIRB clip in the relevant session		unless `-f` or `-u`

> **Note:** Recording session can be specified using one of `--fit`, `--uuid`, `--video`. These options are mutually exclusive. `--fit` returns a list of sessions present in the FIT-file, from which the user can select the relevant one. `--uuid` and `--video` require no further user input.

Example 1

`geoelan`	`virb2eaf`	`-v VIRB0001-1.MP4`	`-i INDIR/`	`-o OUTDIR/`	`--geotier`
	command	clip in session	input directory	output directory	insert coordinate tier

Result: Locates all clips for the recording session containing the clip `VIRB0001-1.MP4` (`-v`) in the input directory `INDIR/` (`-i`). These will be concatenated, and the audio track exported as a WAV for use in ELAN. The resulting files are then copied together with the corresponding FIT-file to the output directory `OUTDIR/` (`-o`). The generated ELAN-file will also have synchronised coordinates inserted as a tier (`--geotier`).

Example 2

`geoelan`	`virb2eaf`	`-f 2017-01-28-05-16-40.FIT`	`-i INDIR/`	`-o OUTDIR/`	`--low-res-only`
	command	FIT-file	input directory	output directory	do not concatenate hi-res MP4

Result: The relevant recording session is specified via the FIT-file `2017-01-28-05-16-40.fit` (`-f`). This presents the user with a list of sessions to select from, which allows GeoELAN to locate the clips in the input directory `INDIR/` (`-i`). Only the low-resolution clips (`--low-res-only`) will be concatenated. All resulting files are then copied together with the corresponding FIT-file to the output directory `OUTDIR/` (`-o`).

> **Tip:** If you are unsure of the whereabouts of the FIT-file, make the search wider. Specifying the root of an external hard drive as input directory (`--indir`) will make the search process take slightly longer, but should work well. Otherwise, just specify the FIT-file separately (`--fit`), which can be useful if it is located outside of the input directory.

eaf2geo

- *Camera model:* GoPro, VIRB
- *Command/alias:* `eaf2geo` / `e2g`
- *Help:* `geoelan eaf2geo --help`
- *Basic usage:* `geoelan eaf2geo --eaf VIRB0001-1.eaf --fit 2017-01-28-05-16-40.fit`

`eaf2geo` generates a KML-file by geo-referencing the annotations in one tier for the specified ELAN-file. The user is presented with a list of tier names to select from. Several options exist for the KML-file, depending on the `--geoshape` option. The result can be either points or polylines (see below). The resulting KML-file contains absolute timestamps and will embed annotation values as a description for any point that was logged within an annotation's timespan. If the relevant FIT-file can not be automatically located, it must be specified separately.

Flags

Short	Long	Description
	`--cdata`	KML-option, added visuals in Google Earth

Options

Short	Long	Description	Default	Possible	Required
`-d`	`--downsample`	Downsample factor for coordinates	`1`		
`-e`	`--eaf`	ELAN-file			yes
`-f`	`--fit`	VIRB FIT-file			unless `-g`
`-g`	`--gpmf`	GoPro MP4-file			unless `-f`
	`--geoshape`	Output options for KML-file	`point-all`	`point-all`, `point-multi`, `point-single`, `line-all`, `line-multi`, `circle-2d`, `circle-3d`	
	`--height`	Circle height ('circle-3d')	`10.0`		

Short	Long	Description	Default	Possible	Required
	`--radius`	Circle radius ('circle-2d', 'circle-3d')	`2.0`		
`-t`	`--time-offset`	Time offset, +/- hours	`0`		
`-u`	`--uuid`	UUID for first VIRB clip in a session			unless `-e` or `-v`
	`--vertices`	Circle vertices/roundness ('circle-2d', 'circle-3d')	`40`		

Example

`geoelan`	`eaf2geo`	`-f 2017-01-28-05-16-40.fit`	`-e VIRB0001-1.eaf`	`--geoshape point-single`
	command	FIT-file	ELAN-file	output option

Result: Geo-references annotations and generates a KML-file with a single point per annotation (`--geoshape point-single`) in the ELAN-file `VIRB0001-1.eaf` (`-e`). Since no video is specified (i.e. the first clip in the recording session), the user will be presented with a list of UUIDs in the specified FIT-file `2017-01-28-05-16-40.fit` (`-f`) to choose from, each representing the first clip in a recording session.

The *geoshape* option

Different geographical representations are available for the output KML and GeoJSON-files, including points and lines. Six possible `--geoshape` values are accepted:

Option	Description
`point-all`	All logged points exported (default if no option passed)
`point-multi`	Exported points correspond to marked/annotated events only

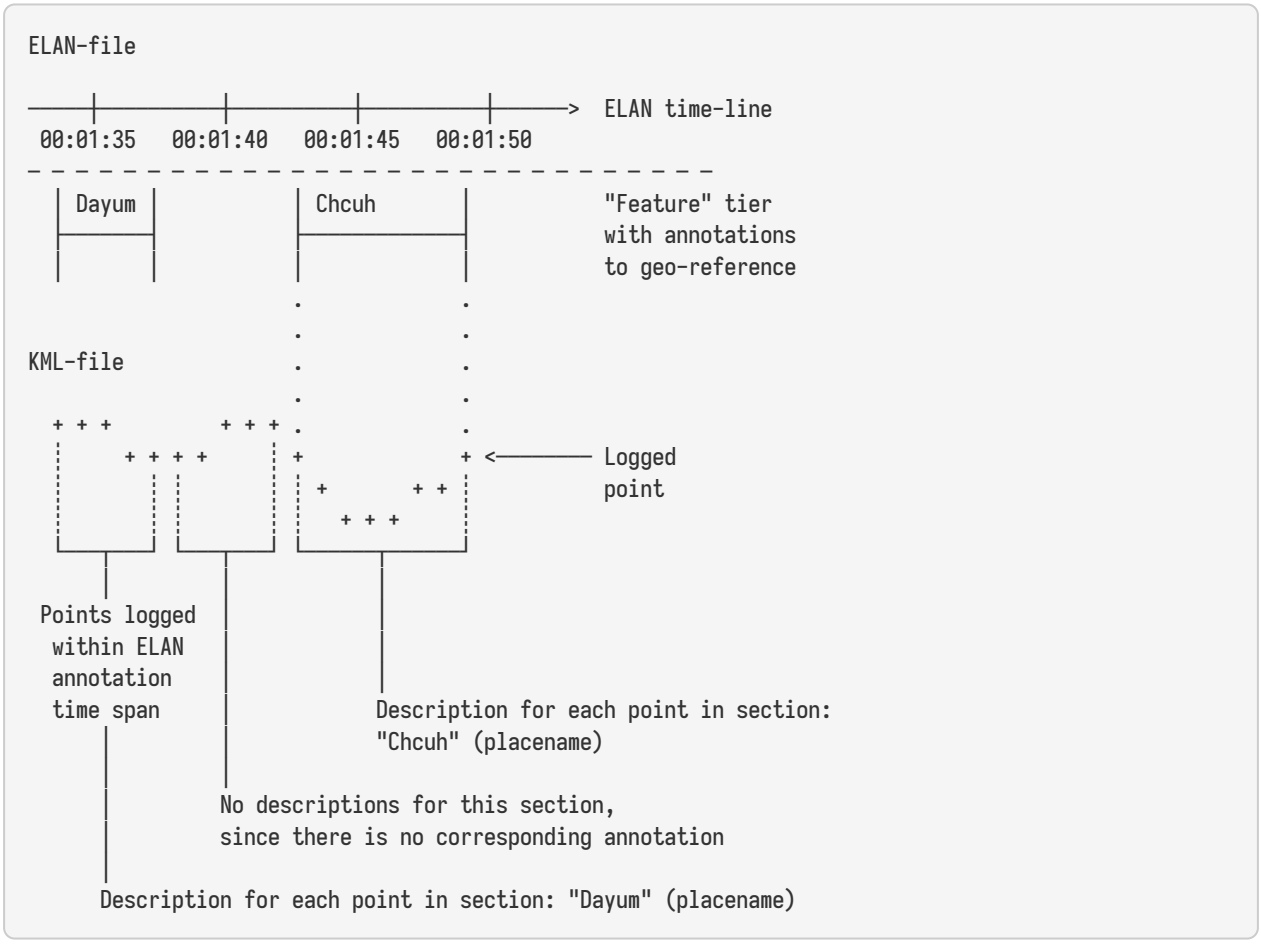
---

Option	Description
`point-single`	A single, averaged point for each annotation
`line-all`	Polyline from all logged points
`line-multi`	Polyline, corresponds to marked/annotated events only
`circle-2d`	2D polygon, corresponds to marked/annotated events only
`circle-3d`	3D polygon, corresponds to marked/annotated events only

`--downsample` can be used with all these options, but will be ignored for `point-single`. `circle-2d` /`circle-3d` allows for further customisation, such as height (`circle-3d`, KML-only) and radius. The circle options are mostly a visualisation flair, since radius and height are not currently derived from ELAN annotation values.

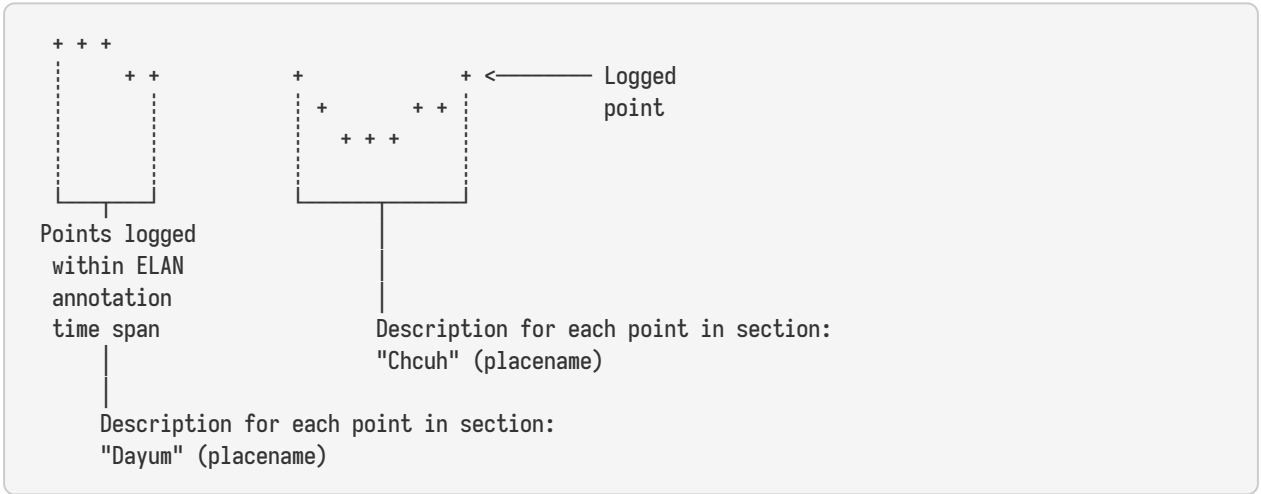
`point-all`

All points logged during the recording session will be exported. Any point that intersects with the time span of an annotation will inherit the annotation text as the coordinate description. Points that do not, will have no description.



`point-multi`

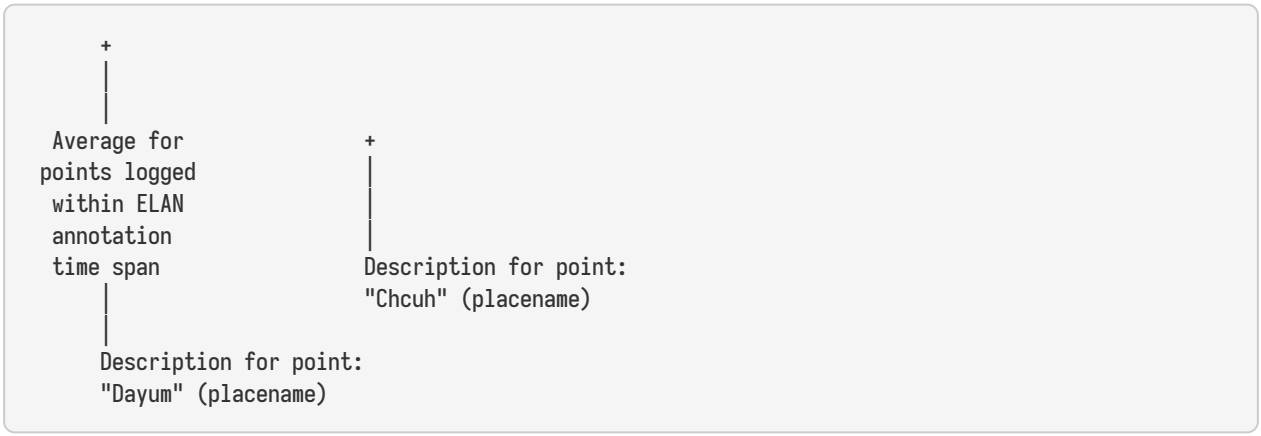
Only points that intersect with the time span of an annotation will be exported and will inherit the annotation text as the coordinate description. Points that have no corresponding annotation will be discarded. *Useful for including points corresponding to marked events only*.



`point-single`

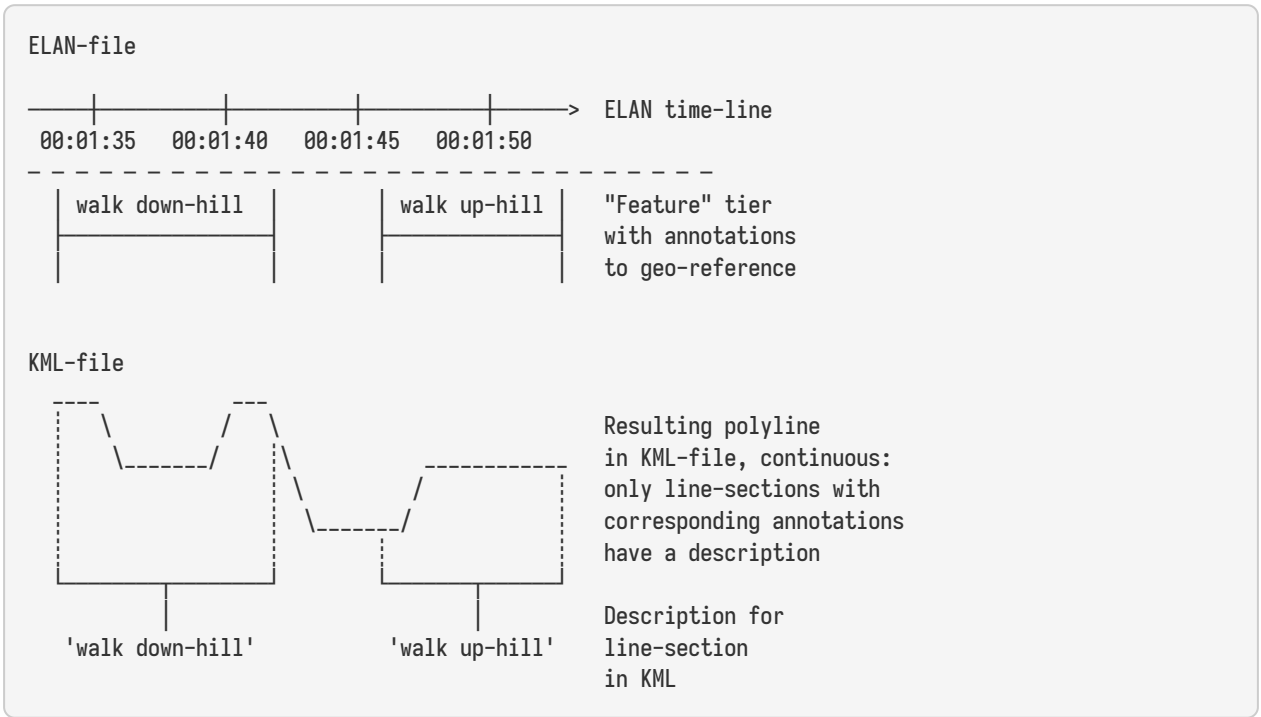
Only points that intersect with the time span of an annotation will be considered for export. The

difference to `point-multi` is that each annotation will only generate a single point: an average of those logged within the annotation's time span. Note that a custom `--downsample` value will be ignored for `point-single` since it may affect the result negatively (it also has little use, since the number of points in the output will not change and will be quite low compared to the other options). *Useful for distilling marked events, such as place names, to a single point for each event*.



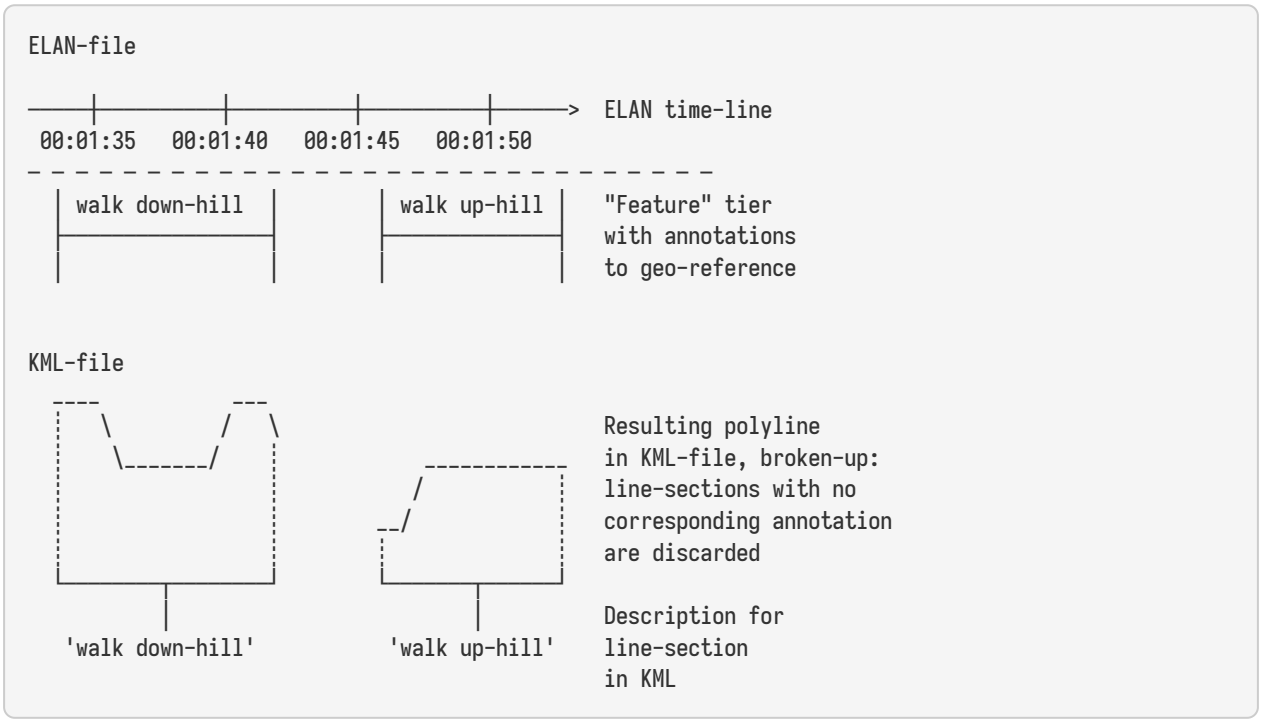
`line-all`

All points logged during the recording session will be exported, resulting in a continuous polyline. Sub-sections that intersect with an annotation inherit the annotation text as a description, whereas those that do not will have no description.



`line-multi`

Only points that intersect with the time span of an annotation will be exported, resulting in a broken-up line. Each sub-section inherits the text value of the annotation it intersects with. *Useful for representing paths corresponding to marked events only*.



`circle-2d`, `circle-3d`

`circle-2d`, `circle-3d` works almost exactly like `point-single` with the difference that a circle is generated around the calculated average point. It is mostly a visual flair and its shape is currently not affected by annotation values. `circle-2d` is flat against the ground, whereas `circle-3d` can take a height value to become a cylindrical 3D shape (only applies to KML, not GeoJSON). If circle output is specified, three more options become available:

`height`	**Height relative to ground in meters (`circle-3d`)**
`radious`	Radius in meters (`circle-2d`, `circle-3d`)
`vertices`	Roundness, valid range 3 - 255 (3 will literally be triangle)

The 'cdata' option

Only affects KML-files. Using `--cdata` will insert extra information into the KML-file in the form of HTML inside the `<description>` element for each point (see the CDATA section in Google's KML documentation). This will cause an information bubble to pop-up in Google Earth when a point is

clicked on, as a visual flair for e.g. presentations.

locate

- *Camera model:* GoPro, VIRB
- *Command/alias:* `locate` / `l`
- *Help:* `geoelan locate --help`
- *Basic usage:* `geoelan locate --indir INDIR/ --kind gopro`

`locate` will locate and match original GoPro and VIRB clips in the input path. For VIRB, corresponding FIT-file/s will also be matched. By optionally specifying the first UUID (`--uuid`, `--fit`) or the first clip (`--video`) for a specific session, only paths for the files in that recording session will be returned. A CSV-file of the result can also be saved for future reference. If you are unsure of the location of all VIRB-files, use an input path closer to the root, such as the root of an external hard drive. If duplicate files are found, only the first one encountered will be reported.

Flags

Short	Long	Description
	`--duplicates`	Include duplicate files in match results
	`--quiet`	Do not print file-by-file search progress

Options

Short	Long	Description	Required
`-f`	`--fit`	VIRB FIT-file for selecting session	
`-i`	`--indir`	Input path for locating files	yes
`-u`	`--uuid`	UUID for first VIRB clip in a session	
`-v`	`--video`	First VIRB clip in a session	

Example 1

`geoelan`	`locate`	`-i INDIR/`	`--kind gopro`
	sub-command	input directory	only look for GoPro files

Result: Locates all GoPro clips in `INDIR/` (`-i`) and groups them in recording sessions.

Example 2

`geoelan`	`locate`	`-i INDIR/`	`-v VIRB0001-1.MP4`
	sub-command	search directory	first clip in session

Result: Locates all VIRB clips in `INDIR/` (`-i`) for the recording session that contains `VIRB0001-1.MP4` (`-v`) together with the corresponding FIT-file.

inspect

- *Camera model:* GoPro and VIRB
- *Command/alias:* `inspect` / `i`
- *Help:* `geoelan inspect --help`
- *Basic usage:*
  - GoPro: `geoelan inspect --gpmf GH010026.MP4`
  - VIRB: `geoelan inspect --fit 2017-01-28-05-16-40.fit`

`inspect` prints an overview or the detailed contents of a GoPro MP4 or a Garmin FIT-file. If a Garmin VIRB MP4 is passed with no other options, the embedded UUID will be printed. Options include filtering to print only a sub-set of the data, such as GPS-data only, data corresponding to a specific recording session, or both. As previously mentioned, it is more of a technical aid or for example to verify that the GPS really did log coordinates. Optionally, a KML-file can also be generated.

Flags

Short	Long	Description
	`--debug`	Print FIT definitions and data while parsing
	`--kml`	Generate a KML-file
	`--ikml`	Generate an indexed KML-file

Short	Long	Description
	`--json`	Generate a GeoJSON-file.
	`--verbose`	Print raw data
	`--gps`	Print processed GPS data
	`--meta`	Print MP4 custom user data (`udta` atom)
	`--atoms`	Print MP4 atoms: FourCC, offset, size
`-s`	`--session`	VIRB: Select session from a list. GoPro: Merges session data.

Options

Short	Long	Description	Default	Required
`-f`	`--fit`	FIT-file		yes
`-g`	`--gpmf`	GoPro-file (MP4 or raw GPMF-file)		yes
`-t`	`--type`	Data type (VIRB: numerical, GoPro: string)		
`-u`	`--uuid`	UUID, first in session (VIRB)		
`-v`	`--video`	VIRB video clip		

Inspecting telemetry

`inspect` will mostly print raw values that require further processing to be of use. The exact nature of this data differs between GoPro and Garmin. For GPS data, the flag `--gps` can be used for either device to print a processed GPS-log showing coordinates in decimal degrees etc. The other GeoELAN commands, such as `eaf2geo`, always convert data to the relevant forms.

If a GoPro MP4 or a Garmin FIT-file can not be properly parsed, GeoELAN will often return an error message that may hint at the issue. Try `inspect` on files that raise errors with the other commands.

MP4-files have a few options, besides inspecting embedded GoPro GPMF data. The `--meta` flag will show raw (i.e. bytes) content for the so-called `udta` atom. GoPro also embeds undocumented GPMF data in the `udta` atom which will also be listed, whereas Garmin embeds a unique identifier

(UUID). The `--atoms` flag will show MP4 atom FourCC, offset. To most users these will not be of much use, but may provide technical context for troubleshooting. If `--atoms` is not enough, try the command line tool AtomicParsley.

GoPro

GoPro logs and embeds all telemetry within the MP4-files. In contrast to Garmin FIT, data types have no numerical identifier (see below) so internal text descriptions are used instead. GPS is identified as "GPS (Lat., Long., Alt., 2D speed, 3D speed)", for example.

To list all data types logged in a GoPro MP4-file, run:

```
geoelan inspect --gpmf GOPROVIDEO.MP4
```

This will list all data streams (`DEVC` is an internal container of sorts that holds the data listed):

```
Unique data stream types (1018 DEVC streams in total):
    Accelerometer
    Average luminance
    Exposure time (shutter speed)
    Face Coordinates and details
    GPS (Lat., Long., Alt., 2D speed, 3D speed)
    Gyroscope
    Image uniformity
    Predominant hue[[hue, weight], ...]
    Scene classification[[CLASSIFIER_FOUR_CC,prob], ...]
    Sensor ISO
    Sensor read out time
    White Balance RGB gains
    White Balance temperature (Kelvin)
```

Find the data type you wish to inspect further in the list. To print GPS data in its "raw" form, run:

```
geoelan inspect --gpmf GOPROVIDEO.MP4 --type "GPS (Lat., Long., Alt., 2D speed, 3D speed)"
```

Note the citation marks. These are necessary to pass the description as a single string to GeoELAN. Your terminal shell may otherwise try to interpret anything delimited by space as a command or a parameter, which will fail. Unlike FIT, GPMF has no numerical identifier for data types.

GPS specifically, can be printed in a more conventional form via the flag `--gps`:

```
geoelan inspect --gpmf GOPROVIDEO.MP4 --gps
```

Generate a KML-file with:

```
geoelan inspect --gpmf GOPROVIDEO.MP4 --kml
```

Images

If you have original JPEG-images snapped with a GoPro camera, these can also be inspected, but will contain much less GPMF data than the MP4-files. They are currently not used elsewhere in GeoELAN's workflow. So far no named data streams have been found so use `geoelan inspect --gpmf GOPROIMAGE.JPG --verbose` to print the raw data.

Garmin FIT

The FIT-format is quite different to GoPro's GPMF, apart from being a separate file. There is among other things, additional information about VIRB recording sessions. The VIRB starts logging to a FIT-file the moment the camera is turned on, and only stops when it is turned off. This means that a single FIT-file may contain data corresponding to multiple recording sessions. For as long as the camera is turned on, data is logged continuously - even between recordings.

Inside a FIT-file, data is identified by a numerical identifier. For example, GPS data is `160`, also referred to as `gps_metadata` in the FIT Software Development Kit (FIT SDK). `inspect` lists both identifiers in the summary table.

To list all data types logged in a VIRB FIT-file, run:

```
geoelan inspect --fit FITFILE.FIT
```

This will return a table:

```
 Global ID | Message type                | Count
.............................................
         0 | file_id                     |     1
        18 | session                     |     1
        19 | lap                         |     1
        20 | record                      |  6209
        21 | event                       |     1
        22 | UNKNOWN_TYPE_22             |     2
        23 | device_info                 |     3
        34 | activity                    |     1
        49 | file_creator                |     1
       104 | UNKNOWN_TYPE_104            |   104
       160 | gps_metadata                | 60114
       161 | camera_event                |    24
       162 | timestamp_correlation       |     1
       164 | gyroscope_data              | 20405
       165 | accelerometer_data          | 20405
       167 | three_d_sensor_calibration  |    59
       208 | magnetometer_data           | 20405
       209 | barometer_data              |  6209
       210 | one_d_sensor_calibration    |     1
       219 | UNKNOWN_TYPE_219            |     1
.............................................
                                 Total:  133948
```

Find the data type you wish to inspect further in the list and take note note of the "Global ID". To print GPS data in its "raw" form, run:

```
geoelan inspect --fit FITFILE.FIT --type 160
```

GPS specifically, can be printed in a more conventional form via the flag `--gps`:

```
geoelan inspect --fit FITFILE.FIT --gps
```

Generate a KML-file with:

```
geoelan inspect --fit FITFILE.FIT --kml
```

To print a single type of data belonging to a specific recording session (VIRB only) use `--session`:

```
geoelan inspect --fit FITFILE.FIT --type 160 --session
```

This will return a table listing all VIRB recording sessions together with an input prompt (UUIDs shortened to fit):

```
 Session | Clips | First UUID in session
.....................................................................................
  1.     |  1    | VIRBactioncameraULTRA30_Tall_2688_2016_29..._1_17_2017-01-28-05-16-40.fit
  2.     |  1    | VIRBactioncameraULTRA30_Tall_2688_2016_29..._1_18_2017-01-28-05-16-40.fit
  3.     |  3    | VIRBactioncameraULTRA30_Tall_2688_2016_29..._1_19_2017-01-28-05-16-40.fit
  4.     |  1    | VIRBactioncameraULTRA30_Tall_2688_2016_29..._1_20_2017-01-28-05-16-40.fit
  5.     |  1    | VIRBactioncameraULTRA30_Tall_2688_2016_29..._1_21_2017-01-28-05-16-40.fit
  6.     |  1    | VIRBactioncameraULTRA30_Tall_2688_2016_29..._1_22_2017-01-28-05-16-40.fit
  7.     |  1    | VIRBactioncameraULTRA30_Tall_2688_2016_29..._1_23_2017-01-28-05-16-40.fit
.....................................................................................
Select session:
```

Type the number in the "Session" column for the relevant session and press enter. The output will now be limited to the selected recording session, which also applies when generating KML/GeoJSON-files.

You could also specify an original VIRB video via `--video`, or a UUID directly (any UUID in the session) via `--uuid` to achieve the same result.

To find out the embedded UUID of a VIRB MP4-file, run:

```
geoelan inspect --video VIRBVIDEO.MP4
```

This will return the embedded UUID:

```
UUID: VIRBactioncameraULTRA30_Expansive_1920_1440_29.9700_3937280306_3af2a648_1_299_2021-05-03-14-23-23.fit
```

Most FIT-files, from e.g. watches, bike computers, will work with `inspect`. Custom developer data is also supported. However some FIT features are exclusive to VIRB, such as UUID and selecting sessions, and other features are not implemented, such as compressed timestamp headers. In such cases, the tool will report the error and exit. Missing features may or may not be implemented in future versions.

> **Tip:** For those who wish to dig deeper, the Garmin FIT Software Development Kit contains a spreadsheet, `Profile.xlsx`, which lists the kinds of data a FIT-file may contain. Not all of those apply to every device however (VIRB has a limited set of data types from the ones listed in `Profile.xlsx`), and undocumented data types exist.

manual

- *Command/alias:* `manual` / `m`
- *Help:* `geoelan manual --help`
- *Basic usage:* `geoelan manual --pdf`

`manual` exports or prints the contents of this file to screen, bute the full PDF-manual is also embedded within the compiled executable for convenience. Running `geoelan manual` with no flag prints the full manual to screen.

Flags

Short	Long	Description
		Print full plain text version to screen
	`--pdf`	Save the full manual as a PDF to current directory
	`--pdf-a4`	Save the A4-guide as a PDF to current directory

Appendix

A few notes and help texts on ELAN, telemetry formats, video processing etc.

GoPro and Garmin telemetry formats

As of GeoELAN v2.0 GoPro Hero 5 Black and newer is supported. There are significant differences between Garmin's FIT-format and GoPro's GPMF-format. Here are a few:

	Garmin FIT	GoPro GPMF
Storage form	Separate file (binary)	Embedded in MP4 (binary)
Time stamps	Explicit, absolute time stamps for each data point	Absolute time stamps for GPS log only, otherwise have to be derived from MP4 timing
GPS	10Hz, individual points time stamped	10 or 18Hz, logged in one-second clusters. Only the cluster is time stamped.

Documentation and development

Support for GPMF (GoPro) and FIT (VIRB) formats were written from scratch for GeoELAN with the help of the official documentation for both formats.

- Garmin FIT development kit and documentation: https://developer.garmin.com/fit/
- GoPro GPMF documentation and example code: https://github.com/gopro/gpmf-parser

GoPro

File structure

GoPro recording sessions are split over multiple clips depending on recording time, quality settings, and SD card size. GoPro provide an estimate here: https://community.gopro.com/s/article/GoPro-Camera-File-Chaptering-Information?language=en_US

SD Cards

GoPro cameras use micro SD Cards. GoPro provide recommendations here: https://community.gopro.com/s/article/microSD-Card-Considerations?language=en_US.

Internal file layout

Since Hero 5 Black all GoPro cameras use a telemetry format called GPMF developed by GoPro. Currently, the best overview can be found in their Github repository: https://github.com/gopro/gpmf-parser. (Note that GeoELAN does not make use of GoPro's parser. Instead, we developed a GPMF-parser from scratch tailored for our needs).

Garmin VIRB

File structure

Example VIRB SDCard file structure:

```
├── DCIM
│   └── 100_VIRB
│       ├── VIRB0001-1.GLV          Recording session split up
│       ├── VIRB0001-1.MP4          as 10 minutes clips. Low (GLV)
│       ├── VIRB0001-2.GLV          and high (MP4) resolution clips.
│       └── VIRB0001-2.MP4
└── GMetrix
    ├── 2017-01-01-12-00-00.fit     Telemetry files, a.k.a "FIT-files".
    └── 2017-01-02-12-00-00.fit     May contain data, such as GPS-logs,
                                    for multiple recording sessions.
```

The VIRB and the FIT-format

To pair and match VIRB video clips belonging to the same recording sessions with a FIT-file unique identifiers (UUID) are embedded both within the original video clips and the FIT-files. Preserving these are key to synchronise and extract relevant GPS-data.

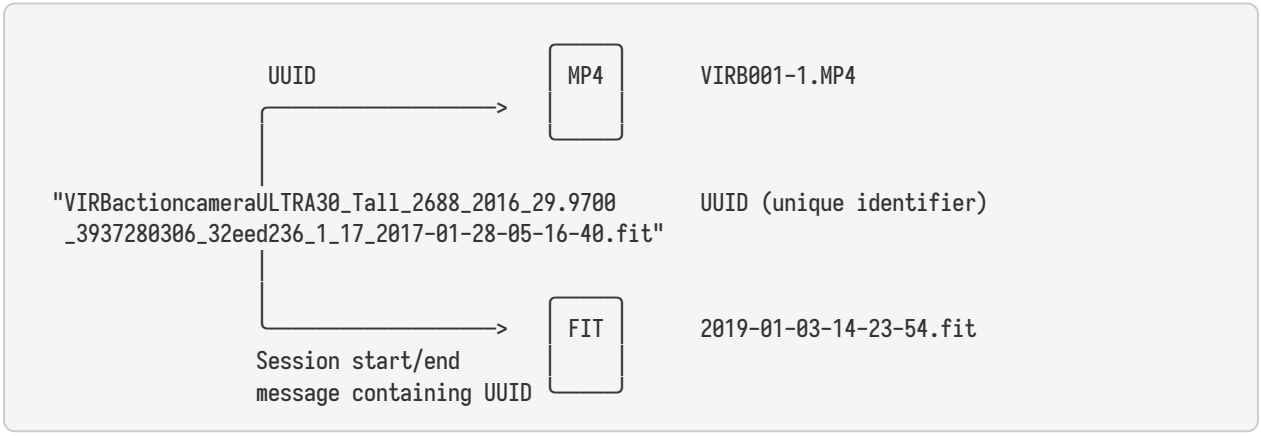
When synchronising and locating data, GeoELAN will sometimes list all sessions present in the FIT-file. As a help, the number of video clips and the *UUID for the first clip* in each session is listed.

A single FIT-file may contain telemetry for multiple recording sessions. When the camera is turned on, it immediately starts logging data into a new FIT-file, regardless of a video being recorded or not. The camera will keep logging to this file until completely turned off. If turned on again, a new FIT-file will be created. All data points in a FIT-file are explicitly timestamped, which technically allows synchronisation against any data type in the file. Further, with the help of the built-in GPS, absolute timestamps can be derived for all data types. These can be used for documentation purposes or to synchronise against external data sources.

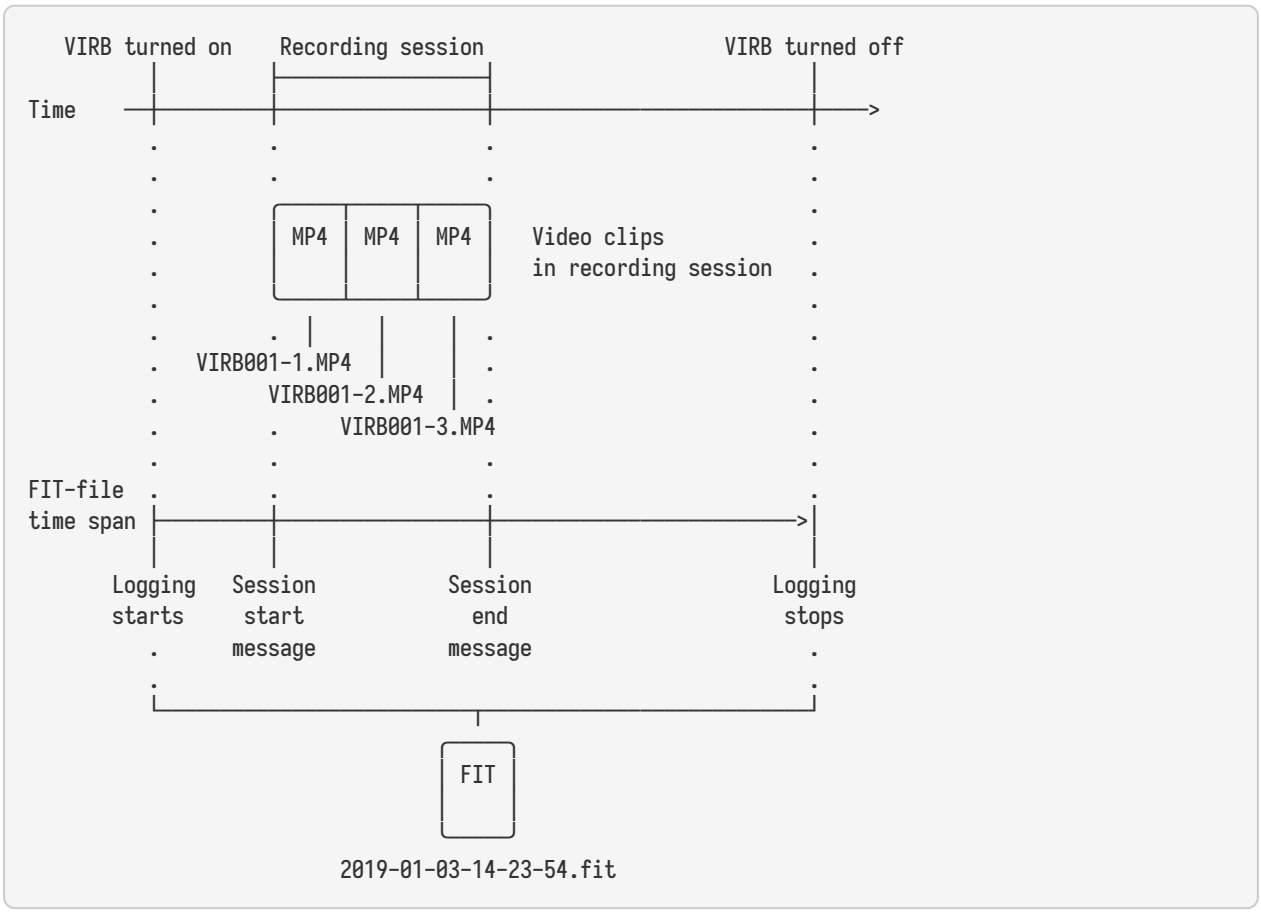
For geo-referenced annotations, GeoELAN always embeds absolute timestamps in the resulting KML-file.

The VIRB cameras split up recording sessions into video clips, each approximately 10 minutes in length, with no option to turn this off. To link VIRB video to its corresponding telemetry (e.g. coordinates logged by the GPS during the recording session), both the clips and the FIT-file contain UUIDs. When the user starts recording, a "video recording session start" message is logged to the current FIT-file together with the UUID embedded in the first clip, denoting the start of a recording session. Similarly, when recording ends, a "video recording session end" message is logged together with the UUID embedded in the last clip in the session. Since all logged FIT-data is timestamped, this creates a timeline for the session that can be related to any logged data in the FIT-file.

Matching MP4 and FIT-files via embedded UUIDs



Logging telemetry and boundaries for a recording session in a FIT-file



The VIRB logs location, barometric pressure, and rotation among many other data types. Since the FIT-format is not a text based data format, and thus cannot be inspected using a text editor, the `inspect` command allows for some exploration of a FIT-file (see command *inspect*). GeoELAN will also help out with matching recording sessions to the corresponding FIT-files (see commands *virb2eaf*, and *locate*).

Preserving UUIDs

Concatenating or converting the video clips will usually discard the UUIDs, so the user is advised to save the original video clips. The `inspect` command can be used to display the UUID for a specific VIRB MP4-file, just run `geoelan inspect --video VIRBVIDEO.MP4` with no other options.

Most of the commands allow for selecting UUID from those present in the relevant FIT-file when matching files or geo-referencing annotations. The `locate` command can also be used to locate all files for a specific session.

Video file management and options

On the VIRB MicroSD card, the low-resolution clips have a `.GLV` extension. These are generated by the VIRB for quick viewing on the internal camera display. If available, GeoELAN will prefer to link these in the ELAN-file over the high-resolution video due to their smaller size (both resolutions will still be concatenated by default). GeoELAN will not be able to identify the low-resolution `.GLV` as such if renamed to `.MP4` and they may even be mistaken for the high-resolution versions. If you only require the low-resolution videos to be concatenated, use the `--low-res-only` flag when running `virb2eaf`. This will ignore the high-resolution `.MP4`-files as a concatenation target, with an option to copy these as-is (`--copy`) to the output directory (see the *virb2eaf* section for further information).

FFmpeg

The `gopro2eaf` and `virb2eaf` sub-commands require FFmpeg for concatenating MP4-clips and to extract the audio track as a WAV-file (required to display a wave form in ELAN while annotating).

The video and audio streams are by default only concatenated, not converted, to avoid data loss and to save time, but note that **VIRB UUID and GoPro telemetry will still be discarded - save the original files**. See the *virb2eaf* section for more information.

There are two main options for installing FFmpeg:

1. Download the *static build* of FFmpeg, and specify its path using the `--ffmpeg` option
2. Install via a *package manager*. FFmpeg will be automatically available to `cam2eaf` in this case.

Static build The *static build* option means that the relevant media codecs are included in a single, executable file that can be used as is. The FFmpeg download page provides links to static builds for macOS, Windows and Linux. Put the downloaded `ffmpeg`-file in a convenient location and use the `--ffmpeg` option when running `gopro2eaf` or `virb2eaf`. Optionally moving or symlinking this file to a directory in PATH will yield the same result as using a package manager below.

Package manager Installing via a *package manager* means the `ffmpeg` command can be executed from anywhere in a terminal. Linux distributions usually come with one pre-installed. For macOS Homebrew is a popular choice, whereas Windows has Chocolatey (or WSL). This option means you do not have to specify the location of `ffmpeg` each time `gopro2eaf` or `virb2eaf` is run. If a package manager is not for you, go with the *static build* for your platform.

ELAN

ELAN is a completely free, advanced application for time-aligned annotations of audiovisual media developed by the The Language Archive, Max Planck Institute for Psycholinguistics in Nijmegen. While it is well-known in academia and particularly the humanities for transcribing recordings, its use goes well beyond this, since anything observed can be annotated, and thus time-aligned (GeoELAN is but one example). Since annotations are kept aligned in parallel on separate tiers - as many as required, similar to multi-track audio editors - the possibilites are almost endless. The ELAN Annotation Format is XML-based which makes it both human-readable and fairly straightforward to parse.

- Download (Windows, macOS, Linux): https://archive.mpi.nl/tla/elan/download
- Documentation: https://archive.mpi.nl/tla/elan/documentation

GeoELAN Rust crates

GeoELAN is written in Rust and uses three custom libraries (aka crates) that were developed from scratch in parallel with the tool itself. This was partly due to the need to get to know the formats better, but mainly because the only existing option was to create bindings to existing libraries in several other programming languages. Since these Rust crates are still in development they are not yet available on crates.io, but can if needed be copied from the GeoELAN source and used with the Apache 2.0 license attached to GeoELAN until released properly.

Three crates were developed for GeoELAN:

- `eaf-rs`: Read, write, and process EAF-files. Uses quick-xml and its serialization support via serde.
- `gpmf-rs`: Read GoPro GPMF-files. Either directly from GoPro MP4-files, or "raw" GPMF-files (e.g. a track extracted to file with FFmpeg).
- `fit-rs`: Read Garmin FIT-files. Supports custom developer messages. Some added bloat due to VIRB-specific functionality that seemed better to include in `fit-rs` directly.
- `mp4iter`: A simple crate to move around, search atoms on FourCC, and read values in an MP4 container. Includes finding byte offsets for tracks (`trak`) via atom `hdlr` component name, and deriving duration for the longest track. This crate does not (and will not) support any kind of media en/decoding. See https://developer.apple.com/library/archive/documentation/QuickTime/QTFF for more information on the QuickTime/MP4 container.

Data extracted with both `gpmf-rs` and `fit-rs` will mostly require further processing. Support for this is built-in for some data types (e.g. GPS data, since this is fundamental for GeoELAN), but for others you will have to develop and expand on this yourself. A first pass, extracting and parsing data, should always work for both crates. GeoELAN's `inspect` command with the `--verbose` flag prints data in this "raw" form.

References

Larsson, Jens, Niclas Burenhult, Nicole Kruspe, Ross. S Purves, Mikael Rothstein and Peter Sercombe. 2020. Integrating behavioral and geospatial data on the timeline: towards new dimensions of analysis. *International Journal of Social Research Methodology*. doi: 10.1080/13645579.2020.1763705

ELAN (Version 6.4) [Computer software]. 2022. Nijmegen: Max Planck Institute for Psycholinguistics. Retrieved from https://archive.mpi.nl/tla/elan