flight-analyzer

Overview

The `flight-analyzer` program is part of the scientific paper "Fliegen am Limit - Aktive Sicherheit im Gleitschirmsport", that was first published on 10/24/2022 and is being further developped by 03/31/2024 as of the "Schweizer Jugend forscht 2024" initiative. The application is designed to analyze flight data. The main functionality of the tool is the application of a selection of algorithms to process this data. As part of the paper, this tool is designed to deliver a clean dataset that can be used to conduct and optimize advanced analyses and simulate the stationary glide of a paraglider. Please find a detailed description of the algorithms in the sections below, in the paper itself or in the code.

The original paper (10/24/2022) can be downloaded here: nicolas-huber.ch/docs.

As soon as the refined version of the 2022 version has been published, the link will be listed here. Until then, please refer to the original paper.

The documentation is available in `.pdf` format and can be downloaded here.

Contents

Technical documentation

Introduction

The goal of the `flight-analyzer` application is to automatically manipulate large datasets, which contain the track logs of paragliding flights. The program evaluates for each point of the flight if it is on a straight line or not. This allows further analysis and the filtering by position of the point. After the input dataset has been filtered the tool can apply a selection of algorithms, which have been developped as of the "Fliegen am Limit" paper, to simulate the stationary flight of a paraglider. This serves as a foundation for the development of new algorithms, more advanced models and conduct further analyses. In addition, the tool offers some helpers such as tools to visualize the manipulated data or preprocess raw input files.

Detailed descriptions can be found in docstrings and comments within the source code of this project.

Getting started

Make sure to install `Python 3.9` or higher on your machine. The code has only been tested for the Python versions 3.9, 3.10, 3.11 and 3.12-dev and should properly work on MacOS, Linux distributions and Windows. It's recommended to use pyenv to manage local python environments as well as dependencies. To run this project make sure to activate an environment that supports Python 3.9 or higher and then run `pip install -r requirements.txt`. The application should work fine with the dependencies (indicated version) listed in `requirements.txt`.

Architecture

The application is structured as follows:

```
flight-analyzer
├── .github/
├── assets/
├── docs/
│   ├── datasets/
│   └── reports/
├── src/
│   ├── algorithms/
│   ├── executor/
│   ├── helpers/
│   └── constants.py
├── tests/
├── main.py
├── main.ipynb
├── LICENSE.md
├── README.md
└── requirements.txt
```

Basic Usage

The main entry point of this application are the `main.py` and `main.ipynb` files. Both the Python file and the Jupyter Notebook output the same result - you can chose the file format that fits you best. If you'd like to visualize data or run a specific analysis check out the `/src/executors/` directory, which contains a collection of Jupyter Notebooks. Please find listed below some instructions and examples for the Notebooks in `src/executors/`, the `main` scripts and and some guidenines for preprocessing.

Preprocessing flight data

The flight data consumed by this program is read from `.igc files`, which are written by flight computers such as variometers or similar. Reading these files is currently not supported in this program, which is why external applications are used. It's recommended to use IGC2KML for `.igc` to `.kml` conversion and KML2CSV for `.kml` to `.csv` conversion as well as Microsoft Excel to clean up the raw table data.

After processing the tracklog using the linked tools you're dealing with `.csv` data of the following format:

▶ Show Data

```
name,description,altitudeMode,visibility,tessellate,WKT
<<< SOME RANDOM HTML>>>
"12:25:30 0m 5kmh 0m/s 0km",,"clampToGround",,"true","LINESTRING Z
(7.530683 46.213083 2612, 7.5307 46.213083 2612)"
"12:25:31 1m 0kmh +1m/s 0km",,"clampToGround",,"true","LINESTRING Z
(7.5307 46.213083 2612, 7.5307 46.213083 2612)"
...
```

This data is to be processed using a tool like Microsoft Excel to remove the HTML content that's marked as `<<< SOME RANDOM HTML>>>` as well as clean up the data. After your final manual preprocessing steps the `.csv` data is supposed to look like this:

▶ Show Data

```
name,description,altitudeMode,visibility,tessellate,WKT
12:25:30 0m 5kmh 0m/s 0km,,clampToGround,,TRUE,"LINESTRING Z (7.530683
46.213083 2612, 7.5307 46.213083 2612)"
12:25:31 1m 0kmh +1m/s 0km,,clampToGround,,TRUE,"LINESTRING Z (7.5307
46.213083 2612, 7.5307 46.213083 2612)"
```

Please check your input data before running any of the algorithms in this application to prevent unexpected errors. The preprocessed input is consumed by the `main` scripts in `.csv` or `.xlsx` format. It can also be manually fed into the `FileConvertor` helper class, on which further records can be found here.

Running main.py / main.ipynb

In order to run the main script of the `flight-analyzer` application make sure to prepare all necessary files and properly preprocess them. Please find instructions on preprocessing here.

Further documentation will follow as soon as the `main` script have been developped and properly tested.

Run individual analysis

To run a particular algorithm of the `flight-analyzer` application, e.g. to visualize data, please refer to the executor scripts that can be found in `src/executors/`. The algorithms and some examples can be found in the Algorithms section.

This application provides the following Notebooks:

- `execute_file_convertor.ipynb`: This Notebook allows you to normalize the manually preprocessed data.
- `execute_angle_analyzer.ipynb`: This Notebook allows you to process a normalized dataset. The AngleAnalyzer algorithm is applied.
- `execute_data_analyzer.ipynb`: This Notebook allows you to extract points on a straight line from a processed dataset.
- `execute_optimize_thresholds.ipynb`: This Notebook allows you to automatically determine the best thresholds and constants to process your dataset.

Further documentation on the inputs and outputs of these executors can be found in the Notebooks.

Algorithms

Development

Conventions

Please find naming conventions for this project linked here: click. In addition, static type annotations are used in this project. The codebase has been tested using the `pytest` module. The recent CI/CD status can be found at the top of this page. Click here for a detailed overview and unit testing logs. The code is formatted and linted in VS Code using the Black Formatter Extension and Pylint.

Contributing

At this time, the `flight-analyzer` project is not open for community contributions. The development is currently handled exclusively by Nicolas Huber. Your interest is appreciated and this section will be updated if the policy changes in the future.

Changelog

- **[1.0.0]** - Not released yet.

License & Intellectual Property

The source code of this application is licensed under the license linked here.

If not stated differently, the source code of this project is Nicolas Huber's intellectual property. External sources can be found in the code and are marked as such. Additionally, to improve code quality and speed up workflows, tools like GitHub Copilot and ChatGPT were used. AI generated content is flagged with the following notes:

- For documentation files: *This document { TITLE } has been written by { SOURCE } and verified by Nicolas Huber on { DATE }.*

- For code snippets: *# AI content ({ SOURCE }, { DATE }), verified and adapted by Nicolas Huber.*

AI tools are a powerful and valuable addition to improve the development workflow, as long as sources and contents are scientifically listed. Thus, it's valued a lot to provide proper listings. The following utilities have been used: GitHub Copilot, ChatGPT.

In consideration of the `LICENSE.md`, the licensee, who is considered as such at the point of downloading this application, agrees to respect the terms and conditions. The licensee undertakes to show respect for Nicolas Huber's intellectual property and to use it only in accordance with his instructions.

Thanks for noticing!

Disclaimer

The author is not responsible for any damage caused by the use of the software.
