traffic

a toolbox for processing and analysing air traffic data

Xavier Olive, ONERA

MONDAIS, 7th June 2021

xoolive / **traffic**

A toolbox for processing and analysing air traffic data

traffic-viz.github.io/

MIT License

- Code: https://github.com/xoolive/traffic/
  Documentation: https://traffic-viz.github.io/
- Started early 2018
- traffic, a toolbox for processing and analysing air traffic data, *Journal of Open Source Software* (4), 2019. DOI: 10.21105/joss.01518

Architecture

Several modules with a particular focus on:

- **access to data** (ADS-B, DDR, AIXM, B2B, METAR, OpenStreetMap, etc.)
- **trajectory processing**, *the most time consuming...*
- **specialised algorithms**
  → CPA, clustering, operational event labelling, etc.
- **data visualisation**
  → maps, interactive visualisations, etc.

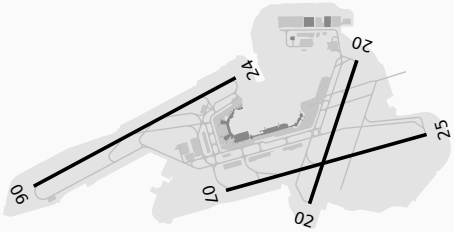
Core structures

```
from traffic.core import Flight, Traffic

Flight.from_file(...)  # one single trajectory
Traffic.from_file(...)  # a collection of Flight

flight.duration
flight.first("10 minutes")  # a new Flight
flight.intersects(my_tma)  # a boolean
flight.simplify()  # Douglas-Peucker simplification
flight.aligned_on_ils("LFPO")  # iterates on segments on final approach
flight.go_around()
```

Access to data

```
from traffic.data.samples import *  # for documentation and testing
from traffic.data.datasets import *  # public datasets included in publications

from traffic.data import airspaces  # public sources or AIRAC data
from traffic.data import airports  # airports, runways, apron structure
```

```
airports["LFPO"]
```



What coding can look like?

```python
for flight in quickstart:  # a sample dataset for the documentation
    if not flight.intersects(lfbo_tma):
        continue
    filtered = flight.filter()
    if filtered.min("altitude") < 10_000:
        if filtered.mean("vertical_rate") < -500:
            if filtered.aligned_on_ils("LFBO").next() is not None:
                landing_trajectories.append(filtered)
```

How coding “should” look like?

```
quickstart
.intersects(lfbo_tma)
.filter()  # clean values
.feature_lt('altitude_min', 10_000)
.feature_lt('vertical_rate_mean', -500)
.has("aligned_on_LFBO")
.eval()  # go!
```

- a flattened representation of the preprocessing is easier to proofcheck
- define a grammar of features meaningful from an operational point of view

Example code

- Select go-around situations at Zurich airport

```
my_dataset.has("aligned_on_LSZH").has("go_around").eval()
```

- Compute occupancy stats for a given airspace

```
my_dataset
.clip(airspaces["LFEE5R"])
.eval()
.summary(["callsign", "icao24", "typecode", "start", "stop", "duration"])
```

Demonstration

Use-cases

In today's presentation (hopefully):

- Go-around detection
- Midair collision
- Milestones for airport ground operations
- Point merge

More in the documentation:

- Trajectory clustering
- Weather reconstruction
- Landing configurations
- ...

Soon(ish) in the documentation:

- Occupancy metrics
- Closest point of approach
- Turbulence detection
- ...

Perspectives

- Provide reference data sets, enriched with metadata
- “Standardise” definitions of aircraft trajectory processing
  Encourage implementations in more programming languages:
  → `trrrj` (R), Javascript (in progress), Julia, OCaml, etc.
- Better tested, better documented (community effort)
- Go more scalable (Spark?)
- Interactive visualisation tools for in-depth analysis (Javascript, WebGL, more?)

Key take-aways

- Open-source is better than closed source
  … but it does not mean it is perfect ××× DISCLAIMER ×××
- Data has no immediate value
- Information is valuable, but the extraction process is hard
- Code is not precious, expertise is
- Humans read code, help experts read it too
- Use declarative style for better reproducibility