fishproviz: Fish Tracking Data Processing and Visualization Module

Project 21: Developing Exploration Behavior

Holds the scripts to visualize the molly's trajectory. To make use of the links to the mp4 and csv-files – connect to the server `loopbio_data`. Currently the links work for MacOS systems and best with Adobe Reader.

How to use the fishproviz module on the window PC in the Lab?

- fishproviz is installed on the left PC in the lab. These are the steps:

1. open Ubuntu-App and in the terminal type `cd project/fishproviz` and enter.
2. open the file `fishproviz/config.env` with: `notepad.exe fishproviz/config.env` - to configure the environment file and set the path variables that point to your working directory.
3. note that the prefix to the different data sources is `/mnt/`.

Requirements

- `python3, gcc, latex`
- Install dependencies:
  - `pip3 install -r requirements.txt`
  - `pip3 install pre-commit`
  - `pre-commit install` install the git hooks

Build

- To compile the $Cython$ code and creating you own `fishproviz/config.env`, run:
  - `python3 setup.py build_ext --inplace`
- To install the package and import functions elsewhere:
  - `python3 -m pip install .`
- Use Pandoc to generate a README.pdf:
  - `pandoc README.md -o README.pdf`

File Structure

The variable $path\_csv\_local$ in fishproviz/config.env is the root of the project and the place where all generated data is stored. In addition to the initial front an back directory where all the tracking data is stored you will find the following directories after the corresponding program executes.

Folder Structure:

.

```
|-- path_csv_local          # root folder of the data directory
|   |-- front               # POSITION_STR_FRONT front compartments tracks
|   |-- back                # POSITION_STR_BACK back compartments tracks
|   |-- area_config         # area_config Area Configurations
|   |   |-- front           # front
|   |   |-- back            # back
|   |-- visualizations      #
|   |   |-- trajectory      #
|   |   |-- feeding         #
|   |   |-- plots           # single plots
|   |-- config_data         # where we store feeding zones, area coordinates, calibration, e
|   |-- results             # folder we the results of metrics are stored
```

Configuration of the Program

After *building* - the file fishproviz/config.env contains all configuration and can be edited.

Run the main.py

```
usage: python3 main.py [-h] [-ti TIME_INTERVAL] [-fid FISH_ID] [--include_median]
                       {trajectory,feeding,trial_times,activity,turning_angle,
                       abs_angle,tortuosity,entropy,wall_distance,all,clear}

This program computes metrics and visualizations for fish trajectories,
the results are saved in the directory:
'DIR_CSV_LOCAL'

positional arguments:
  {trajectory,feeding,trial_times,activity,turning_angle,abs_angle,
  tortuosity,entropy,wall_distance,all,clear}
                        Select the program you want to execute

options:
  -h, --help            show this help message and exit
  -ti TIME_INTERVAL, --time_interval TIME_INTERVAL
                        Choose a time interval in second to compute
                        averages of metrics. Also possible [day, hour].
  -fid FISH_ID, --fish_id FISH_ID
                        Fish ID to run can be set by 'camera_position'
                        or index, default is all fish_ids
  --include_median      Include median or not only for activity
  -logs, --print_logs   Print logs

Example of use: python3 main.py trajectory -fid 0
```

1. Trajectory Visualization PDFs

The file fishproviz/config.env contains the paths to the trajectory data. One can configure these to point to the correct location of the data. Reading the data directly from the server `loopbio_data` results in long running times. It is recommended to use a external hard drive. If you path uses spaces, for example the name of the hard drive, rename it to underscores – _.
Accessing the data from the server is very slow.

1.1 Generate the Trajectory Visualizations, *run*:

- Trajectories: `python3 main.py trajectory`
- Feeding Trajectories: `python3 main.py feeding` The CSV-file for time spend feeding and number of visits are stored at `results/feeding`.
  - Requirements: Provide a csv-file (;-separated) with start and end time for the feeding measures with columns in the following format:

day	time_in_start	time_in_stop	time_out_start	time_out_stop
dd.mm.yy	hh:mm	hh:mm	hh:mm	hh:mm

An example template can be found at data/recordings_feeding_times_template.csv

Then run the `bash`-script:

- `bash scripts/build-trajectories.sh`
- Optional argument:
  - `--feeding` or `-f` for the feeding trajectories.
  - `--test`, `-t` is used to test the script, to generate only the fist pdf.
  - `--local`, `-l` to use the paths of the local hard drive to link the csv file in the pdf.
  - `--cam-id, -cam` followed by `cameraID_position`, to create only the pdf for the given camera.

Remark: For the bash-script you can not build feeding and non feeding trajectories in parallel as they use the same files.

2. Data File and Path Validation

The python script `path_validation.py` is used to validate the filenames and paths of the data files. It logs all error messages into `log-path-validation.txt`.

```
usage: python3 path_validation.py [-h] [--delete] [--n_files N_FILES] [--path PATH]
options:
  -h, --help         show this help message and exit
  --delete           If set, the duplicates will be deleted.
```

```
--n_files N_FILES  Number of files to expect in each folder, default is 15, for feeding us
--path PATH        Path to the directory that contains the folders front and back, default
```

3. Trajectory Analysis

- run: `python3 main.py <<metric>>`
- For **metric** use one keyword out of:
  - `activity, turning_angle, tortuosity, entropy, abs_angle, wall_distance`.
- run `python3 main.py <<metric>> --time_interval <<hour/day>>` to record mean and standard derivation per fish per hour/day in one csv-file.

3.1 Metrics:

- step length is the length of the vector drawn between to consecutive data frames.
- the mean and standard derivation illustrated in the visualization is computed from filtered data frames, removing obvious error point and normed by the distance between data frame when erroneous data point where removed.
- The number of spikes is defined by the threshold of ' > 10'cm in one step. The threshold can be changed in the config.env file.
- For the sum of angles we take each angle between consecutive steps anticlockwise '$\alpha \in [-\pi, \pi]$'.
- For the average angle each angle '$\alpha > 0$'

4 Data Visualizations

4.1 Entropy Density

- run: `python3 -m fishproviz.visualizations.entropy_plots` *plotly needs to be installed*
- run: `bach scripts/entropy_density.sh` The PDFs with show in tex/entropy_density

4.1 Metrics over 4 Weeks

- run: `python3 main.py program=all time_interval="day"` to calculate all metrics by day an save them to a csv
- run: `python3 -m fishproviz.visualizations.activity_plotting` to plot the data of the csv-files.
- run: `bash scripts/metrics.sh` to create the summery PDF.

Git Routines

- `git pull` to pull the changes from the remote repository

- `git checkout filename` to discard the changes in the file
- `git status` to check the status of the repository
- `git add .` to add all files to the staging area
- `git commit -m "commit message"` to commit the changes
- `git push` to push the changes to the remote repository

5. Linter and Style Checker for Python Use `flake8` to check the code style and linting. Install `flake8` with `pip install flake8` and run `flake8` in the root directory of the project. THe configuration file is `.flake8`. Use `black path_to_file.py` to format the code.

Further documentation will follow here...