Fish-Tracking-Visualization

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.

Requirements

- `python3, gcc, latex`
- Install dependencies:
  - `pip3 install -r requirements.txt`

Build

- To compile the *Cython* code and creating you own `script/env.sh`, run:
  - `python3 setup.py build_ext --inplace`

File Structure

The variable *path_csv_local* in scripts/env.sh is the root of the project and the place where all generated data is stored. In addition to the front an back directory where all the tracking data is stored you will find the following directories after the corresponding program executes. - visualizations/trajectory - visualizations/feeding - config_data - where we store feeding times, area coordinates, calibration, etc. - results

1. Trajectory Visualization PDFs

scripts/env.sh 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 program=trajectory`
- Feeding Trajectories: `python3 main.py program=trajectory feeding=1`
  The CSV-file for time spend feeding and number of visits are stored at `results/block{i}/feeding`.
- optional arguments: `fish_id=<<cam_pos or index in {0,...,23}>>`, `feeding={0,1}`
- Requirements: Provide a csv-file (;-separated) with start and end time for the feeding measures with columns in the following format:

block	day	time_in_start	time_in_stop	time_out_start	time_out_stop
blocki	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`.

2.1 Run the script:

- `python path_validation.py path="path/to/root"` – where you would find the directories for front and back position.
- for example: `python path_validation.py path="/Volumes/Extreme_SSD/FE_tracks"`
- Optional arguments:
  - `delete=1` – to delete duplicated filenames
  - `n_files=<<number of files>>` – to change the expected number of files in a folder for a day. The default is *15* for feeding use *8*.

3. Trajectory Analysis

- run: `python3 main.py program={metric}`
- For **metric** use one keyword out of:
  - `activity, turning_angle, tortuosity, entropy, abs_angle, wall_distance, all`.
- Optional arguments:
  - `time_interval=<<time in seconds>>` – default: `time_interval=100`
  - `fish_id=<<cam_pos>>`

  - `feeding={0,1}`
  - `include_median={0,1}` – only in combination with `program=activity`

- run `python3 main.py program={metric} time_interval=hour` to record mean and standard derivation per fish per hour in one csv-file.
- run: `bash scripts/build_analytics.sh` to generate the pdfs.

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 ' $> \mu + 3\sigma$'
- For the sum of angles we take each angle between consecutive steps anticlockwise '$\alpha \in [-\pi, \pi]$'.
- For the average angle each angle '$\alpha > 0$'

Compute: `function(fish_id, time_interval in sec)` In methods.py there are four function to calculate the metrics and store mean and standard derivation in `/results/<<time_interval>>_<<metrics_name>>.csv`.

- `activity_per_interval`
- `turning_angle_per_interval`
- `tortuosity_per_interval`
- `entropy_per_interval`

4 DATA Visualizations

4.1 Entropy Density

- run: `python3 -m src.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 src.visualizations.activity_plotting` to plot the data of the csv-files.
- run: `bash scripts/metrics.sh` to create the summery PDF.

Further documentation will follow here...