Fish-Tracking-Visualization

Project 21: Developing Exploration Behavior

Holds the scripts to visualize the molly's trajectory. The folder tex/trajectory contains the resulting pdfs. 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`

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=feeding`
- optional arguments: `fish_id=<<cam_pos>>`, `test={0,1}`

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`
- 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*.

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>>`
- 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 anti-clockwise '$\alpha \in [-\pi, \pi]$'.
- 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`

Further documentation will follow here...