Locomotion Python Package

Enabling the Package

The locomotion python package was developed using Python 2.7 and requires the plotly, numpy, and scipy packages.

To access the locomotion python package, you can

1. place the ‘locomotion’ folder into your work directory, or
2. add the package to your environment by entering the following* in your terminal:

```
export PYTHONPATH=$PYTHONPATH:/path/to/locomotion
```

*Note that /path/to/locomotion/ is the directory containing the locomotion package, not the package itself.

Now you can start python and call `import locomotion` from your python shell.

If the package cannot find or has trouble accessing the extendedDTW file, you will need to build the package to make it compatible to your system. To do so, you can use the extendedDTW code supplied with the locomotion package and **setup.py** in the extendedDTW directory to run:

```
python setup.py build_ext --inplace
```

Data Format

File Format

The package accepts csv and tsv files. However, because it distinguishes between tsv and csv by doing a simple check to see if tabs or commas are present in the first (header) line, make sure to avoid using both delimiters in the header of your data file. If your data must use both, you will need to edit the **`getRawData`** function in **animal.py**.

Header Format

The computations require X and Y coordinate data with corresponding column titles in string format, so the coordinate data columns must be labelled "X" and "Y", including the quotation marks. They can also be single quotes.

Information File Format

The information about the animals are stored in a json file, which is required to read in relevant data for each computation. The fields and format are as in this sample entry below. Note that all times are in minutes. In general, avoid using spaces in the field values.

```
{
        "name": "NSS_01", //Can be anything. Make it unique
        "data_file_location": "/data/medaka/NSS_01.dat",
        //full path to the data file
        "animal_attributes": {
            "species": "Medaka", //species name
            "exp_type": "NSS", //experiment type
            "control_group": "True", //True or False
            "ID": "01" //number, but in quotations
        },
        "capture_attributes": {
            "frames_per_sec": 20, //integer
            "pixels_per_mm": 1.6, //float
            "dim_x": 200, //in mm
            "dim_y": 100, //in mm
            "start_time": 0, //in min
            "end_time": 10, //in min
            "baseline_start_time": 0, //in min
            "baseline_end_time": 2 //in min
        }
    }
```

To generate the info file, you can use **infosheetGenerator.py** provided along with the locomotion package, which will populate a json file in the correct format by prompting the user for each necessary piece of information. It should run similarly to the following snippet.

```
Specify the directory the json file will be saved: /path/to/json/files/
Specify the json file name: sample.json
Write new file or append to an existing file? [w/a] w
Read in all data files in a directory or individual data files? [d/f] f
Specify the data file to read from: /path/to/data/files/SS_01.tsv
Required information for /path/to/data/files/SS_01.tsv...
Name: SS_01
Species: medaka
Experiment type: SS
ID: 01
Is this the control group? [y/n] n
Horizontal dimension of the capture area (in mm): 200
Vertical dimension of the capture area (in mm): 100
Pixels to mm ratio: 2.4
Frame rate (per second): 20
Experiment starts at (in min): 0
Experiment ends at (in min): 10
Baseline segment starts at (in min): 0
Baseline segment ends at (in min): 2
Do you want to add another file? [y/n] n
Wrote the information entered into /path/to/json/files/sample.json
```

Using the Package

Once you import the locomotion package, you will need to first initiate animal objects using the **`locomotion.getAnimalObjs`** command, which returns a list of animal objects with basic X and Y data from the data files.

The routines for calculating Behavioral Distortion Distance (BDD) are located in the **trajectory.py** file and can be called by `locomotion.trajectory.[routine_name]`.

Example script:

```
import locomotion
info_file = "/path/to/animal_info.json"
animals = locomotion.getAnimalObjs( info_file )
for a in animals:
  locomotion.trajectory.getCurveData( a )
variables = ['Y','Velocity','Curvature']
start_time, end_time = 0, 1
norm_mode = 'spec'
distances = locomotion.trajectory.computeAllBDD( animals, variables, start_time, end_time,
norm_mode )
output_directory, outfile_name = "/path/to/outdir", "results"
sort_table, square_table = False, False
color_min, color_max = 0.1, 0.5
locomotion.write.postProcess( animals, distances, output_directory, outfile_name,
sort_table, square_table, color_min, color_max )
```

Alternately, you can use the **computeBDD.py** script and follow prompts to run comparisons among animals in a given info file by running `python computeBDD.py` in your terminal, which should run similar to this sample snippet.

```
Specify the path to the json file with animal information: /path/to/animal_info.json
Use all entries in the info file? [y/n] y
Which variables do you want to use? (e.g., 'Y Velocity Curvature') Y Velocity Curvature
Specify the start time of the segment you want to compare: 0
Specify the end time of the segment you want to compare: 1
Which time segment do you want to normalize over: the predetermined baseline or the segment
specified above? [b/s] b
Do you want to write the results into a file? [y/n] y
Specify the output directory: /path/to/outdir
Specify the output file name: results
Do you want to sort the output? [y/n] n
Do you want the distance table to be square instead of upper triangular? [y/n] n
```

To calculate the intra-individual variation in BDD for each animal in a specified info sheet, one can run a script like the following:

```python
import locomotion
info_file = "/path/to/animal_info.json"
animals = locomotion.getAnimalObjs( info_file )
for a in animals:
  locomotion.trajectory.getCurveData( a )
variables = ['Y','Velocity','Curvature']
norm_mode = 'spec'
number_of_comparisons_per_animal, specified_durations = 100, None
output_directory, outfile_name = "/path/to/outdir", "results"
start_time, end_time = 0, 1
locomotion.trajectory.runIndividualVariabilityTests( animals, variables, norm_mode,
number_of_comparisons_per_animal, specified_durations, output_directory, outfile_name,
start_time, end_time )
```

Alternately, you can use the **computeIndVar.py** script and follow prompts to run comparisons among animals in a given info file by running `python computeIndVar.py` in your terminal, which should run similar to this sample snippet.

```
Specify the path to the json file with animal information: /path/to/animal_info.json
Use all entries in the info file? [y/n] y
Which variables do you want to use? (e.g., 'Y Velocity Curvature') Y Velocity Curvature
Specify the start time of the overall segment in which you want to run comparisons: 0
Specify the end time of the overall segment in which you want to run comparisons: 1
Which time segment do you want to normalize over: the predetermined baseline or the segment
specified above? [b/s] b
Do you want to write the results into a file? [y/n] y
Specify the output directory: /path/to/outdir
Specify the output file name: results
```

The routines for calculating Conformal Spatiotemporal Distance (CSD) are located in the **heatmap.py** file and can be called by `locomotion.heatmap.[routine_name]`.

Example script:

```python
import locomotion
info_file = "/path/to/animal_info.json"
animals = locomotion.getAnimalObjs( info_file )
grid_size, start_time, end_time = 10, 0, 2
for a in animals:
  locomotion.trajectory.getSurfaceData( a, grid_size, start_time, end_time )
distances = locomotion.trajectory.computeAllCSD( animals )
output_directory, outfile_name = "/path/to/outdir", "results"
sort_table, square_table = False, False
color_min, color_max = 0.1, 0.5
locomotion.write.postProcess( animals, distances, output_directory, outfile_name,
sort_table, square_table, color_min, color_max )
```

Alternately, you can use the **computeCSD.py** script and follow prompts to run comparisons among animals in a given info file by running `python computeCSD.py` in your terminal, which should run similar to this sample snippet.

```
Specify the path to the json file with animal information: /path/to/animal_info.json
Use all entries in the info file? [y/n] y
Specify the start time of the segment you want to compare: 0
Specify the end time of the segment you want to compare: 1
Specify the grid size for the heat map (in the same units as the x- and y-dimensions): 10
Do you want to write the results into a file? [y/n] y
Specify the output directory: /path/to/outdir
Specify the output file name: results
Do you want to sort the output? [y/n] n
Do you want the distance table to be square instead of upper triangular? [y/n] n
```