ANIMAL TRACKING

This program is intended to track single animals (tested on mice and fish). It is also possible to use it for tracking multiple animals, but performance is suboptimal (use it only to get an overview of the group behavior). The program automatically extracts the position of an animal (its centroid) in a defined field of view. This can be done for the whole video or for a defined time window corresponding to an event of interest, such as stimulus delivery. Time of interest can be defined using visual or auditory landmarks, or simply by entering the time in seconds.

This program has been used to track mice in the following paper:
https://doi.org/10.1101/297226

Prerequisites

Matlab_R2013 or newer versions
For video creation, the QTWriter function: http://horchler.github.io/QTWriter/

Content

*analysis_videos.m: main script
*video_trace.m: to create a 30 seconds video of the detected animals
*timepoints_visual.m, timepoints_audio.m and timepoints_manual.m: to define window of stimulus presentation
*analysisofdata.m: to perform basics analysis

Input: video in the .mp4, .mov, .m4v, .avi format
Output:
* nameofvideo_roi.mat (related to section f)
* nameofvideo_background.mat (related to section g and h)
* nameofvideo_output.mat (related to section i)
optional:
* nameofvideo_toi.mat (related to section e)
* nameofvideo_sections.mat (related to section e and i)
* nameofvideo_output.mov (related to section c)

How to run the program

1) Open Matlab
2) Run "analysis_videos"

In the command window, a series of question will be asked:

a) Name of video?

Type in the name of your video with its extension in single quote (e.g: 'myvideo.mp4')

b) 'Single tracking (1) or Multitracking (2)?'

Select Multitracking if more than one animal is tracked.

For Single tracking:

* the 'manual detection' option is available:

If no animal is detected by the software in a frame, you have the option to manually click on the animal (semi-automated detection). A window with the video Frame will pop up. Otherwise, in the matrix storing the animal's centroid, a NaN will be stored.

c) '30 s video of detected animals?'

If you select yes, the animal's centroid depicted as a cross will be superimposed on 30s of your video.

d) 'Analysis?'

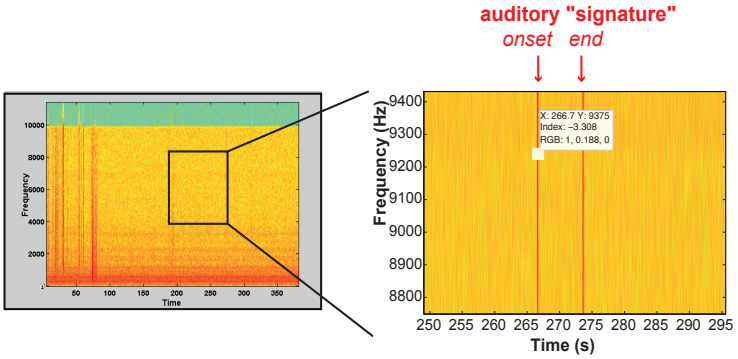
Performs basic analysis.

e) 'Define time window of interest'

Visual (1), auditory (2), manual (3) or whole video (4)?

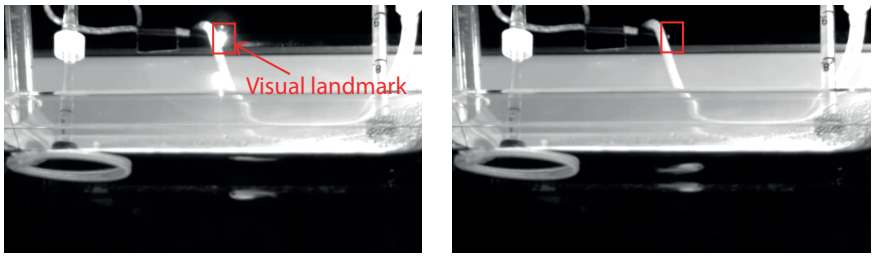
In case of visual/auditory landmarks or manual definition of the time window, the user has to indicate the number of stimuli `(nombrestimuli)` and the duration before `(befstim)` and after `(afstim)` stimulus onset.

*Auditory landmarks: the spectrogram of the audio file is displayed, as valve opening and closing have a characteristic ”signature”. The user then clicks on each stimuli onset.



In case of a stimuli with auditory landmarks, the spectrogram of the audio data is displayed. Here the opening and closing of valves controlling odor delivery have a characteristic ”signature”. The user clicks on each valve opening ”signature” to define the time of stimuli onset.

*Visual landmarks: one video frame is displayed. The user delimits the location in space where light intensity changes when the stimulus is delivered (LED ON when stimulus ON for example).



In case of stimuli with visual landmarks, the user delimits the region (in red) which changes when the stimulus is delivered.

*Manual detection: type in the beginning of the time window in seconds.

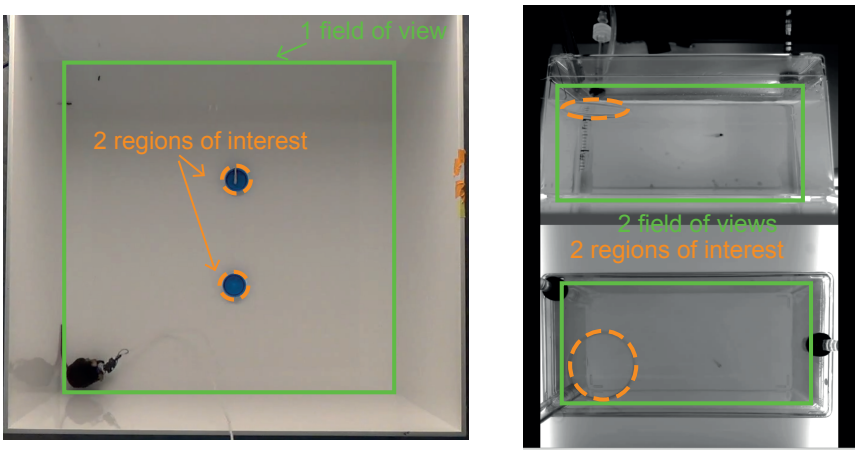
Saved variables:

`*befstim,afstim and nombrestimuli`

`*data: matrix (1xnombrestimuli) storing the time in sec of stimulus onsets.`

f) 'Define the field of view' and 'the region of interest'

The field of view is the field of view delimits the region within wich the animal can move.
The region of interest is a specific location in space of interest for further analysis.



Defining the field of views and region of interests

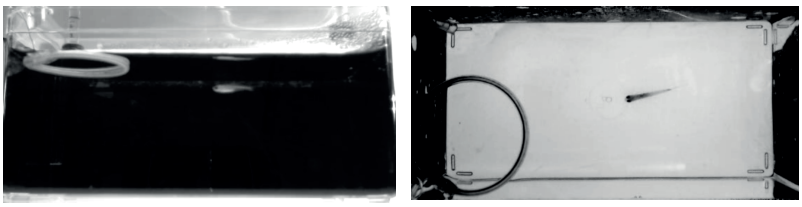
For the field of view and region of interest,

*type: rectangle or ellipse

For the field of view:

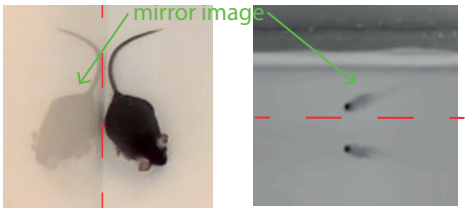
*its width and length (in cm) are asked for scaling purposes (`scale,` type in 0 if unknown).

*the user needs to indicate if the animal is brighter or darker than the background.



Left: The animal is brighter than the background.
Right: The fish is darker than the background.

Depending on the set-up, there can be some mirror effects and thus incorrect detection of the animal (mirror image instead of actual animal). We therefore advise to cautiously delimit the field of view.



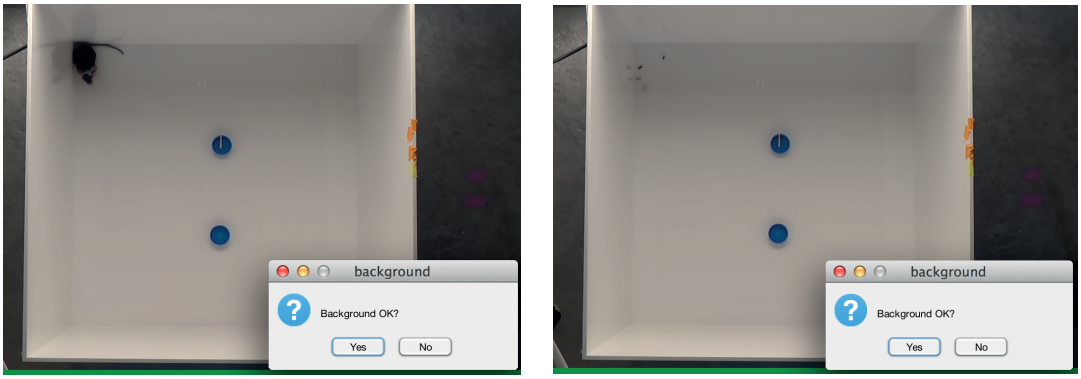
In case of reflective surfaces, field of views should be carefully drawn.

Saved variables:

```
*numb_roi and numb_roibeh: number of field of views and region of interests
*scale (numb_roi*2): length and width of each field of view in cm
*pos (2*2*numb_roi): [xmin ymin; xmax ymax]
*y_extrem (2*numb_roi): [ymin ymax]
*xy (dim*numb_roibeh): [vertices] if ellipse, [xmin ymin width height] if
rectangle
```

g) Background calculation

The program calculates the background image by taking for each pixel the median value over the first 30 seconds, considering 1 frame every 2 seconds (default mode). This works well in the majority of the cases because animals are usually in movement, and seldom stay still in one part of the region of interest. A window with the computed background will open and a pop-up window will ask if the background is OK. If it's not the case, the time over which the background is increased by 30 seconds, and the user will be asked again for confirmation. This will be reiterated until the background is OK.



Computing the background: on the left, incorrect background; on the right, correct one

Saved variables:

`*background: background image`

h) Setting the threshold (for each field of view)

After the background is defined, the background is subtracted from each frame of the video and the image is thresholded. The threshold value is defined by the user in the following way: The program uses an initial threshold value to generate a window with two video frames and their corresponding thresholded image. A pop-up window asks if the thresholded image is OK ('Yes'/'No'). An optimal thresholded image corresponds to the contour of the animal filled in white, and with minimal 'parasite' white objects. The program increments the threshold value until the user clicks on 'Yes'. Once a good thresholded image is obtained, the program will continue the iterations until the user finds the thresholded animal object too different from the original animal (contour, size etc...). In brief, the procedure stops when the user clicks on 'No' after having clicked once or more on 'Yes').

When used for multiple tracking, we recommend using a more stringent threshold value, where the thresholded animals only represents a portion of the original animals.

Left: two original video Frames are displayed. On the right, three examples of thresholded images after background subtraction. The threshold value of the left and right panels is suboptimal (mouse not clearly defined or fragmented, additional objects). The threshold value of the middle panel is optimal.

Saved variables:

```
*th_whole (1*numb_roi): for each field of view, threshold value
*m_area_r (1*numb_roi): for each field of view, median area of detected
animal (in pixels^2)
*length_r (1*numb_roi): for each field of view, median length of the
detected animal (in pixels)
```

i) Automated detection

The centroid of the animal in each frame is stored in the centroids_whole matrix (number of frames*2*number of animals*number of field of views).
In case of defined time windows (nameofvideo_sections.mat), the matrix indices corresponding to a new time window is saved in the fram_s variable. The matrix indices corresponding to the beginning of the stimulus (in case befstim is non-zero) is stored in fram_toi.

j) Analysis (if answered yes in d))

In construction...

Thanks to Nesibe Z. Temiz and Wilson M. Orostica for providing additional videos for testing the program.