Prepare: setting the video parameters e.g. dimensions, analysis arena

Detect setting video settings for analysis e.g. thresholds

Github.com/andrewpapale/odorTrails

All of

getStarttime1.m : permits user to designate frame start

Process VT1.m : takes raw optimouse data and removes what are considered bad coordinates, and generates 8 different variables x0,y0,nx0,ny0,V,nV,Nmm,mL

- drag, drop positions data from optimouse onto matlab workspace

In order to use the Process\_VT1 function, need to type in the following into command line: **[x0,y0,nx0,ny0,V,nV,Nm,mL]=Process\_VT1(position\_results,startFrame,arena\_data)** where “startFrame” is the frame number in which you want to begin the analysis e.g. time=0s

Once Process\_VT1 is run, matlab has data of positions from video’s optimouse “detect” file. This is stored in workspace. We can now compute data.

**Time spent on particular sides of chamber**

**[N,edges]=histcounts(x0,edges)**

to specify what boundaries to calculate the mouse positions, type following:

**edges = [0 x1 x2]** where x1 an x2 are the approximate boundaries positions of the chamber. You can find out by typing **plot(x0,y0,’.’)** and looking at where the mouse positions are located.

**Time spent on particular sides of chamber, ½ assay**

[x0,y0,nx0,ny0,V,nV,Nm,mL]=Process\_VT1(position\_results,startFrame,arena\_data)

frames = 1:length(x0);

frames = frames’;

N = histcounts2(x0,frames,[Ledge middle Redge],[1 length(frames)/2 length(frames)]) % boundaries determined via plot(x0,y0)

N = N/30 % convert from frames to seconds, assuming 30Hz framerate

% columns are 1/2s of assay; rows are left and right sides of chamber

**Determine distance traveled between two time points**

**getStartTime1;**

**[x0,y0,nx0,ny0,V,nV,Nm,mL]=Process\_VT1(position\_results,1,arena\_data);**

**>> D = ComputeDistanceTraveled(x0,y0) ;**

**>> timeins = linspace(1,length(x0)/(30),length(x0));**

**>> indx = find(timeins >= 48 & timeins <= 78);**

**D = ComputeDistanceTraveled(x0,y0) % type this!!**

**timeins = linspace(1,length(x0)/(30),length(x0)); % specify time in s**

**indx = find(timeins >= 60 & timeins <= 120); % get a linear index for times between 60s and 120s**

**nansum(D(indx))/(conversion factor) % gives total distance traveled in cm; %conversion factor can be found in arena\_data**

**H1 = nansum(H(:,indx),2); %ignore this for now**

frame = 1:length(x0); % specify the frame number

**Determine distance traveled across entire assay**

**D = ComputeDistanceTraveled(x0,y0) % type this!!**

% 2017-09-27 AndyP

% D = ComputeDistanceTraveled(x0,y0);

% compute distance traveled using body center of mass positions <x0,y0>.

nT = length(x0);

D = nan(nT,1);

for iT=2:nT

D(iT)=sqrt((x0(iT)-x0(iT-1)).^2+(y0(iT)-y0(iT-1)).^2);

end

**Distance traveled between defined areas**

**H = histcn(x0,[1 300 600],'AccumData',D,'fun',@nansum)**

**H = H./(conversion factor x10) % pixels / pixels/cm = cm %conversion factor can be found in arena\_data**

**Immobile times:**

[sumTimmobile,immobile,V] = getImmobileTimes(x0,y0,arena\_data,1,[1 300 600],60) where 1 = velocity threshold in cm/s, [1 300 600] are the regions being calculated, 60 = time interval in frames in which velocity is calculated less than 1

Type sumTimmobile

H = histcn(x0,[0 300 600],'AccumData',D,'fun',@nansum);   % calculate total distance traveled in pixels

% the [0 300 600] indicates brackets for where to calculate the distance traveled in an area

% the problem with this code is that the distance formula sqrt(x0.^2+y0.^2) does not take into account chronologically neighboring points

H = H./14.8 % pixels / pixels/cm = cm

Speed of movement (either averaged over some duration e.g. when laser is on for prolonged period or specific time points)

Plot graphs

Plot(x0,y0,’.’) generates plot of positions % ‘.’ Generates dots vs. ‘-‘ generates a smoothed line

H = histcn ([x0,y0],1:10;600, 1:10:400); %specifies 10 px bins over 400x600

Imagesc(1:10:600,1:10:400,H’) %H’ is transposes matrix

Things I want:

Locomotor data e.g. distance traveled, average speed, graphical tracking mouse movement

Heat map of time mouse spends in particular area

**Immobile times including body and nose:**

function [sumTimmobile,immobile,V,nV] = getImmobileTimes\_v2(x0,y0,nx0,ny0,arena\_data,**thresholdbody**,**thresholdnose**,**bins**,**timethreshold**) % fill in bold (see below for instructions what parameters these reflect)

% 2017-10-13 AndyP

% 2017-10-30 AndyP, modified to use second threshold based on nose position

% get times that the animal is immobile

% [sumTimmobile,immobile,V,nV] = getImmobileTimes(x0,y0,nx0,ny0,arena\_data,thresholdbody,thresholdnose,,bins,timethreshold)

% Example: [sumTimmobile,immobile,V] = getImmobileTimes(x0,y0,nx0,ny0,arena\_data,0.1,0.1,[1 300 600],60); indicates body movement <0.1cm/s, nose movement <0.1cm/s, boundaries of arena [1 300 600] over 60 frames (2 seconds) velocity is < 1cm

doTest = true;

m = 30;

d = 0.5;

postSmoothing = 1;

dT = 1/30;

conv\_factor = arena\_data.pixels\_per\_mm\*10;

dx = foaw\_diff(x0,dT,m,d,postSmoothing);

dy = foaw\_diff(y0,dT,m,d,postSmoothing);

V = sqrt(dx.^2+dy.^2)./conv\_factor; % cm/s

dnx = foaw\_diff(nx0,dT,m,d,postSmoothing);

dny = foaw\_diff(ny0,dT,m,d,postSmoothing);

nV = sqrt(dnx.^2+dny.^2)./conv\_factor; % cm/s

immobile = nan(size(V));

for iT=timethreshold:timethreshold:length(V)

    meanVpertime = nanmean(V(iT-timethreshold+1:iT));

    meannVpertime = nanmean(nV(iT-timethreshold+1:iT));

    if meanVpertime < thresholdbody & meannVpertime < thresholdnose %#ok<AND2>

        immobile(iT-timethreshold+1:iT)=1;

    else

        immobile(iT-timethreshold+1:iT)=0;

    end

end

sumTimmobile = histcn(x0,bins,'AccumData',immobile,'fun',@nansum);

sumTimmobile = sumTimmobile/30;

if doTest

   scatter(x0,y0,20,V);

   hold on;

   plot(x0(immobile==1),y0(immobile==1),'rx');

end

Binning code

nP = length(x0); % number of frames in session

nB = 30; % in seconds, bin session into 30s segments

H = histcn([x0,(1:length(x0))'],[0 300 600],linspace(1,nP,ceil(nP./(30\*nB)))); % assumes 30Hz frame rate for camera, assumes center of chamber is at 300 pixels and borders are at 0 and 600 pixels

H = H/30; % convert from frames to time in seconds, assumes 30Hz frame rate for camera

% Then,

% H(1,:) = 1x40 30s bins of times on left side of chamber

% H(2,:) = 1x40 30s bins of times on right side of chamber

**Heat maps:**

>> H = histogram2(x0,y0,linspace(0,700,100),linspace(0,600,80),'FaceColor','flat'); %scales for x0 (0-700) and y0 (0-600) and binning?

>> caxis([0 50]) %color axis scale, brackets scale of color e.g. 0-100 vs. 0-50

>> colormap hot % change color scheme to “hot” (red and yellow)

H0 = H.Values;

>> pcolor(smooth2a(H0',2,2)) %’ flips axis

>> shading flat % removes grid background to white

H0(H0==0)=nan % removes regions where mouse was not present

>> doc colormap % pulls up colormap guide

**Binning for locomotion, distance traveled between two specified time points**

t1 = 1; t2 = 200; plotrange = (round(t1\*30):round(t2\*30)); plot(x0(plotrange),y0(plotrange),'g-'); % plot first t1:1 to t2: 200s

% 30 refers to Hz of camera, g for green, ‘-‘

**Instantaneous velocity**

- [x0,y0,nx0,ny0,V,nV,Nmm,mL] = Process\_VT2(position\_results,1,arena\_data); % process 1 session to get V, body velocity at each timepoint, use updated Process\_VT1, attached

- threshold = 2.5\*nanmean(V); % set a threshold to get V, I chose 2.5 times the mean velocity

- Vhigh = nanmean(V(V>threshold)); % get the mean of velocity above the threshold

**Time to 1st instantaneous velocity**

-[x0,y0,nx0,ny0,V,nV,Nmm,mL] = Process\_VT2(position\_results,1,arena\_data); % process 1 session to get V, body velocity at each timepoint, use updated Process\_VT1, attached

- threshold = 2.5\*nanmean(V); % set a threshold to get V, I chose 2.5 times the mean velocity

- Vhigh = nanmean(V(V>threshold)); % get the mean of velocity above the threshold

**Optimouse lines:**

prepare\_arena\_batch; %runs prepare function in batch

calculate\_arena\_batch; %runs detect function in batch