Directional-tracking

Animal Heading Calculator

Purpose

Calculates average heading of moving animals using Spike2.

Required Input

This script requires x and y animal centroid coordinates obtained from an animal tracking software (e.g. ImagePro Plus, Media Cybernetics). Tab-delimited data with x and y values arranged in column fashion are acceptable.

Function

This script bins animal centroid tracks spatially and temporally to provide average headings over time. The script bins each track 20 times. Angle headings are calculated as a straight line from the first frame of the spatial bin to the last frame of the bin. Each angle is then placed into temporal bins from 10 minutes to 100 minutes based frame number in 10 minute intervals. The script generates an output log file of each spatially binned heading and its corresponding time bin.

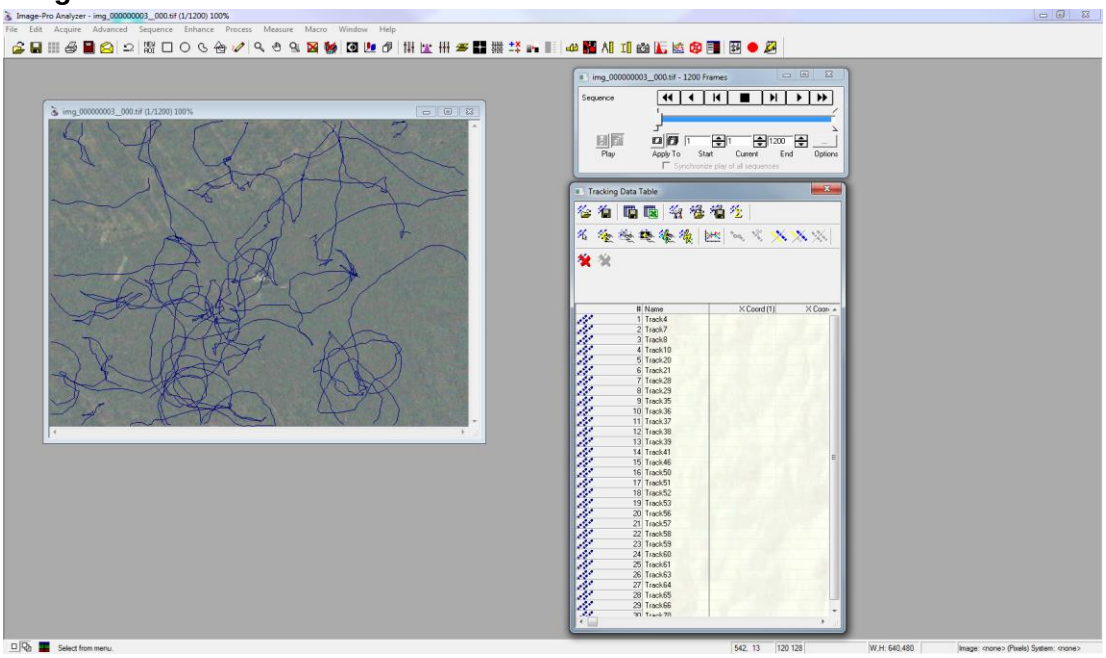
Output

Output consists of a text file with columns containing track bin number, angular heading, and corresponding time stamp for each of the twenty track bins of each track.

Post hoc analysis

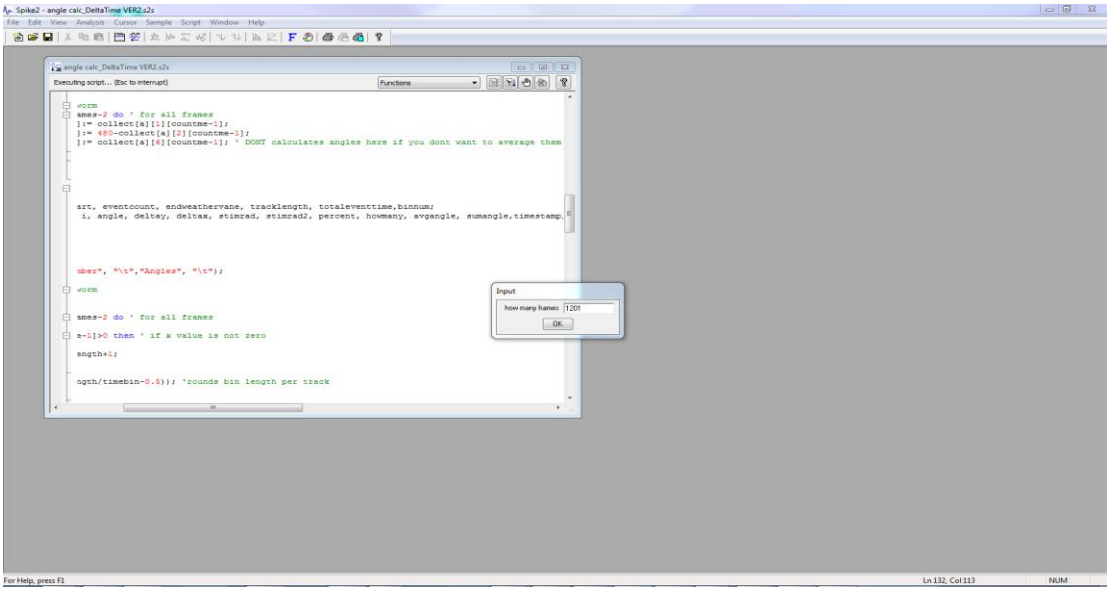
Post hoc analysis of heading data requires exporting log file output of the script into Microsoft Excel. Heading data is then sorted by time bin for each individual assay. Heading data for each time bin (mean angle) is then averaged across all animals within an assay using Matlab (Mathworks) circular statistics module. This results in each individual assay contributing one average mean heading per time bin. Averaged headings are then pooled across assays and sorted by their respective time bin so that there is one reported heading per time bin, per assay. Pooled data are then manually sorted into 10-30 minutes, 40-60 minutes, and 70-90 minute categories, according to their time bins. These categories are analyzed (Rayleigh test) and plotted using Matlab (Mathworks) circular statistics module.

ImagePro Plus v7. Was used to track the centroid of each worm

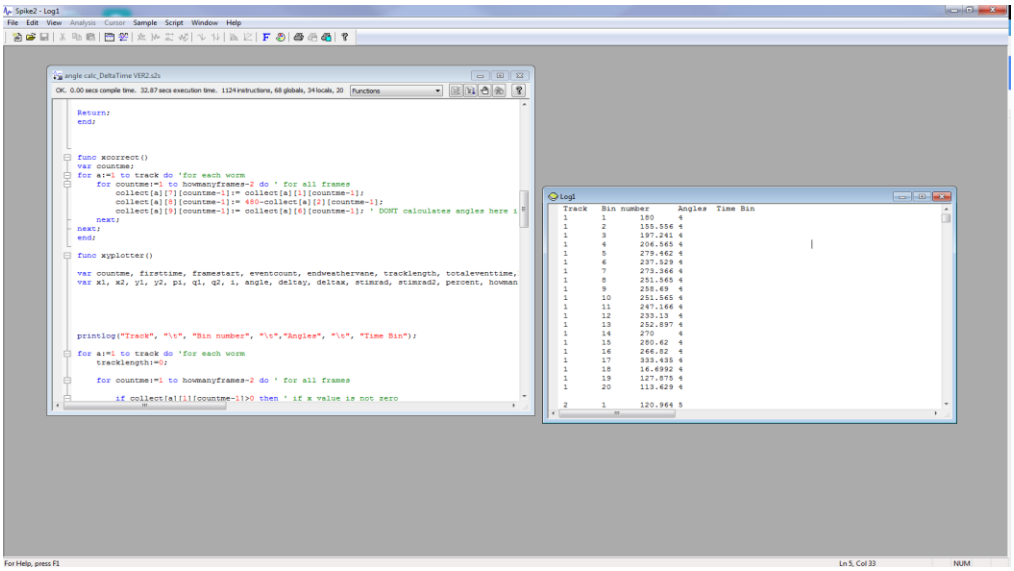


Tracking data (X, and Y coordinates) should be exported to Microsoft Excel by hitting the green “excel button”, and then saved as a tab delimited file from within Microsoft Excel. It is important to note the frame number of video being analyzed. Data from this tracking set is included in this submission.

Using Spike2 to analyze data

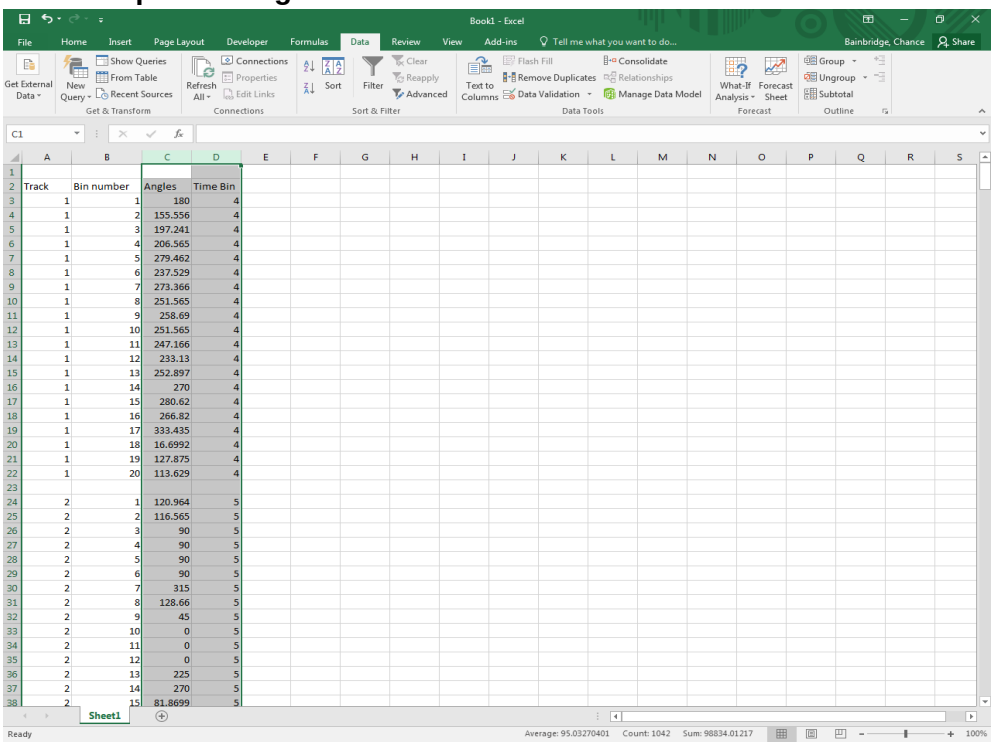


Run the script using the tab delimited files. Frame number should be reported as n+1 frames (e.g. if there are 1200 frames in a video, 1201 should be reported to the script.) When asked if you want to “plot in sequence” select “n” and hit OK.



Output consists of a text file with columns containing track bin number, angular heading, and corresponding time stamp for each of the twenty track bins of each track. This data should be exported to a new excel document for further post hoc analysis. A time bin of 1 equals first 10 minutes.

Post hoc processing



Track	Bin number	Angles	Time Bin
1	1	180	4
1	2	155.556	4
1	3	197.241	4
1	4	206.565	4
1	5	279.462	4
1	6	237.529	4
1	7	273.366	4
1	8	251.565	4
1	9	258.69	4
1	10	251.565	4
1	11	247.166	4
1	12	233.13	4
1	13	252.897	4
1	14	270	4
1	15	280.62	4
1	16	266.82	4
1	17	333.435	4
1	18	16.6992	4
1	19	127.875	4
1	20	113.629	4
2	1	120.964	5
2	2	116.565	5
2	3	90	5
2	4	90	5
2	5	90	5
2	6	90	5
2	7	315	5
2	8	128.66	5
2	9	45	5
2	10	0	5
2	11	0	5
2	12	0	5
2	13	225	5
2	14	270	5
2	15	81.8699	5

Data should be sorted by the time bin column. Once sorted, you can group the time bins easily by using the filter function in excel to organize by individual time bins and export to Matlab (Mathworks) for analysis and calculation of mean heading.

Script

```
var timebin;
timebin:=20;
var dummy[30000], countit, FileNam, line$, wert,a, blue, red, green;
var channel, logi, countmore, headerlines, minframenumber;
minframenumber:=4; ' smallest number of interior angles (inter-frames) for weathervane
countit:=0;
headerlines:=1; 'how many headerlines are in the ascii file?
var howmanyframes, collect[200][15][1201],track, smallesttrack, counter, yesno$; 'NOTE if track number changes, increase size of array to accomodate additional track data, (e.g. increase
var weathervanelength, countagain, printme$,i, startframe[200];
'USER INPUT
howmanyframes:=input("how many frames",1201); 'this could theoretically be detected by reading the header
yesno$:=Input$("plot in sequence?","n"); ' THIS IS NOT NECESSARY AND CAN BE IGNORED OR DELETED

'go to logfile and empty it
logi:=LogHandle();
View(logi);
EditSelectAll();
EditClear();
' end logfile

'call loadfile function (main function)
loadfile();
HALT;

'LOAD AND READ THE FILE - PUT DATA IN 3 DIMENSIONAL ARRAY
func loadfile()
' open a tab delimited data file with X and Y and angle values.
FileNam:=FileOpen("*.*", 8, 0,"Select the ASCii file");
if (FileNam > 0) then                ' if we opened a file...
   ReadSetup("","");            ' cancel soft separators
   ' while Read(line$) >= 0 do      ' read while not EOF or error. THIS WOULD BE FOR THE HEADER

   while Read(dummy) >= 0 do    ' read while not EOF or error, EOF = end of file
      countit:=countit+1; 'count the rows in the file
      if countit>headerlines then 'once beyond the header
         i:=0;
         track:=track+1; ' count worms
         for countmore:=1 to howmanyframes-1 do 'for each frame taken
```

```
            if dummy[countmore+1]>0 then ' if the x value of the data is not 0
                i:=i+1; if i=1 then startframe[track]:=countmore; endif;
              collect[track][0][COUNTER]:=countmore; 'worm number
              collect[track][1][COUNTER]:=dummy[countmore+1]; 'X
              collect[track][2][COUNTER]:=dummy[countmore+howmanyframes]; 'Y

collect[track][3][COUNTER]:=dummy[countmore+howmanyframes+howmanyframes-1]; 'angle

collect[track][5][COUNTER]:=dummy[countmore+howmanyframes+howmanyframes+howmanyf
rames-3]; 'distance

collect[track][6][COUNTER]:=dummy[countmore+howmanyframes+howmanyframes+howmanyf
rames+howmanyframes-5]; 'velocity
              COUNTER:=COUNTER+1; 'increase counter

              if yesno$<>"n" then 'if wanted, plot in sequence. THIS IS NOT NECESSARY AND
CAN BE IGNORED OR DELETED

Printlog(countmore,"\t",dummy[countmore+1],"\t",dummy[countmore+howmanyframes],
"\t",dummy[countmore+howmanyframes+howmanyframes-1]);
              endif;

            endif;
          next;

          COUNTER:=0; ' reset counter for next track / worm
       endif;
   wend;
   ReadSetup();              ' restore standard separators
   FileClose();              ' we are done with the file
else
   message("no file selected");
   halt;
endif;

'CALL THE ANALYSIS FUNCTIONS, in order
xcorrect();
xyplotter(); 'plot the data

Return;
end;
```

```
func xcorrect()
var countme;
for a:=1 to track do 'for each worm
   for countme:=1 to howmanyframes-2 do ' for all frames
      collect[a][7][countme-1]:= collect[a][1][countme-1];
      collect[a][8][countme-1]:= 480-collect[a][2][countme-1];
      collect[a][9][countme-1]:= collect[a][6][countme-1]; ' DONT calculates angles here if you dont want to average them
   next;
next;
end;

func xyplotter()

var countme, firsttime, framestart, eventcount, endweathervane, tracklength, totaleventtime,binnum;
var x1, x2, y1, y2, pi, q1, q2, i, angle, deltay, deltax, stimrad, stimrad2, percent, howmany, avgangle, sumangle,timestamp, time;




printlog("Track", "\t", "Bin number", "\t","Angles", "\t", "Time Bin");

for a:=1 to track do 'for each worm
   tracklength:=0;

   for countme:=1 to howmanyframes-2 do ' for all frames

      if collect[a][1][countme-1]>0 then ' if x value is not zero

         tracklength:=tracklength+1;

      endif;
      percent:=round((tracklength/timebin-0.5)); 'rounds bin length per track

   next;
 '   if tracklength>timebin then
   ' endif;
   countme:=1;

   if tracklength >timebin then
      'i:=1;
```

```
repeat
   'calculates angle headings from x and y data while binning with respect to time.

   if collect[a][1][countme-1]>0 then ' if x value is not zero
      x1:=collect[a][7][countme-1]; '= first x value
      x2:= collect[a][7][countme-1+percent]; ' = last x value
      y1:=collect[a][8][countme-1]; '= first y value
      y2:=collect[a][8][countme-1+percent]; '= last y value
      pi:=3.14159265359;
      deltax:= x1-x2;
      deltay:= y1-y2;


      stimrad:= sqrt(pow(x1-x2,2)+pow(y1-y2,2)); 'calculates distance between x,y coordinates of track bin
      if deltay<>0 and deltax<>0 then 'prevents divide by 0
         q1:=(abs(deltay))/(abs(deltax));
         q2:=(abs(deltax))/(abs(deltay));
      endif;
      if deltay=0 and deltax>0 then angle:=180; ' RULES to adjust the angle to the frame of reference, moves to the left
      endif;
      if deltay=0 and deltax<0 then angle:=0; ' moves to the right
      endif;
      if deltax=0 and deltay>0 then angle:=90; ' moves down
      endif;
      if deltax=0 and deltay<0 then angle:=270; ' moves up
      endif;
      if deltax<0 and deltay<0 then angle:=270+(180/pi)*ATan(q2); ' moves up  and right
      endif;
      if deltax>0 and deltay<0 then angle:=180+(180/pi)*ATan(q1); ' moves up and left
      endif;
      if deltax>0 and deltay>0 then angle:=90+(180/pi)*ATan(q2); ' moves down and left
      endif;
      if deltax<0 and deltay>0 then angle:=(180/pi)*Atan(q1); ' moves right and down
      endif;
      'collect[a][10][i-1]:= angle;
      countme:= countme+percent;
      binnum := binnum+1;
      'i:= i+1;
      howmany:= howmany+1;
      timestamp:=countme-percent+startframe[a];
      if timestamp>=0 and timestamp<120 then time:=1;  ' 10 minute bin as described by the frame number of the video. each frame is 5 seconds.
```

```
                endif;
            if timestamp>=120 and timestamp<240 then time:=2; ' 20 minute bin
                endif;
            if timestamp>=240 and timestamp< 360 then time:=3; '30 minute bin
                endif;
            if timestamp>=360 and timestamp<480 then time:=4; '40 minute bin
                endif;
            if timestamp>=480 and timestamp<600 then time:=5; '50 minute bin
                endif;
            if timestamp>=600 and timestamp<720 then time:=6; '60 minute bin
                endif;
            if timestamp>=720 and timestamp<840 then time:=7; '70 minute bin
                endif;
            if timestamp>=840 and timestamp<960 then time:=8; '80 minute bin
                endif;
            if timestamp>=960 and timestamp<1080 then time:=9; '90 minute bin
                endif;
            if timestamp>=1080 and timestamp<1200 then time:=10; '100 minute bin
                endif;

            if howmany<timebin+1 then
                PrintLog(a,"\t", binnum,"\t", angle, "\t",time, "\t");
            endif;
        else
            countme:=countme+1;
        endif;

    until countme>= howmanyframes-2;
    binnum:=0;
    'tracklength:=0;
    howmany:=0;
    'i:=0;
  endif;
  printlog ("\n");
next;
' sumangle:=Arrsum(collect[a][10][0:20],avgangle);

' printlog("Average angle:", "\t",avgangle,"\t");

Return;
end;


func reorder() ' make meaningful Excel file
```

```
var doplot$, x$, y$, angle$, countme, anglecalc$;

'header for file
for a:=1 to track do 'for each worm
   doplot$:=doplot$+"\t"+"worm                                                                    "+STR$(a)+"
X"+"\t"+"Y"+"\t"+"angle"+"\t"+"calculated_angle"+"\t";    ' the \t is a tab
next;
PrintLog(doplot$+"\n");
doplot$:="";
''end header


for countme:=1 to howmanyframes do ' for all frames

   for a:=1 to track do 'for each worm
      x$:=STR$(collect[a][1][countme-1]); ' define x
      y$:=STR$(collect[a][2][countme-1]); ' define y
      angle$:=STR$(collect[a][3][countme-1]); ' define angle
      anglecalc$:=STR$(collect[a][4][countme-1]); ' define calculated angle
      if x$="0" and y$="0" then  ' if x or y are zero do not print (= make them empty)
         x$:=""; y$:=""; angle$:=""; anglecalc$:="";
      endif;
      doplot$:=doplot$+"\t"+x$+"\t"+y$+"\t"+angle$+"\t"+anglecalc$+"\t"; ' add data to xy plot
   next;


   if doplot$<>"" then 'do not plot empty lines
      PrintLog(doplot$+"\n");
   endif;

   doplot$:=""; 'reset string
next;

'Goto log and copy data
View(logi);
EditSelectAll();
EditCopy();
message("data has been copied");
Return;
end;
```