

گزارش پروژه

درس علوم اعصاب محاسباتي

نویسنده: حمیدرضا ابوئی

شماره دانشجویی: 402617509

استاد: دکتر دلیری

تیر ۱۴۰۳

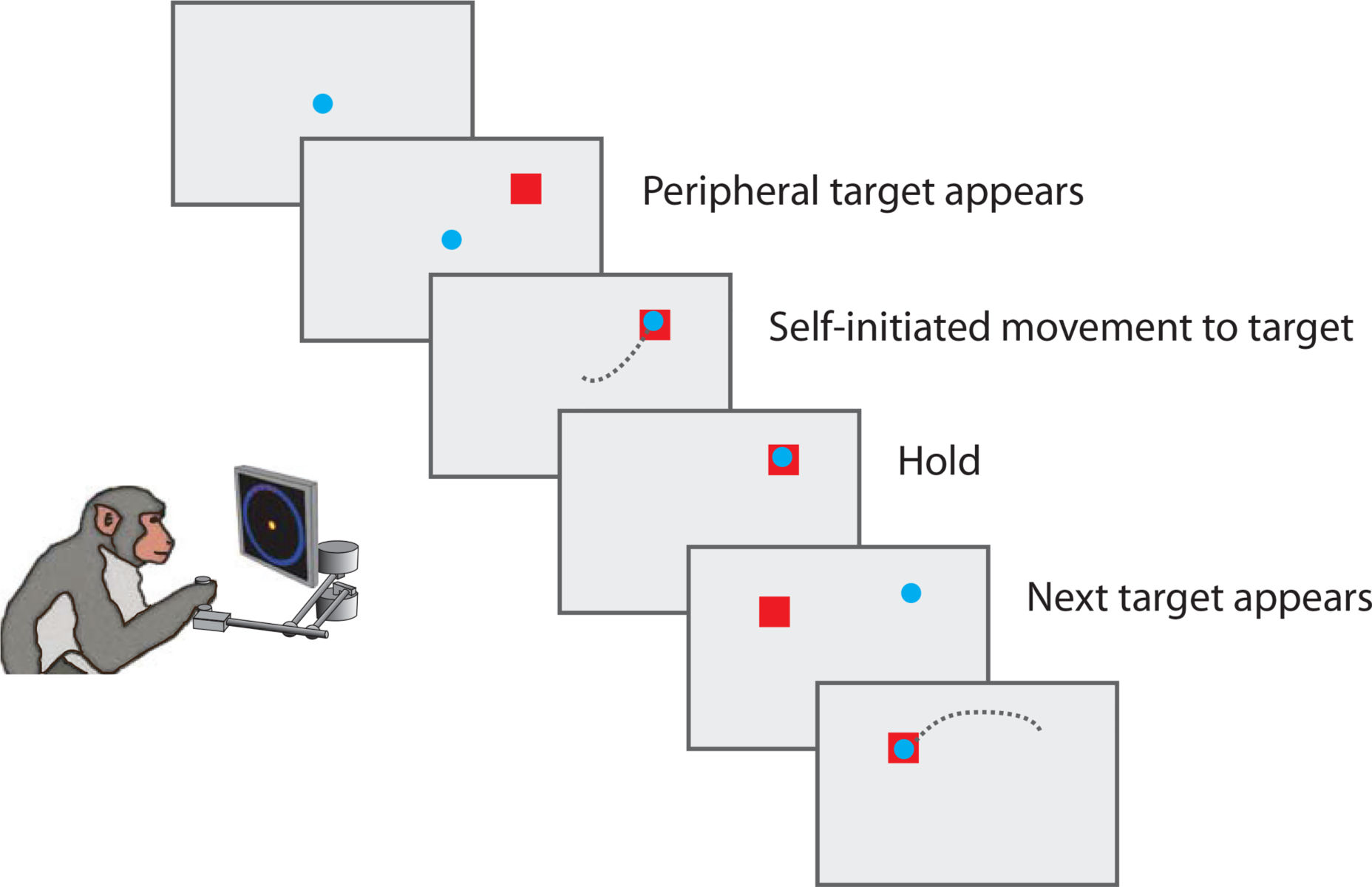
# توضیحات دیتاست:

این مجموعه داده شامل ضبط‌های خارج سلولی و رفتار ماکاک در آزمایش‌هایی است که نقش قشر پیش حرکتی[[1]](#footnote-1) (PMd) و قشر حرکتی اولیه[[2]](#footnote-2) (M1) را در یک کار رسیدن[[3]](#footnote-3) متوالی بررسی می‌کنند. در این آزمایش، میمون یک مکان نما روی صفحه را کنترل می کرد و برای جابجایی آن مکان نما به هدف دستیابی مشخص شده، با چندین هدف ارائه شده در هر آزمایش، پاداش دریافت کرد. حداقل نیازهای سینماتیکی برای حرکات رسیدن وجود داشت (مثلاً زمان‌های نگه‌داری بسیار کوتاه)، به این معنی که میمون معمولاً یک سری نسبتاً صاف از رسیدن را ایجاد می‌کرد. متغیرهای رفتاری ثبت شده شامل موقعیت، سرعت و شتاب هستند. دسترسی به مکان های هدف و زمان های تقریبی ارائه گنجانده شده است. ضبط‌های الکتروفیزیولوژیک با آرایه‌های چند الکترودی یوتا از دو میمون جمع‌آوری شد، در مجموع چهار جلسه. برای هر جلسه ده‌ها واحد کاملاً جدا شده از قشر حرکتی اولیه (M1) و قشر پیش حرکتی پشتی (PMd) وجود دارد.

# توضیح آزمایش

دو میمون (Monkey M, MM؛ Monkey T, MT) یک کار رسیدن را انجام دادند که در آن مکان نما کامپیوتر را با استفاده از حرکات بازو کنترل کردند (شکل ۱). میمون در حالی که یک دستکاری مسطح[[4]](#footnote-4) دو پیوندی را کار می کرد، روی یک صندلی نخستی نشسته بود. حرکات دست به یک صفحه افقی در فضای کاری 20 در 20 سانتی متر محدود می شود. در این کار، یک نشانه بصری روی صفحه (مربع 2 سانتی متر) مکان هدف را برای هر دستیابی مشخص می کرد و پس از انجام یک سری چهار تا رسیدن صحیح به اهداف، به میمون جایزه داده شد. مکان هدف به صورت شبه تصادفی انتخاب شد تا در داخل یک حلقه (شعاع = 5-15 سانتی متر، زاویه = 360 درجه) در مرکز هدف فعلی قرار گیرد.

زمان نمایش هدف برای اولین دستیابی هر آزمایش (آغاز سری، پس از استراحت) با رسیدن به دو تا چهار (وسیله میانی) متفاوت بود. برای اولین دستیابی، هدف ارائه شد و به میمون اجازه داده شد بدون یک دستور دوره تاخیر حرکت کند. برای فاصله های دو تا چهار، میمون هدف بعدی را با نگه داشتن مکان نما برای مدت کوتاهی در یک جعبه 2 سانتی متری در 2 سانتی متری در مرکز هدف فعلی آغاز کرد. پس از رسیدن به هدف فعلی، هدف بعدی 100 میلی‌ثانیه بعد فعال شد (نکته: نمایش داده نمی‌شود). زمان ظاهر شدن هدف بعدی دقیقاً ثبت نشد، اما با آزمایش‌های بعدی مشخص شد که به طور متوسط ​​96 میلی‌ثانیه پس از شروع به کار ظاهر می‌شود. بنابراین، به طور متوسط، هدف بعدی تقریباً 200 میلی ثانیه پس از رسیدن میمون به هدف فعلی ظاهر شد. علاوه بر این، یک دوره توقف تحمیلی 100 میلی ثانیه وجود داشت که همزمان با شروع به هدف بعدی آغاز شد. بنابراین، بین زمانی که میمون به هدف فعلی رسید و زمانی که اجازه داشت حرکت بعدی را آغاز کند، یک فاصله زمانی 200 میلی ثانیه وجود داشت. این دوره کوتاه نگهداشتن باعث شد که میمون‌ها با نزدیک شدن به هدف، سرعت خود را کاهش دهند. در عمل، این کار منجر به یک سری حرکات نسبتاً نرم بازو با فاصله متغیر شد.



شکل 1 شماتیک آزمایش رسیدن

فایل‌های زیر برای این پروژه استفاده شده است. قابل ذکر است که اطلاعات قشر حرکتی M1 فقط در مورد اول قابل دسترس است و در نشست‌های دیگر موجود نیست بنابراین در پردازش‌ها باید این نکته در نظر گرفته می‌شد.

“Raw” files

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| File name | Monkey | # trials | # neurons, M1 | # neurons, PMd |
| MM\_S1\_raw.mat | MM | 140 | 67 | 94 |
| MT\_S1\_raw.mat | MT | 116 | 0 | 49 |
| MT\_S2\_raw.mat | MT | 178 | 0 | 46 |
| MT\_S3\_raw.mat | MT | 174 | 0 | 57 |

# پیاده‌سازی پروژه

کد به دو قسمت تقسیم می‌شود. بخش اول نشست‌های مختلف (۴ نشست دو میمون) را برای خواندن آماده سازی می‌کند و کد دوم تمام مراحل پردازش را در بر دارد.

بخش اول به شرح زیر است:

clc;

clear;

close all;

%% Start reading data1

main\_file\_reader\_flag = 1;

fname = 'MM\_S1\_raw.mat'; % Monkey M (MM), session 1; PMd and M1

code1

clear;

main\_file\_reader\_flag = 1;

fname = 'MT\_S1\_raw.mat'; % Monkey T (MT), session 1; PMd only

code1

clear;

main\_file\_reader\_flag = 1;

fname = 'MT\_S2\_raw.mat'; % Monkey T (MT), session 2; PMd only

code1

clear;

main\_file\_reader\_flag = 1;

fname = 'MT\_S3\_raw.mat'; % Monkey T (MT), session 3; PMd only

code1

clear main\_file\_reader\_flag

کد‌های فایل اصلی به شرح زیر است:

is\_main\_reader\_executed = exist("main\_file\_reader\_flag","var");

if ~is\_main\_reader\_executed

clc

clear;

close all;

fname = 'MM\_S1\_raw.mat'; % Monkey M (MM), session 1; PMd and M1

end

%% Load data

fpath = './../data/data\_and\_scripts/source\_data/raw/';

ipath = './../images/';

% fname = 'MM\_S1\_raw.mat'; % Monkey M (MM), session 1; PMd and M1

% fname = 'MT\_S1\_raw.mat'; % Monkey T (MT), session 1; PMd only

% fname = 'MT\_S2\_raw.mat'; % Monkey T (MT), session 2; PMd only

% fname = 'MT\_S3\_raw.mat'; % Monkey T (MT), session 3; PMd only

if (strcmp(fname,'MM\_S1\_raw.mat'))

img\_header\_name = "MM\_S1\_";

elseif (strcmp(fname,'MT\_S1\_raw.mat'))

img\_header\_name = "MT\_S1\_";

elseif (strcmp(fname,'MT\_S2\_raw.mat'))

img\_header\_name = "MT\_S2\_";

else

img\_header\_name = "MT\_S3\_";

end

load([fpath fname])

alldays.tt = trial\_table;

% Check for existence of M1 data - only one animal (MM) has it

M1\_present = exist('M1','var');

%% User inputs

dt = .01; % in seconds. .01 = 10ms, .02 = 20ms, etc

time\_end\_extend = 0.4; % in seconds; time after end of reach to continue grabbing data

time\_before\_extend = -0.3; % in seconds; time before reach start to start grabbing data; needs to be negative to go backwards

%% Initialize

blocks = 1; % there is only one block in this data set

correct\_trials\_only = 0; % All included trials are correct by default; this is legacy

col\_start\_time = 1; % column # in the table with experimental data for time of trial start; user shouldn't change this

col\_end\_time = 22; % column # in the table with experimental data for time of trial end; user shouldn't change this

bl\_trials{1} = ones(size(trial\_table,1),1);

% Initialize M1\_units and PMd\_units

PMd\_units = cell(length(PMd.units),1);

if M1\_present

M1\_units = cell(length(M1.units),1);

end

%% Extract data

% For PMd units

for i = 1:length(PMd.units)

PMd\_units{i} = PMd.units(i).ts;

end

% For M1 units

if M1\_present

for i = 1:length(M1.units)

M1\_units{i} = M1.units(i).ts;

end

end

kin.pos(:,1) = cont.t;

kin.pos(:,2:3) = cont.pos;

kin.vel(:,1) = cont.t;

kin.vel(:,2:3) = cont.vel;

kin.acc(:,1) = cont.t;

kin.acc(:,2:3) = cont.acc;

% Extract neural data

tic

num\_units\_PMd = length(PMd\_units);

if M1\_present

num\_units\_M1 = length(M1\_units);

end

max\_spk\_time = max(kin.pos(:,1)); % Set max timestamp to collect spikes

edges = 0:dt:(max\_spk\_time + dt); % Defines edges for histogram; used for binning data later

% Initialize variables that will contain spikes

neural\_data\_temp\_PMd = nan(num\_units\_PMd,length(edges)-1);

if M1\_present

neural\_data\_temp\_M1 = nan(num\_units\_M1,length(edges)-1);

end

% Bin data using histcounts()

for nrn\_num = 1:num\_units\_PMd

neural\_data\_temp\_PMd(nrn\_num,:) = histcounts(PMd\_units{nrn\_num},edges);

disp(['PMd unit: ' num2str(nrn\_num) '/' num2str(num\_units\_PMd)])

end

if M1\_present

for nrn\_num = 1:num\_units\_M1

neural\_data\_temp\_M1(nrn\_num,:) = histcounts(M1\_units{nrn\_num},edges);

disp(['M1 unit: ' num2str(nrn\_num) '/' num2str(num\_units\_M1)])

end

end

% IMPORTANT:

% Spikes collected from 0 ms onward; Kinematics from 1000ms onward

% This gets rid of the spikes before kinematics are collected

% I.e., with 10ms bins, 1000ms = 100 bins \* 10ms/bin

num\_bins\_discard = round(1/dt); % number of bins to discard before kinematics are colletcted

bin\_start = num\_bins\_discard + 1; % bin to start keeping track of data

neural\_data\_temp\_PMd = neural\_data\_temp\_PMd(:,bin\_start:end);

if M1\_present

neural\_data\_temp\_M1 = neural\_data\_temp\_M1(:,bin\_start:end);

end

disp(['Neural data extracted: ' num2str(toc) ' seconds'])

% Extract kinematic data

tic

kin\_ts = downsample(kin.pos(:,1),round(1000\*dt)); % This doesn't need to be filtered (it's just timestamps)

% Decimate (smooth+downsample, rather than just downsample)

x\_pos = decimate(kin.pos(:,2),round(1000\*dt)); % Sampling rate of kinematics is 1 kHz. thus, decimate so bin size is same as dt

y\_pos = decimate(kin.pos(:,3),round(1000\*dt));

x\_vel = decimate(kin.vel(:,2),round(1000\*dt));

y\_vel = decimate(kin.vel(:,3),round(1000\*dt));

x\_acc = decimate(kin.acc(:,2),round(1000\*dt));

y\_acc = decimate(kin.acc(:,3),round(1000\*dt));

disp(['Kinematic data extracted: ' num2str(toc) ' seconds'])

% Make timestamps

ts = edges(1:(end-1));

ts = ts(bin\_start:end); % Discard first second as above

%% Arrange data by \*trial\*

clear Data; clear Data2; clear Data\_trials

for block\_idx = 1:length(blocks)

block\_num = blocks(block\_idx);

trials = find(bl\_trials{block\_num}==1);

% Initialize

num\_trials = sum(bl\_trials{block\_num});

Data(block\_num).kinematics = cell(num\_trials,1);

Data(block\_num).neural\_data\_M1 = cell(num\_trials,1);

Data(block\_num).neural\_data\_PMd = cell(num\_trials,1);

Data(block\_num).block\_info = alldays(block\_num).tt(trials,:);

Data(block\_num).trials = bl\_trials{block\_num};

num\_units\_PMd = length(PMd\_units);

if M1\_present

num\_units\_M1 = length(M1\_units);

end

% Arrange data by trial

for trial\_idx = 1:length(trials)

trial\_num = trials(trial\_idx);

disp(['Writing data for trial: ' num2str(trial\_num)])

tr\_start = alldays(block\_num).tt(trial\_num,col\_start\_time) + time\_before\_extend; % Grab data before trial start

tr\_end = alldays(block\_num).tt(trial\_num,col\_end\_time) + time\_end\_extend; % Grab data after trial end

tr\_bins = logical((ts>=tr\_start).\*(ts<=tr\_end)); % Find bins between tr\_start and tr\_end

% Take neural data from between tr\_start and tr\_end

Data(block\_num).neural\_data\_PMd{trial\_idx} = neural\_data\_temp\_PMd(:,tr\_bins);

if M1\_present

Data(block\_num).neural\_data\_M1{trial\_idx} = neural\_data\_temp\_M1(:,tr\_bins);

end

% Take kinematics from between tr\_start and tr\_end

Data(block\_num).kinematics{trial\_idx} = [x\_pos(tr\_bins), y\_pos(tr\_bins), ...

x\_vel(tr\_bins), y\_vel(tr\_bins), ...

x\_acc(tr\_bins), y\_acc(tr\_bins), ...

kin\_ts(tr\_bins)];

% Get timestamps (imposed by me) to make sure they match those of

% the kinematics

Data(block\_num).timestamps{trial\_idx,1} = ts(tr\_bins)';

end

end

Data\_trials = Data; % rename

%% Arrange data by \*reach\* - data has multiple reaches per trial; break trials up

% More user inputs

min\_reach\_len = 2; % in cm

time\_before\_cue = -0.3; % Amount of time before target comes on to grab data (in sec)

max\_reach\_time = round(1.4/dt) + ceil(abs(time\_before\_cue)/dt); % Max time for reach, in bins

spd\_thresh = 8; % in cm/sec; This is different from the speed threshold in visualize\_data.m; this helps to define the end of a reach

buff = 0.3; % Velocity often non-zero when cue comes on. Look forward at least this much to find end of reach (in sec)

end\_buff = 0.3; % Allows reach end to be a little later than the official end of trial. Just to be a little more permissive (larger data window)

pd\_lag = 0.096; % Photodetector wasn't used, so "cue on" is the command signal, not the detection signal. \*Average\* lag in Miller lab is 96 ms. Exact lag varies from trial to trial. See data description document for more information.

% Initialization

% Data2 = Data;

Data2 = struct;

idx = 1;

for tr = 1:num\_trials

reaches = find(~isnan(Data.block\_info(tr,[3 8 13 18]))); % Find successful reaches in this trial. Unsuccessful reach will have a nan in corresponding column 3 (reach 1), 8 (2), 13 (3), 18 (4)

num\_reach = length(reaches);

tr\_end = Data.block\_info(tr,col\_end\_time);

ts = Data.kinematics{tr}(:,end); % Based upon kinematic time stamps

x\_vel = Data.kinematics{tr}(:,3);

y\_vel = Data.kinematics{tr}(:,4);

x\_pos = Data.kinematics{tr}(:,1);

y\_pos = Data.kinematics{tr}(:,2);

spd = sqrt(x\_vel.^2 + y\_vel.^2);

for reach\_idx = 1:num\_reach

reach = reaches(reach\_idx);

idx\_cue\_on = 2 + 5\*(reach - 1); % Find column in trial table which denotes time of target appearance

% If reach wasn't completed or was invalid for some reason

% (Columns referenced here will be NaN if reach is invalid)

if isnan(Data.block\_info(tr,idx\_cue\_on+1)) || isnan(Data.block\_info(tr,idx\_cue\_on+2))

continue

end

% Correct for command signal lag

cue\_on = Data.block\_info(tr,idx\_cue\_on);

cue\_on = cue\_on + pd\_lag;

% Correction: for reaches 2-4, target is displayed 100ms before

% time in trial table

if reach > 1

cue\_on = cue\_on - .1; % Get the actual time from trial table; subtract 100 ms for correction

end

wind\_st = cue\_on + time\_before\_cue; % When the data window starts; grabs data before target appears

[~,idx\_cue\_on2] = min(abs(ts-cue\_on)); % Find time bin when cue comes on

% Determine end time for each reach

if reach < 4 % For reaches 1-3 on a given trial; these will be followed by another reach

% Find time of: min velocity before next go cue

idx\_cue\_on\_next = 2 + 5\*(reach);

cue\_on\_next = Data.block\_info(tr,idx\_cue\_on\_next);

% For reach to end: (this is permissive)

% 1) slow (falls below speed threshold), and

% 2) a certain minimum time after cue onset must have elapsed (buff)

% 3) end has to be before the next reach starts plus buff

cond\_reach\_end = logical((spd < spd\_thresh).\*(ts > (cue\_on + buff)).\*(ts < (cue\_on\_next + buff)));

% If conditions not met, skip

if sum(cond\_reach\_end) == 0

continue

end

% Within times meeting conditions, find one with minimum speed

cond\_reach\_end\_nan = 1.\*cond\_reach\_end; % Initialize and convert to scalar array

cond\_reach\_end\_nan(cond\_reach\_end\_nan < 1) = nan; % Make the bins not meeting conditions above nan

[~,idx\_reach\_end] = min(spd.\*cond\_reach\_end\_nan);

reach\_end = ts(idx\_reach\_end);

else

% Find time of min velocity after last reach. Conditions:

% 1) slow (falls below speed threshold), and

% 2) a certain minimum time after cue onset must have elapsed (buff)

% 3) end has to be before the trial ends plus buff

cond\_reach\_end = logical((spd < spd\_thresh).\*(ts > (cue\_on + buff)).\*(ts < (tr\_end + end\_buff)));

% If conditions not met, skip

if sum(cond\_reach\_end) == 0

continue

end

% Within times meeting conditions, find one with minimum speed

cond\_reach\_end\_nan = 1.\*cond\_reach\_end; % initialize and convert to scalar array

cond\_reach\_end\_nan(cond\_reach\_end\_nan < 1) = nan;

[~,idx\_reach\_end] = min(spd.\*cond\_reach\_end\_nan);

reach\_end = ts(idx\_reach\_end);

end

% Define window of time to save

wind\_reach = logical((ts>=wind\_st).\*(ts<=reach\_end));

% Add meta-data

Data2.trial\_num{idx,1} = tr;

Data2.reach\_num{idx,1} = reach;

Data2.reach\_st{idx,1} = cue\_on; % Time of cue on used as a proxy for when reach starts. It's approximate.

Data2.cue\_on{idx,1} = cue\_on;

Data2.reach\_end{idx,1} = reach\_end;

Data2.reach\_pos\_st{idx,1} = Data.kinematics{tr}(idx\_cue\_on2,1:2);

Data2.reach\_pos\_end{idx,1} = Data.kinematics{tr}(idx\_reach\_end,1:2);

Data2.avgspeed{idx,1}=mean(spd);

delta\_pos = Data2.reach\_pos\_end{idx,1} - Data2.reach\_pos\_st{idx,1};

[Data2.reach\_dir{idx,1}, Data2.reach\_len{idx,1}] = cart2pol(delta\_pos(1),delta\_pos(2));

idx\_target\_on = 1 + ceil(abs(time\_before\_cue)/dt); % Target is on in bin 1 unless extra time before is added, in which case it's on after that extra time

temp = zeros(sum(wind\_reach),1); temp(idx\_target\_on) = 1;

Data2.target\_on{idx,1} = temp;

% Copy stuff

Data2.kinematics{idx,1} = Data.kinematics{tr}(wind\_reach,:);

Data2.neural\_data\_PMd{idx,1} = Data.neural\_data\_PMd{tr}(:,wind\_reach);

if M1\_present

Data2.neural\_data\_M1{idx,1} = Data.neural\_data\_M1{tr}(:,wind\_reach);

end

Data2.block\_info = Data.block\_info;

Data2.time\_window{idx,1} = wind\_reach;

Data2.timestamps{idx,1} = Data.timestamps{tr}(wind\_reach);

% Exclude reach if it doesn't meet requirements

if (Data2.reach\_len{idx} < min\_reach\_len) || (sum(wind\_reach) > max\_reach\_time)

continue

end

% Increment reach index

idx = idx + 1;

end

end

% Rename variables

clear Data

Data = Data2; clear Data2

disp('Data cleaning completed.')

disp('Reach data located in variable: Data.')

disp('Trial data located in variable: Data\_trials.')

%% Show all routes

figure();

all\_kinematics=[];

x\_reach\_pos\_end = nan(length(Data.kinematics));

y\_reach\_pos\_end = nan(length(Data.kinematics));

for tr = 1:length(Data.kinematics)

all\_kinematics = [all\_kinematics; Data.kinematics{tr,:}];

a = Data.reach\_pos\_end{tr,:};

x\_reach\_pos\_end(tr) = a(1);

y\_reach\_pos\_end(tr) = a(2);

end

x = all\_kinematics(:,1);

y = all\_kinematics(:,2);

plot(x,y);

title("All routes monkey went and reach destinations");

xlabel("x");

ylabel("y");

hold on;

scatter(x\_reach\_pos\_end,y\_reach\_pos\_end);

legend("animal pos","reach pos")

saveas(gcf,strcat(ipath,img\_header\_name,"all\_routes\_pos.png"))

%% Show Some data

figure();

subplot(2,2,1);

% h=heatmap(Data.neural\_data\_M1{1, 1}, ...

% 'GridVisible','off');

% h.XData=Data.timestamps{1,1};

% h.XLabel = "time";

% h.YLabel = "channel";

if M1\_present

imshow(Data.neural\_data\_M1{1, 1})

xlabel("time");

ylabel("channel");

title("neural Data M1");

else

subplot(2,2,1);

scatter(Data.kinematics{1, 1}(:,5),Data.kinematics{1, 1}(:,6))

title("kinematic Acceleration")

xlabel("x");

ylabel("y");

end

subplot(2,2,2);

scatter(Data.kinematics{1, 1}(:,1),Data.kinematics{1, 1}(:,2))

title("kinematic position")

xlabel("x");

ylabel("y");

hold on;

scatter(Data.reach\_pos\_st{1,1}(1),Data.reach\_pos\_st{1,1}(2),'black');

scatter(Data.reach\_pos\_end{1,1}(1),Data.reach\_pos\_end{1,1}(2),'red');

subplot(2,2,3);

imshow(Data.neural\_data\_PMd{1, 1})

xlabel("time");

ylabel("channel");

title("neural Data PMd");

subplot(2,2,4);

plot(Data.kinematics{1, 1}(:,3),Data.kinematics{1, 1}(:,4));

title("Kinematic Velocity")

xlabel("x");

ylabel("y");

saveas(gcf,strcat(ipath,img\_header\_name,"overview\_1.png"))

trial\_num\_display = 161;

figure();

if M1\_present

subplot(2,3,1);

imshow(Data.neural\_data\_M1{trial\_num\_display, 1})

xlabel("time");

ylabel("channel");

title("neural Data M1");

subplot(2,3,2);

imshow(Data.neural\_data\_PMd{trial\_num\_display, 1})

xlabel("time");

ylabel("channel");

title("neural Data PMd");

subplot(2,3,3);

scatter(Data.kinematics{trial\_num\_display, 1}(:,1),Data.kinematics{trial\_num\_display, 1}(:,2))

title(strcat("kinematic position|",sprintf("dir=%.2f,len=%.2f",Data.reach\_dir{trial\_num\_display,1},Data.reach\_len{trial\_num\_display,1})))

xlabel("x");

ylabel("y");

hold on;

scatter(Data.reach\_pos\_st{trial\_num\_display,1}(1),Data.reach\_pos\_st{trial\_num\_display,1}(2),'black');

scatter(Data.reach\_pos\_end{trial\_num\_display,1}(1),Data.reach\_pos\_end{trial\_num\_display,1}(2),'red');

subplot(2,3,4);

plot(movmean(sum(Data.neural\_data\_M1{trial\_num\_display, 1}),15));

xlabel("time");

% ylabel("channel");

title("neural Data M1 firing rate");

subplot(2,3,5);

plot(movmean(sum(Data.neural\_data\_PMd{trial\_num\_display, 1}),15))

xlabel("time");

% ylabel("channel");

title("neural Data PMd firing rate");

subplot(2,3,6);

plot(Data.kinematics{trial\_num\_display, 1}(:,3),Data.kinematics{trial\_num\_display, 1}(:,4));

title("Kinematic Velocity")

xlabel("x");

ylabel("y");

else

subplot(2,2,1);

imshow(Data.neural\_data\_PMd{trial\_num\_display, 1})

xlabel("time");

ylabel("channel");

title("neural Data PMd");

subplot(2,2,2);

scatter(Data.kinematics{trial\_num\_display, 1}(:,1),Data.kinematics{trial\_num\_display, 1}(:,2))

title(strcat("kinematic position|",sprintf("dir=%.2f,len=%.2f",Data.reach\_dir{trial\_num\_display,1},Data.reach\_len{trial\_num\_display,1})))

xlabel("x");

ylabel("y");

hold on;

scatter(Data.reach\_pos\_st{trial\_num\_display,1}(1),Data.reach\_pos\_st{trial\_num\_display,1}(2),'black');

scatter(Data.reach\_pos\_end{trial\_num\_display,1}(1),Data.reach\_pos\_end{trial\_num\_display,1}(2),'red');

subplot(2,2,3);

plot(movmean(sum(Data.neural\_data\_PMd{trial\_num\_display, 1}),15))

xlabel("time");

% ylabel("channel");

title("neural Data PMd firing rate");

subplot(2,2,4);

plot(Data.kinematics{trial\_num\_display, 1}(:,3),Data.kinematics{trial\_num\_display, 1}(:,4));

title("Kinematic Velocity")

xlabel("x");

ylabel("y");

end

saveas(gcf,strcat(ipath,img\_header\_name,"overview\_",int2str(trial\_num\_display),".png"))

% %% Try fitting firing rate with

% trial\_num\_Processing = 182;

% fr\_m1 = movmean(sum(Data.neural\_data\_PMd{trial\_num\_Processing, 1}),5);

% fr\_PMd = movmean(sum(Data.neural\_data\_PMd{trial\_num\_Processing, 1}),5);

% xpos = Data.kinematics{trial\_num\_Processing, 1}(:,1);

% ypos = Data.kinematics{trial\_num\_Processing, 1}(:,2);

%

% linearCoef = polyfit(fr\_m1,xpos,3);

% linearFit = polyval(linearCoef,fr\_m1);

% figure()

% plot(fr\_m1,xpos,'s', fr\_m1,linearFit,'r-')

% xlabel('neural\_m1'); ylabel('x\_pos');

%% Show Direction and length

figure();

polarscatter([Data.reach\_dir{:}],[Data.reach\_len{:}]);

title("Direction and length")

saveas(gcf,strcat(ipath,img\_header\_name,"polar\_dir\_len.png"))

%% calculate all firings in each trials and show tuning curve

% Data2 = struct;

%

% idx = 1;

% for tr = 1:num\_trials

% reaches = find(~isnan(Data.block\_info(tr,[3 8 13 18]))); % Find successful reaches in this trial. Unsuccessful reach will have a nan in corresponding column 3 (reach 1), 8 (2), 13 (3), 18 (4)

% num\_reach = length(reaches);

% tr\_end = Data.block\_info(tr,col\_end\_time);

%

% ts = Data.kinematics{tr}(:,end); % Based upon kinematic time stamps

%

% x\_vel = Data.kinematics{tr}(:,3);

% y\_vel = Data.kinematics{tr}(:,4);

% x\_pos = Data.kinematics{tr}(:,1);

% y\_pos = Data.kinematics{tr}(:,2);

% x\_acc = Data.kinematics{tr}(:,5);

% y\_acc = Data.kinematics{tr}(:,6);

%

% spd = sqrt(x\_vel.^2 + y\_vel.^2);

%

% for reach\_idx = 1:num\_reach

% disp(idx);

% Data2.average\_firing\_rate\_PMd{idx,1} = mean(mean((Data.neural\_data\_PMd{idx,1})));

% Data2.average\_firing\_rate\_M1{idx,1} = mean(mean((Data.neural\_data\_M1{idx,1})));

%

%

% % Exclude reach if it doesn't meet requirements

% if (Data.reach\_len{idx} < min\_reach\_len) || (sum(wind\_reach) > max\_reach\_time)

% continue

% end

% idx = idx + 1;

% end

%

%

% end

for trial = 1:length(Data.time\_window)

Data.average\_firing\_rate\_PMd{trial,1} = (mean((Data.neural\_data\_PMd{trial,1}),2));

if M1\_present

Data.average\_firing\_rate\_M1{trial,1} = (mean((Data.neural\_data\_M1{trial,1}),2));

end

end

%% Plot Tuning curves

% select\_neuron = 13;

select\_neuron = nan;

if (strcmp(fname,'MM\_S1\_raw.mat'))

selected\_data\_to\_plot=105;

selected\_data\_to\_plot2=276;

[~,minI] = min(mean([Data.average\_firing\_rate\_PMd{:}]));

[~,maxI] = max(mean([Data.average\_firing\_rate\_PMd{:}]));

selected\_data\_to\_plot3 = minI;

selected\_data\_to\_plot4 = maxI;

[~,minI] = min(mean([Data.average\_firing\_rate\_M1{:}]));

[~,maxI] = max(mean([Data.average\_firing\_rate\_M1{:}]));

selected\_data\_to\_plot5 = minI;

selected\_data\_to\_plot6 = maxI;

elseif (strcmp(fname,'MT\_S1\_raw.mat'))

selected\_data\_to\_plot=171;

selected\_data\_to\_plot2=201;

elseif (strcmp(fname,'MT\_S2\_raw.mat'))

selected\_data\_to\_plot=309;

selected\_data\_to\_plot2=410;

else

selected\_data\_to\_plot = 91;

selected\_data\_to\_plot2 = 60;

end

if isnan(select\_neuron)

figure();

x = ([Data.reach\_dir{:}]);

y = mean([Data.average\_firing\_rate\_PMd{:}]);

f = fit(x.',y.','gauss1');

plot(f,x,y)

hold on;

plot(x(selected\_data\_to\_plot),y(selected\_data\_to\_plot),'o');

plot(x(selected\_data\_to\_plot2),y(selected\_data\_to\_plot2),'o');

if exist("selected\_data\_to\_plot3","var")

plot(x(selected\_data\_to\_plot3),y(selected\_data\_to\_plot3),'o');

plot(x(selected\_data\_to\_plot4),y(selected\_data\_to\_plot4),'o');

end

xlabel("reaching Direction");

ylabel("average firing rate")

title("tuning curve, firing rate PMd vs reach direction")

saveas(gcf,strcat(ipath,img\_header\_name,"fit\_gauss\_PMd\_tunning\_curve.png"))

if M1\_present

figure();

x = [Data.reach\_dir{:}];

y = mean([Data.average\_firing\_rate\_M1{:}]);

f = fit(x.',y.','gauss1');

plot(f,x,y)

hold on;

plot(x(selected\_data\_to\_plot),y(selected\_data\_to\_plot),'o');

plot(x(selected\_data\_to\_plot2),y(selected\_data\_to\_plot2),'o');

if exist("selected\_data\_to\_plot3","var")

plot(x(selected\_data\_to\_plot5),y(selected\_data\_to\_plot5),'o');

plot(x(selected\_data\_to\_plot6),y(selected\_data\_to\_plot6),'o');

end

xlabel("reaching Direction");

ylabel("average firing rate");

title("tuning curve, firing rate m1 vs reach direction");

saveas(gcf,strcat(ipath,img\_header\_name,"fit\_gauss\_M1\_tunning\_curve.png"))

end

figure();

if M1\_present

subplot(2,2,1);

imshow(Data.neural\_data\_M1{selected\_data\_to\_plot, 1});

xlabel("time");

ylabel("channel");

title(strcat("neural Data M1",sprintf(" |Dir%.2f",Data.reach\_dir{selected\_data\_to\_plot,1})));

subplot(2,2,2);

imshow(Data.neural\_data\_PMd{selected\_data\_to\_plot, 1})

xlabel("time");

ylabel("channel");

title("neural Data PMd");

subplot(2,2,3);

imshow(Data.neural\_data\_M1{selected\_data\_to\_plot2, 1})

xlabel("time");

ylabel("channel");

title(strcat("neural Data M1",sprintf(" |Dir%.2f",Data.reach\_dir{selected\_data\_to\_plot2,1})));

subplot(2,2,4);

imshow(Data.neural\_data\_PMd{selected\_data\_to\_plot2, 1})

xlabel("time");

ylabel("channel");

title("neural Data PMd");

saveas(gcf,strcat(ipath,img\_header\_name,"firing\_rate.png"))

figure();

subplot(2,2,1);

imshow(Data.neural\_data\_M1{selected\_data\_to\_plot5, 1});

xlabel("time");

ylabel("channel");

title(strcat("Minfr neural Data M1",sprintf(" |Dir%.2f",Data.reach\_dir{selected\_data\_to\_plot5,1})));

subplot(2,2,2);

imshow(Data.neural\_data\_PMd{selected\_data\_to\_plot3, 1})

xlabel("time");

ylabel("channel");

title(strcat("minfr neural Data PMd",sprintf(" |Dir%.2f",Data.reach\_dir{selected\_data\_to\_plot5,1})));

subplot(2,2,3);

imshow(Data.neural\_data\_M1{selected\_data\_to\_plot6, 1})

xlabel("time");

ylabel("channel");

title(strcat("maxfr neural Data M1",sprintf(" |Dir%.2f",Data.reach\_dir{selected\_data\_to\_plot6,1})));

subplot(2,2,4);

imshow(Data.neural\_data\_PMd{selected\_data\_to\_plot4, 1})

xlabel("time");

ylabel("channel");

title(strcat("maxfr neural Data PMd",sprintf(" |Dir%.2f",Data.reach\_dir{selected\_data\_to\_plot5,1})));

saveas(gcf,strcat(ipath,img\_header\_name,"firing\_rate\_min\_max.png"))

else

subplot(2,1,1);

imshow(Data.neural\_data\_PMd{selected\_data\_to\_plot, 1})

xlabel("time");

ylabel("channel");

title(strcat("neural Data PMd",sprintf(" |Dir%.2f",Data.reach\_dir{selected\_data\_to\_plot,1})));

subplot(2,1,2);

imshow(Data.neural\_data\_PMd{selected\_data\_to\_plot2, 1})

xlabel("time");

ylabel("channel");

title(strcat("neural Data PMd",sprintf(" |Dir%.2f",Data.reach\_dir{selected\_data\_to\_plot2,1})));

saveas(gcf,strcat(ipath,img\_header\_name,"firing\_rate.png"))

end

% figure();

% x = [Data.avgspeed{:}];

% y = mean([Data.average\_firing\_rate\_PMd{:}]);

% f = fit(x.',y.','gauss1');

% plot(f,x,y)

% xlabel("average reaching speed");

% ylabel("average firing rate")

% title("tuning curve, firing rate PMd vs average reach speed")

% if M1\_present

% figure();

% x = [Data.reach\_len{:}];

% y = mean([Data.average\_firing\_rate\_M1{:}]);

% f = fit(x.',y.','poly3');

% plot(f,x,y)

% xlabel("reaching length");

% ylabel("average firing rate")

% title("tuning curve, firing rate M1 vs reach length")

% end

%

% figure();

% x = ([Data.reach\_dir{:}]);

% y = mean([Data.average\_firing\_rate\_PMd{:}]);

% f = fit(x.',y.','gauss1');

% plot(f,x,y)

% xlabel("reaching Direction");

% ylabel("average firing rate")

% title("tuning curve, firing rate PMd vs reach direction")

%

% if M1\_present

% figure();

% x = [Data.reach\_dir{:}];

% y = mean([Data.average\_firing\_rate\_M1{:}]);

% f = fit(x.',y.','gauss1');

% plot(f,x,y)

% xlabel("reaching Direction");

% ylabel("average firing rate")

% title("tuning curve, firing rate m1 vs reach direction")

% end

else

figure();

x = [Data.reach\_dir{:}];

y = [Data.average\_firing\_rate\_PMd{:}];

f = fit(x.',y(select\_neuron,:).','gauss1');

plot(f,x,y)

xlabel("reaching Direction");

ylabel("average firing rate")

title("tuning curve, firing rate PMd vs reach direction")

saveas(gcf,strcat(ipath,img\_header\_name,"fit\_gauss\_PMd\_tunning\_curve\_single\_neuron.png"))

if M1\_present

figure();

x = [Data.reach\_dir{:}];

y = [Data.average\_firing\_rate\_M1{:}];

f = fit(x.',y(select\_neuron,:).','gauss1');

plot(f,x,y)

xlabel("reaching Direction");

ylabel("average firing rate")

title("tuning curve, firing rate m1 vs reach direction")

saveas(gcf,strcat(ipath,img\_header\_name,"fit\_gauss\_M1\_tunning\_curve\_single\_neuron.png"))

end

figure();

x = [Data.reach\_len{:}];

y = [Data.average\_firing\_rate\_PMd{:}];

f = fit(x.',y(select\_neuron,:).','poly3');

plot(f,x,y)

xlabel("reaching length");

ylabel("average firing rate")

title("tuning curve, firing rate PMd vs reach length")

saveas(gcf,strcat(ipath,img\_header\_name,"fit\_gauss\_PMd\_poly\_len\_tunning\_curve\_single\_neuron.png"))

if M1\_present

figure();

x = [Data.reach\_len{:}];

y = [Data.average\_firing\_rate\_M1{:}];

f = fit(x.',y(select\_neuron,:).','poly3');

plot(f,x,y)

xlabel("reaching length");

ylabel("average firing rate")

title("tuning curve, firing rate M1 vs reach length")

saveas(gcf,strcat(ipath,img\_header\_name,"fit\_gauss\_M1\_poly\_len\_tunning\_curve\_single\_neuron.png"))

end

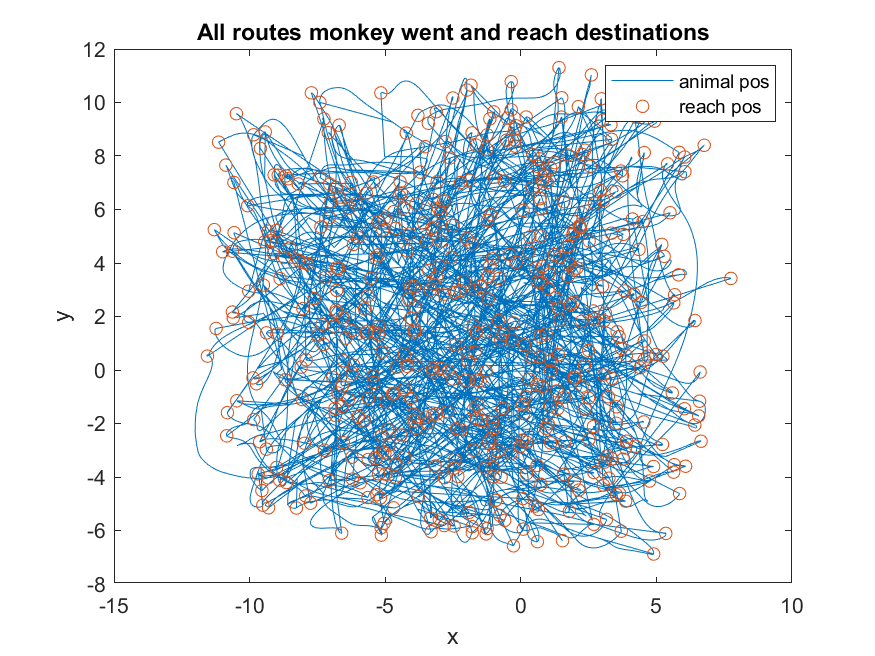
end

# خروجی‌ها

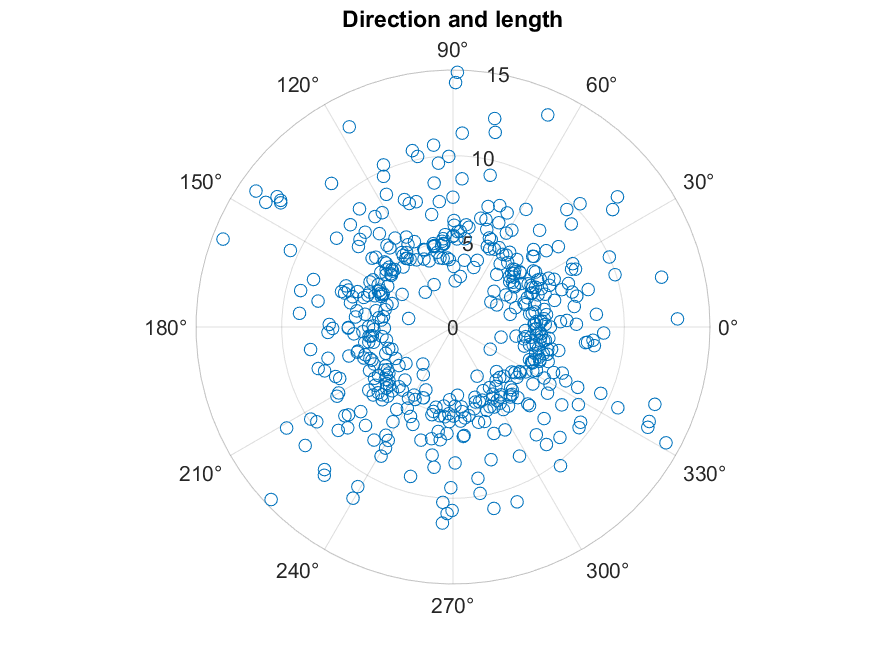
تصاویر به دست آمده به شرح زیر می‌باشد. توضیحات هر یک از نمودار‌ها ذکر می‌شود. تصاویر خروجی از نشست‌های مختلف، توضیحات مشابهی‌دارند.

## تصاویر MM-S1 نشست اول میمون MM

در شکل 2 می‌توان مسیر حرکت دست میمون را در طول آزمایش مشاهده کرد. دایره‌های قرمز، نشان‌دهنده‌ی مکان هدف می‌باشد که در هر آزمایش نشان داده می‌شود و به طور تصادفی در بازه‌های مشخص شده در شکل 3 به روز رسانی می‌شوند. شکل ۳ نمایش‌دهنده‌ی طول و جهت فاصله‌ی بین دو محرک است. همانطور که مشاهده می‌شود، عمده‌ی فاصله‌ها بین ۵ الی ۱۵ سانتی‌متر از محرک قبلی نمایش داده شده اند و زاویه‌ی آنها هم نسبت به هم به صورت یکنواخت در تمام بازه‌ی ۰ تا ۳۶۰ درجه پخش شده است.

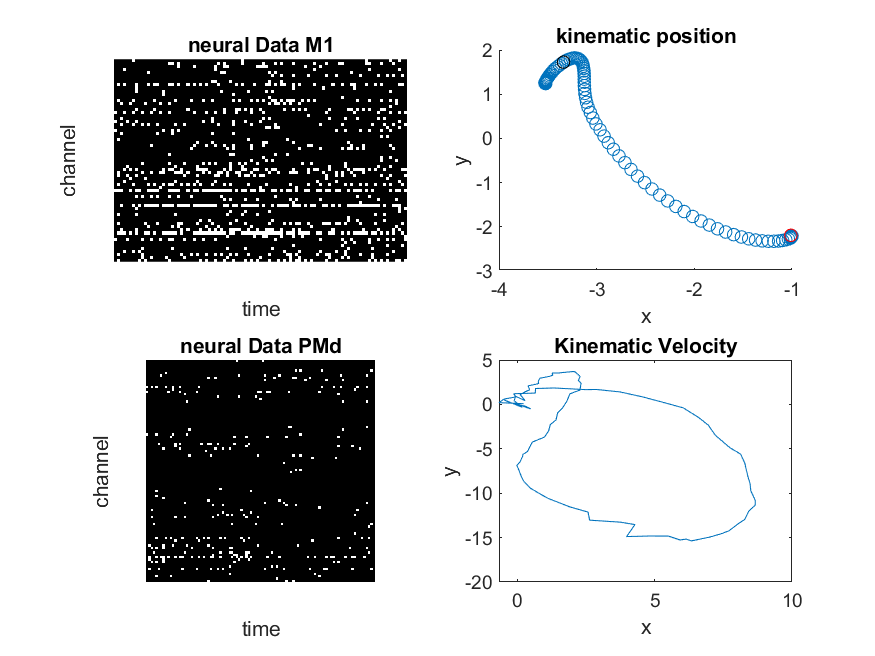


شکل 2 نمایش نقاط تحریک و مسیر حرکت دست میمون



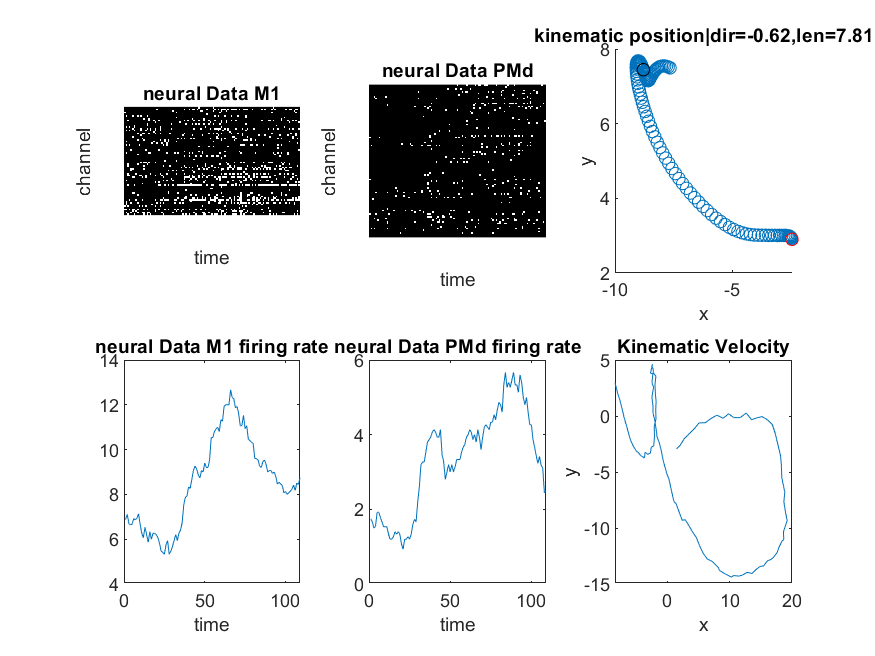
شکل 3 جهت و فاصله‌تا تحریک بعدی

در شکل 4 می‌توان اطلاعات اولین ترایال این نشست را مشاهده کرد. در نمودار سمت راست بالا می‌توان اطلاعات موقعیت دست میمون را مشاهده کرد. دایره‌ی مشکی نشان‌دهنده‌ی هدف قبلی (شروع) و دایره‌ی قرمز نشان‌دهنده‌ی هدف این ترایال می‌باشد. نمودار سمت راست پایین، نشان‌دهنده‌ی تغییر سرعت حرکت دست میمون است که قابل تطبیق با نمودار موقعیت است. نمودار‌های سمت چپ نشان‌دهنده‌ی اسپایک‌ها‌ی ثبت شده از نورون‌ها می‌باشد. شکل بالا مربوط به ناحیهM1 و پایینی مربوط به PMd می‌باشد.



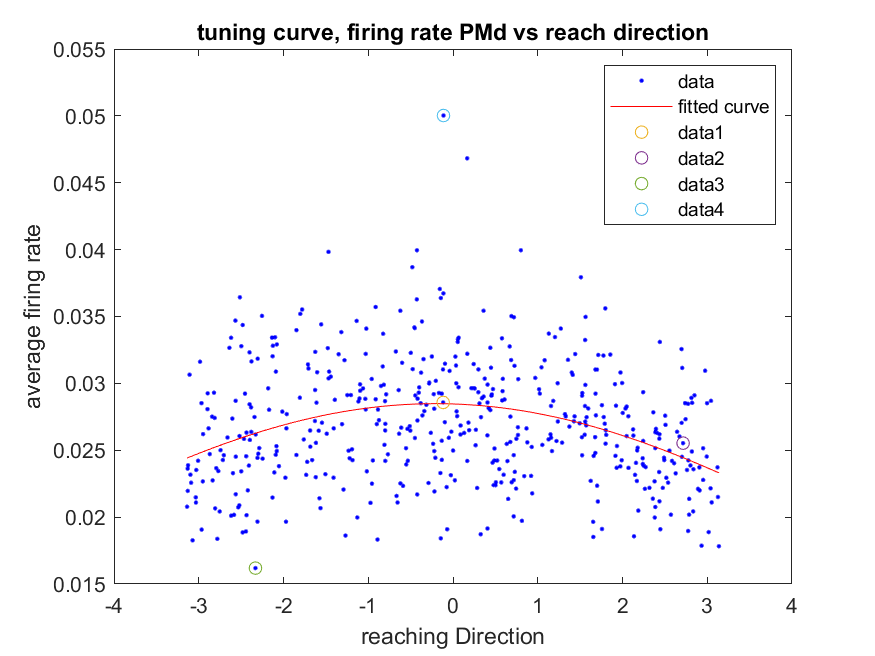
شکل 4نمایش مشخصات موقعیت، سرعت و اسپایک اولین ترایال

شکل 5 نشان‌دهنده‌ی نموداری شبیه شکل 4 می‌باشد. این اطلاعات مربوط به آزمایش Reach شماره ۱۶۲ می‌باشد. در این شکل اطلاعات جهت و فاصله‌ی محرک نیز ذکر شده که برابر با -0.62 رادیان و 7.81 سانتی متر می‌باشد. دو نمودار پایین چپ و پایین وسط نیز تخمین‌هایی از نرخ فایرینگ است. این تخمین با جمع اثر همه‌ی نورون‌ها و با اعمال یک فیلتر moving average با پنجره‌ی ۱۵ نمونه‌ای اعمال شده. که ‌هر یک متناسب با نمودار‌های بالایی خود هستند.

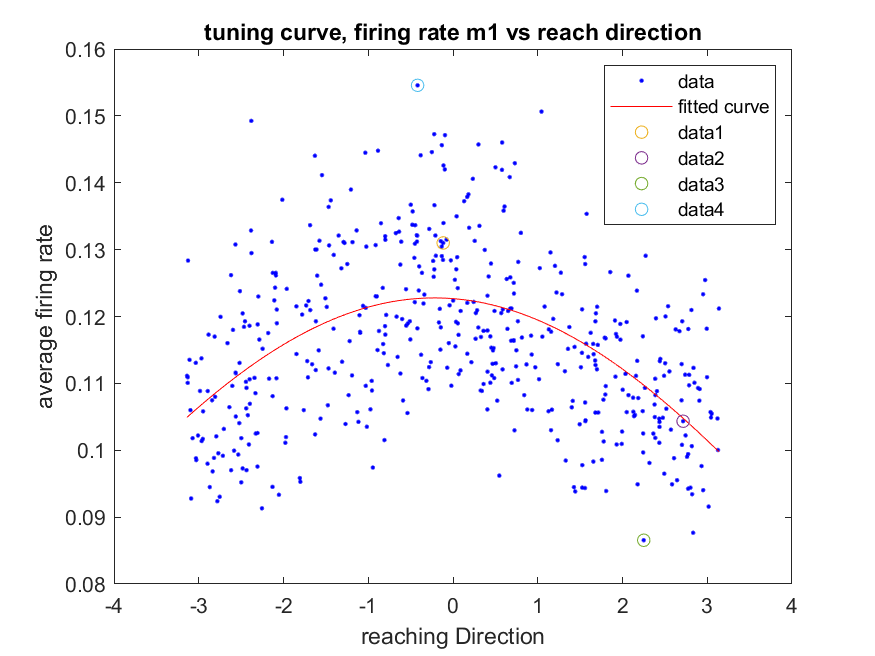


شکل 5

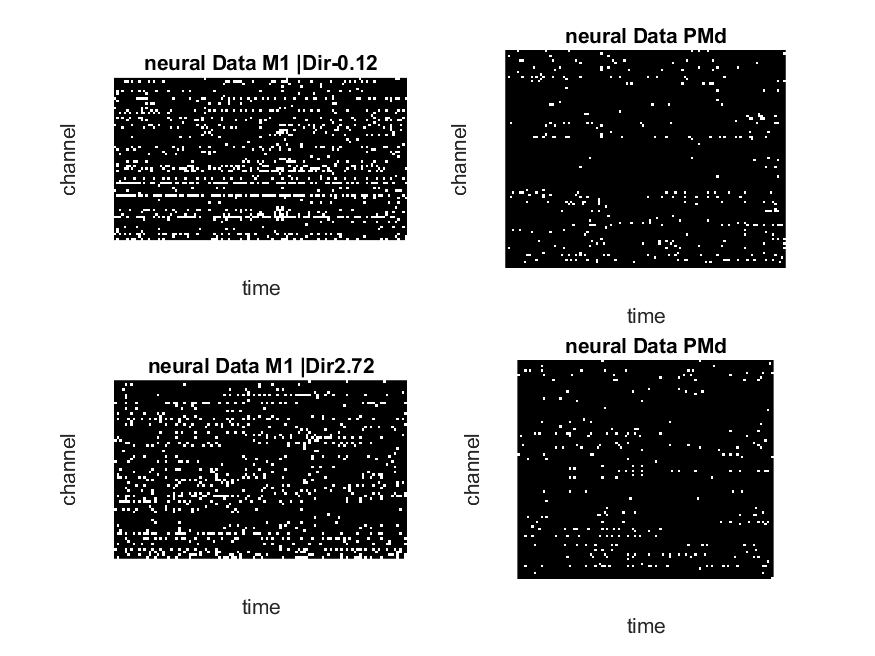
شکل 6 و شکل 7 نمایش tuning curve از نرخ فایرینگ را در مواجهه با جهت ظاهر شدن تحریک نشان می‌دهند.یک نمودار گوسین به این داده‌ها فیت شده است که قابل مشاهده است. همانطور که در این دو شکل قابل ملاحظه است، ۴ داده با دایره مشخص شده که قطار اسپایک‌آنها در شکل 8 و شکل 9 قابل مشاهده است. در این اشکال می‌توان تفاوت فایرینگ ریت را مشاهده کرد. با توجه به این که تحریک ما بر خلاف تحریک‌های بینایی لحظه ای عمل نمی‌کند، بنابراین تاثیر لحظه‌ای نخواهد داشت و تاثیر آن در طول مدت زمان ارائه‌ی محرک خواهد بود.



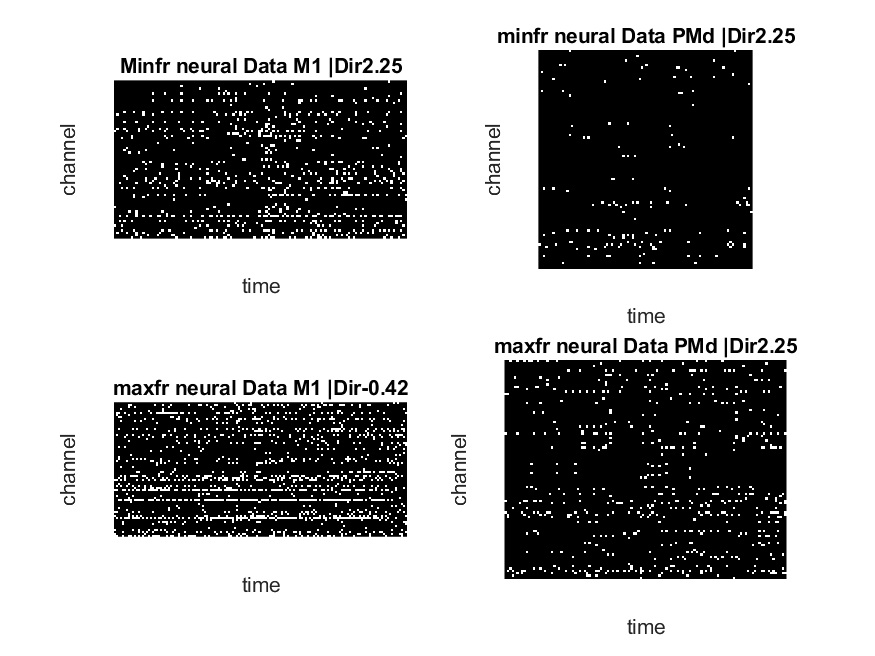
شکل 6 tunning curve بر اساس جهت تحریک و استخراج شده از ناحیه‌ی PMd



شکل 7 tunning curve بر اساس جهت تحریک و استخراج شده از ناحیه‌ی M1



شکل 8 نمایش اسپایک‌ها در جهت‌های مختلف.

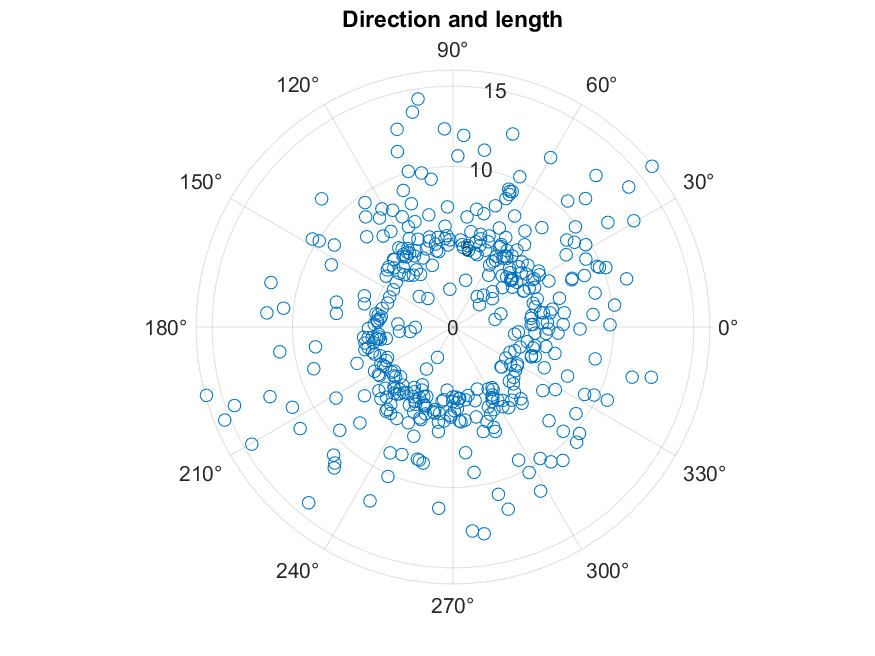


شکل 9 نمایش اسپایک‌ها در حالت‌های کمینه و بیشینه

## تصاویر MT-S1: نشست اول میمون MT

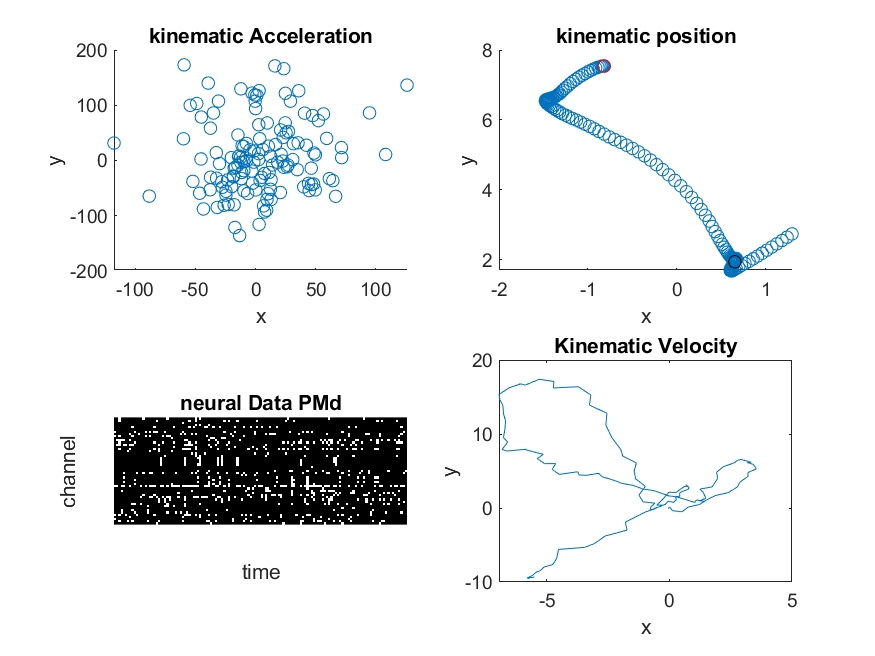


شکل 10

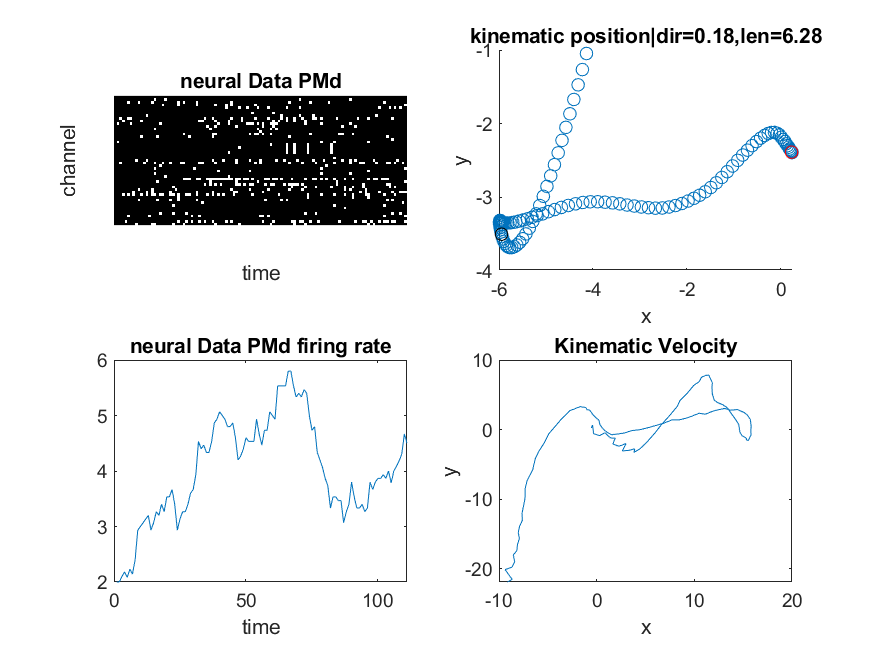


شکل 11

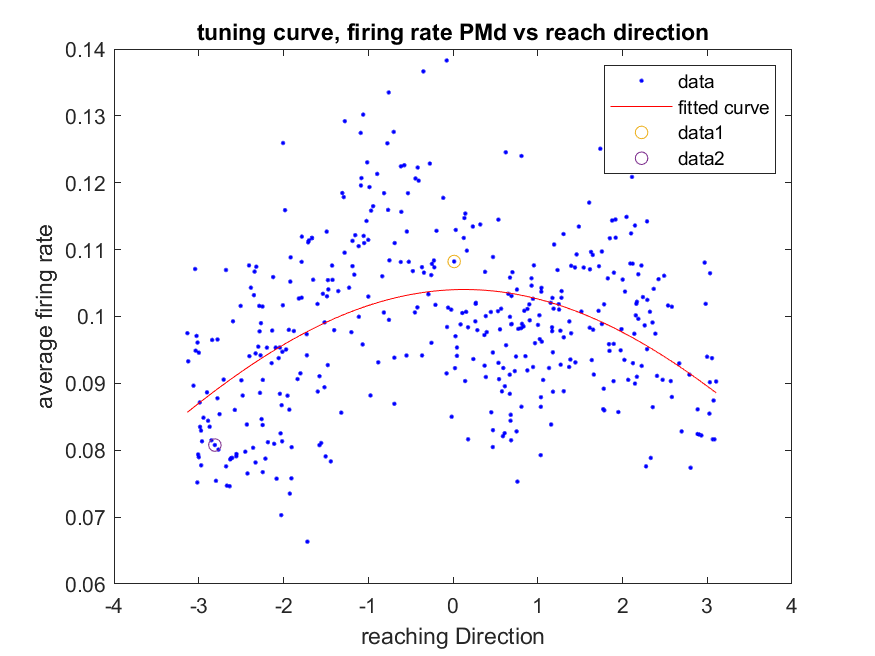
در شکل 12 با توجه به این که M1 نداریم، در تصویر بالا سمت چپ نمودار شتاب نمایش داده شده است.



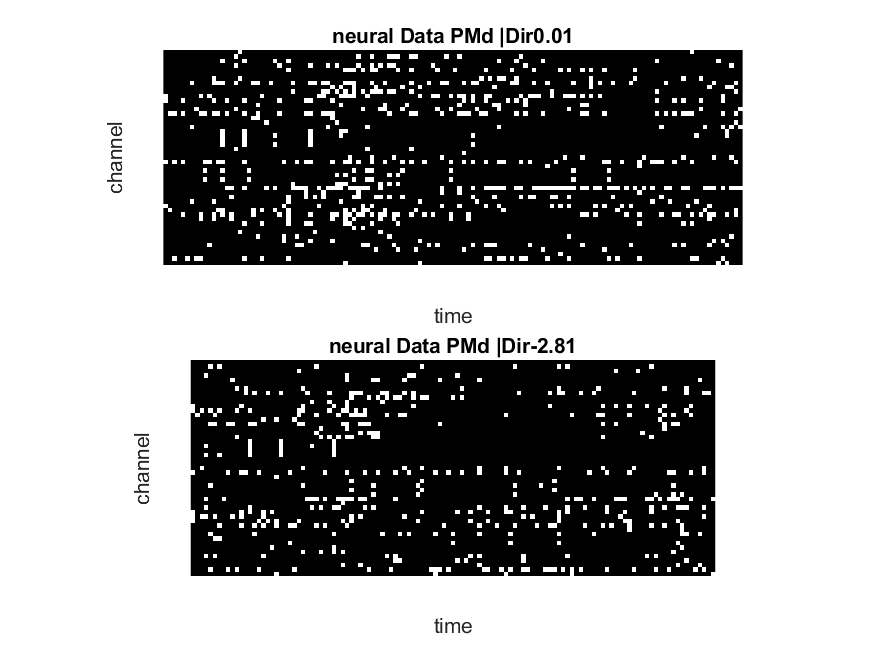
شکل 12



شکل 13

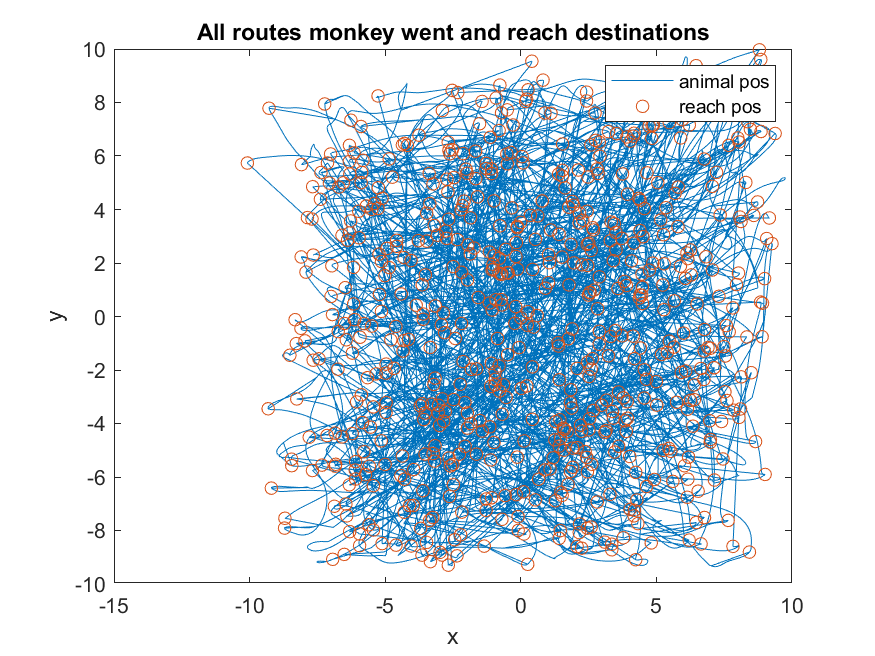


شکل 14

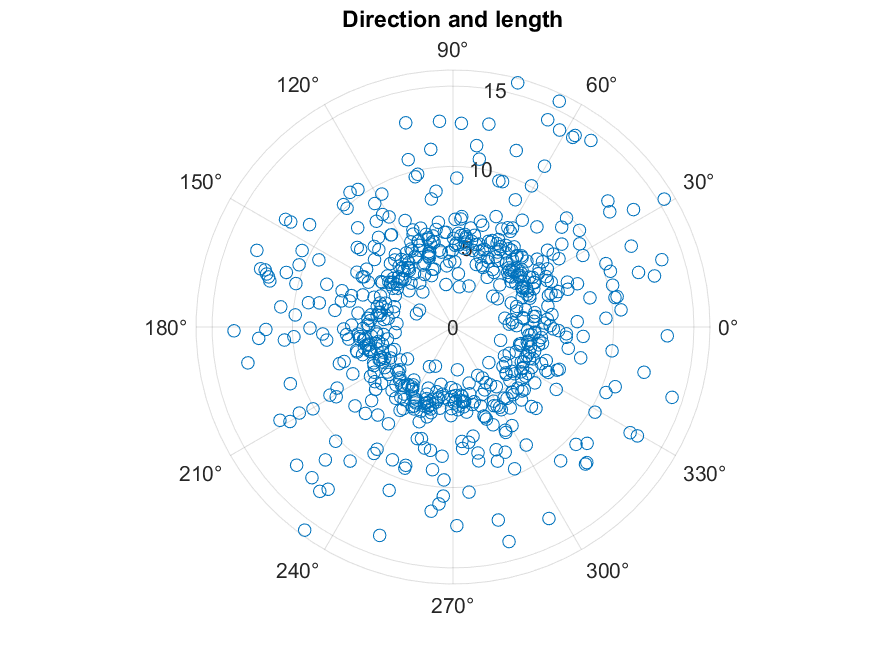


شکل 15

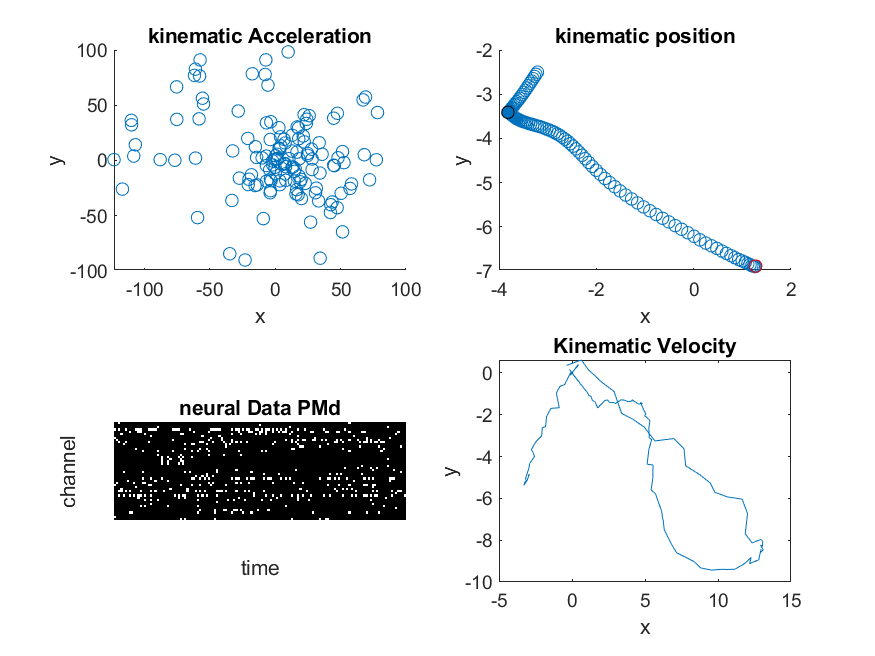
## تصاویر MT-S2:



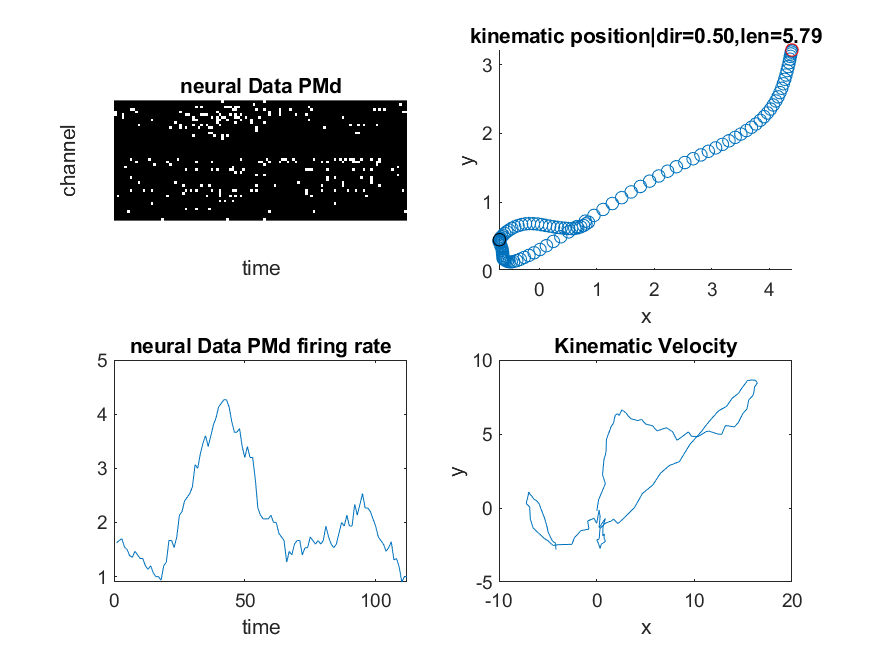
شکل 16



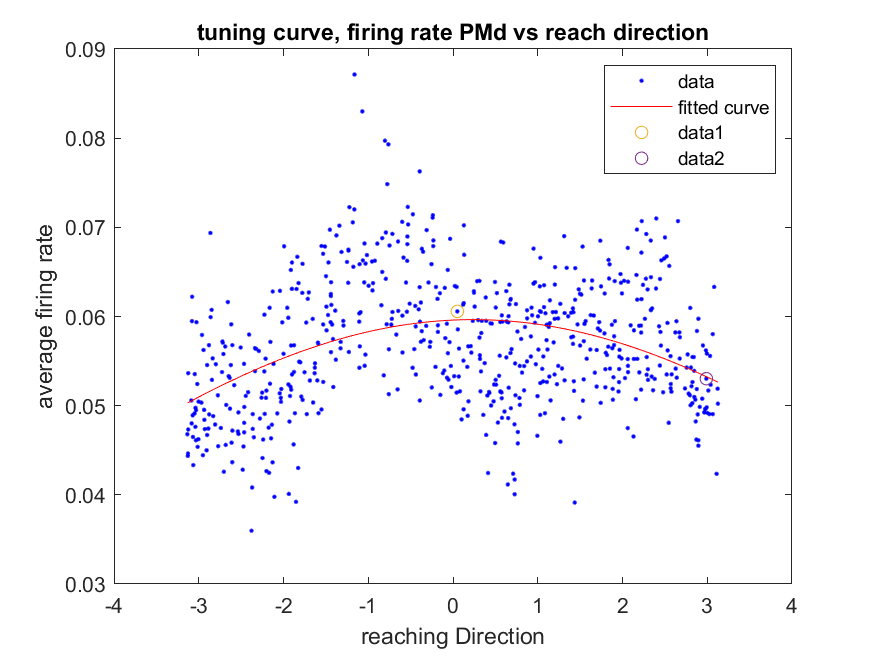
شکل 17



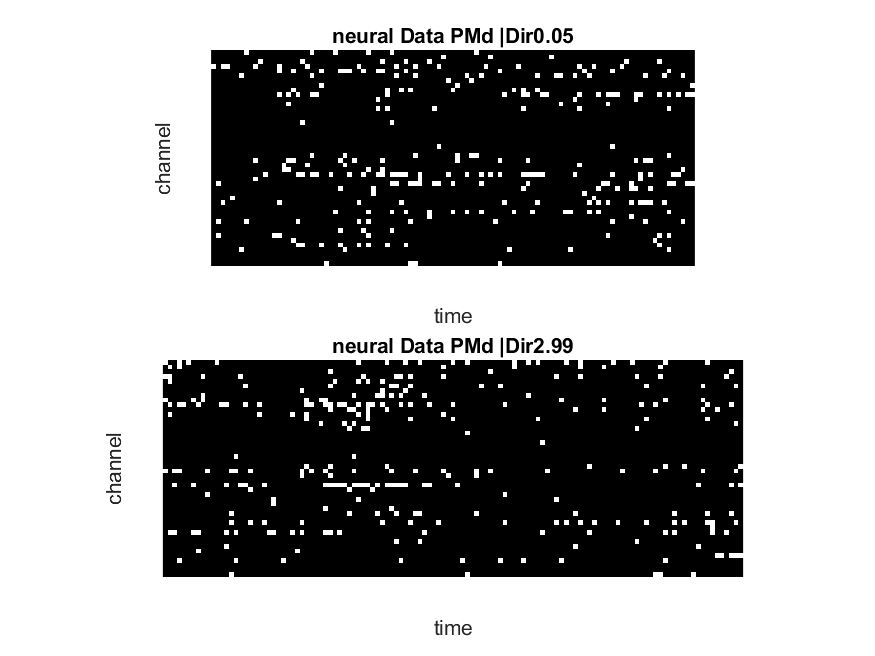
شکل 18



شکل 19

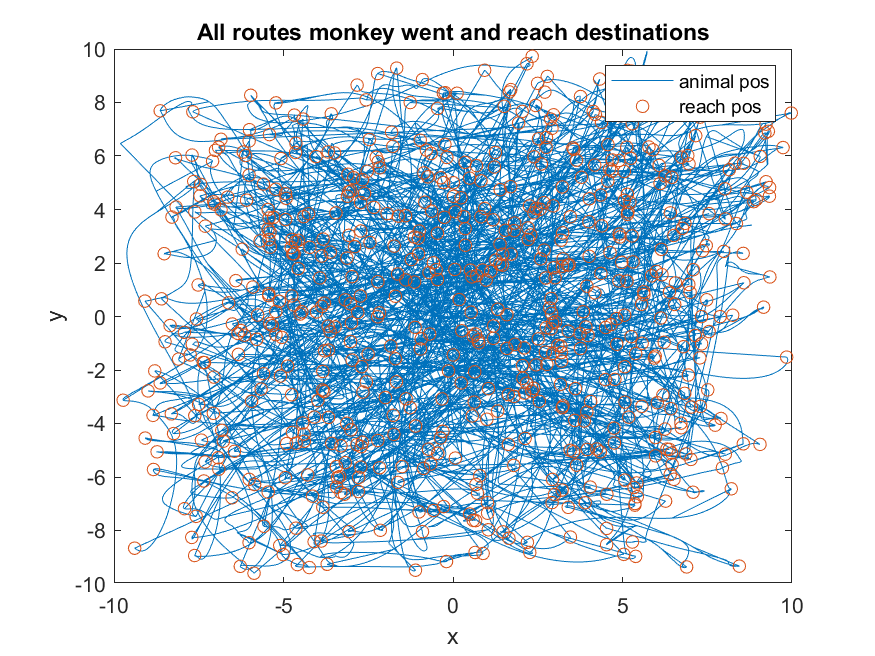


شکل 20

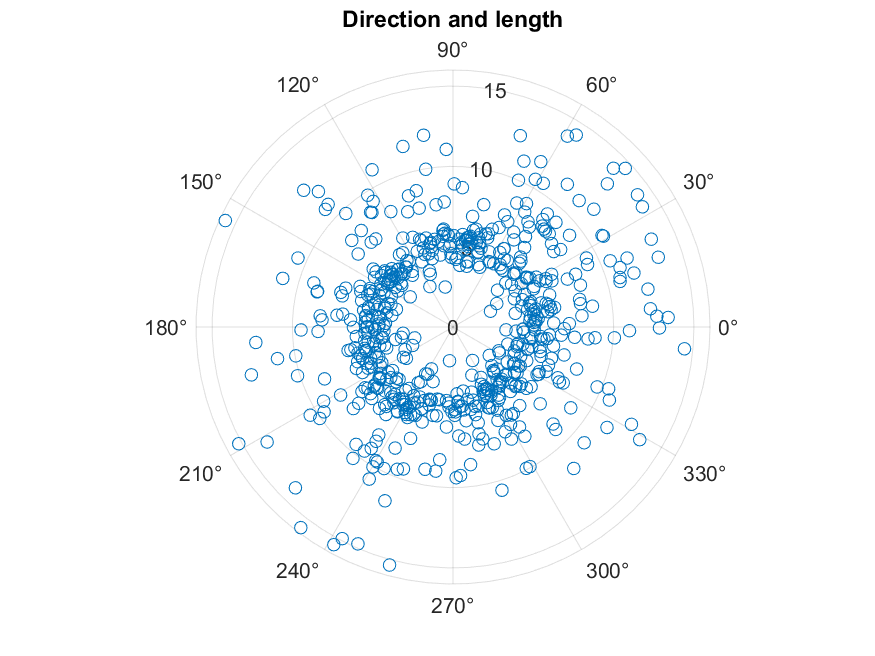


شکل 21

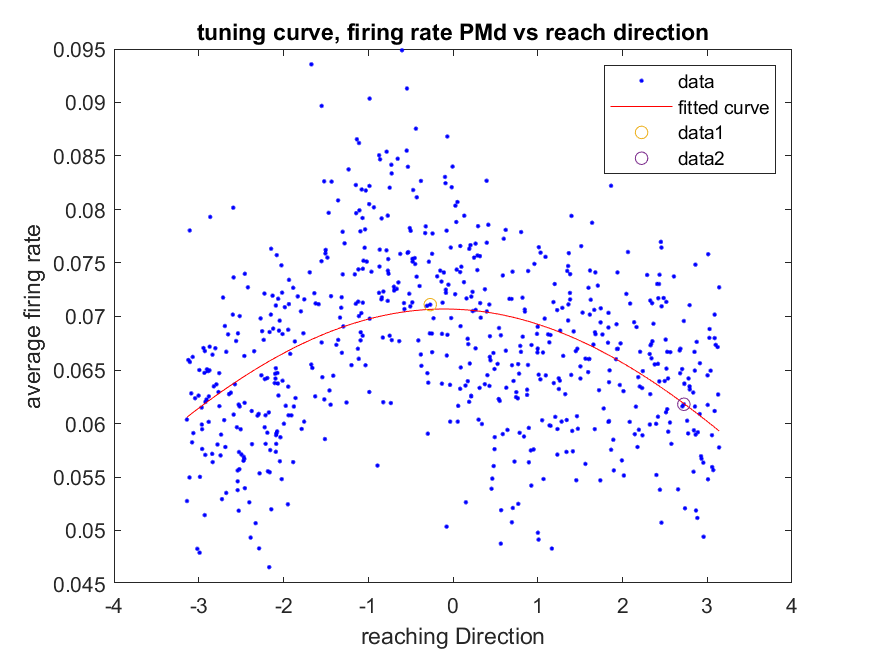
## تصاویر MT-S3:



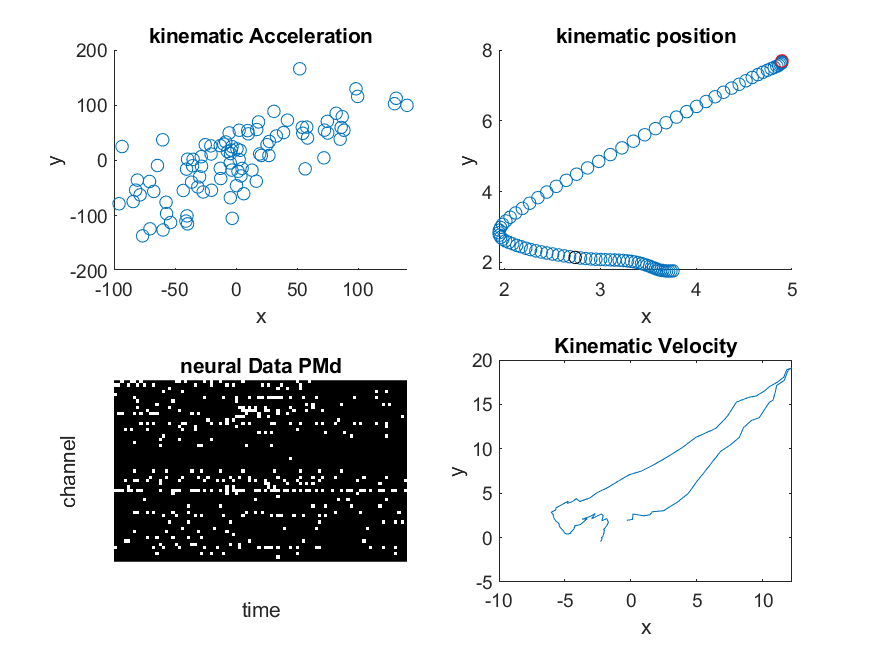
شکل 22



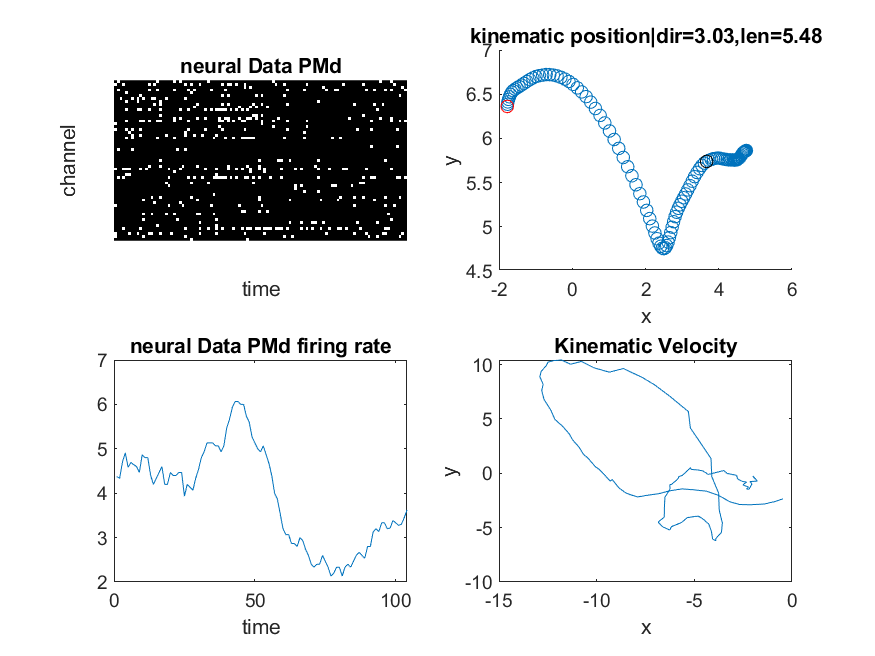
شکل 23



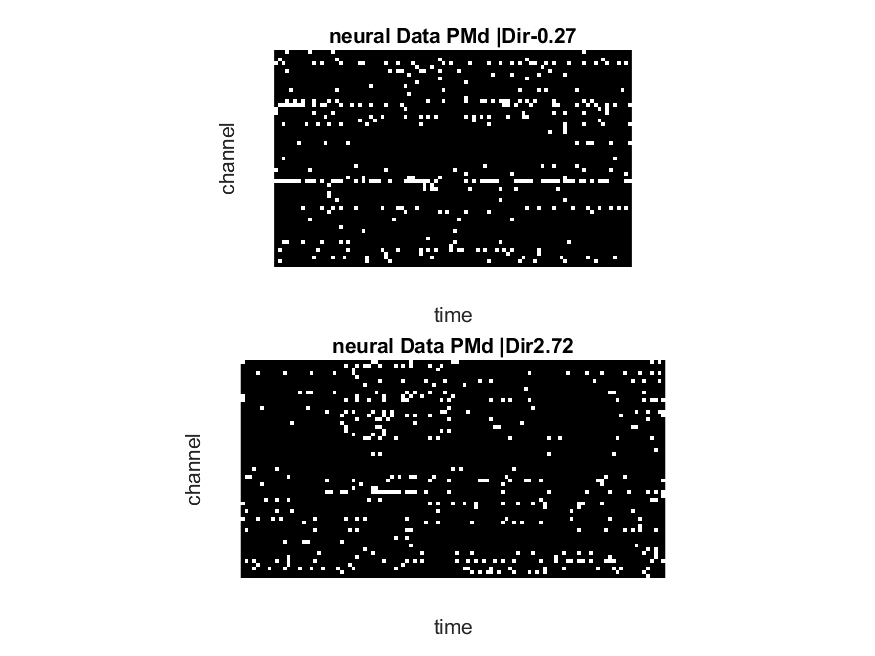
شکل 24



شکل 25



شکل 26



شکل 27

لینک دیتاست:

<https://crcns.org/data-sets/motor-cortex/pmd-1/about-pmd-1>

لینک گیت‌هاب پروژه:

<https://github.com/hamidreza-abooei/Project_computationalNeuroscience>

با تشکر

1. premotor cortex [↑](#footnote-ref-1)
2. primary motor cortex [↑](#footnote-ref-2)
3. reach [↑](#footnote-ref-3)
4. two-link planar manipulandum [↑](#footnote-ref-4)