Tether Conclusions

The following MATLAB Live Script details the conclusions of the tether experiments, including the resulting plots for a total of 30 experiments perfomed on the orange 4-limbed robot on the black mat.

```
% Dependencies:
%   +demos/data/visualtracking
%   +offlineanalysis/GaitTest
%   +gaitdef/Gait

% [0] == Script setup
clear; clc; close all

% Add dependencies to classpath
addpath('../');
addpath('data/visualtracking');

% Configure figure tex interpreters
set(groot, 'defaultAxesTickLabelInterpreter','latex');
set(groot, 'defaultTextInterpreter','latex');
set(0, 'DefaultAxesFontSize', 18);
```

Extract and define parameters for GaitTest() objects.

From 20220707 Experiments:

Gait B-120 [heavy - not following (restting) - left/up]

```
gait_names{1}  = 'B_120_H_NF_L.mat';
```

Gait E-120 [heavy - not following - left/up]

```
gait_names{2}  = 'E_120_H_NF_L.mat';
```

Gait E-60 [heavy - not following - left/up]

```
gait_names{3}  = 'E_60_H_NF_L.mat';
```

Gait E-60* [heavy - not following - right/up] Caution! Not real E gait! Limb A not actuating

```
gait_names{4}  = 'Es_60_H_NF_R.mat';
```

From 20220819 Experiments:

Gait B-120 [light sheath - not following (restting) - right/up]

```
gait_names{5}  = 'B_120_S_NF_R.mat';
```

From 20220829 Experiments:

Gait B Follow (Left)* [light sheath - following - left] Caution! Not real B gait! Limb A not actuating

```
gait_names{6}  = 'Bs_60_S_F_L.mat';
```

Gait B Follow (Right)* [light sheath - following - right] Caution! Not real B gait! Limb A not actuating

```
gait_names{7}  = 'Bs_60_S_F_R.mat';
```

Gait E Left (sheath on) [light sheath - not following - left]

```
gait_names{8}  = 'E_60_S_NF_L.mat';
```

Gait E Right (sheath on) [no sheath - not following - right]

```
gait_names{9}  = 'E_60_S_NF_R.mat';
```

Gait E Left (sheath off) [no sheath - not following - left]

```
gait_names{10} = 'E_60_NS_NF_L.mat';
```

Gait E Right (sheath off)* [no sheath - not following - right] Caution! Not real E gait! Limb B not actuating

```
gait_names{11} = 'Es_60_NS_NF_R.mat';
```

From 20220901 Experiments:

Gait E Left (sheath off) [no sheath - not following - left (flipped)]

```
gait_names{12} = 'E_60_NS_NF_Lf.mat';
```

Gait B Follow (Left) Trial 1 [light sheath - following - left]

```
gait_names{13} = 'B_60_S_F_L.mat';
```

Gait B Follow (Left) Trial 2 [light sheath - following - left (flipped)]

```
gait_names{14} = 'B_60_S_F_Lf.mat';
```

Gait B Left (Sheath on) Trial 1 [light sheath - following - left (flipped)]

```
gait_names{15} = 'B_60_S_NF_L.mat';
```

Gait B Left (Sheath on) Trial 2 [light sheath - following - left (flipped)]

```
gait_names{16} = 'B_60_S_NF_Lf.mat';
```

From 20220908 Experiments:

Gait B Right (sheath on) [sheath - not following - right] (not consistent / semi-following)

```
gait_names{17} = 'B_60_S_NF_R.mat';
```

Gait B Right Follow (sheath off) Trial 1 [no sheath - following - right]

```
gait_names{18} = 'B_60_NS_F_R.mat';
```

Gait B Right Follow (sheath off) Trial 2 [no sheath - following - right]

```
gait_names{19} = 'B_60_NS_F_R_2.mat';
```

Gait B Left Follow (sheath off) Trial 1 [no sheath - following - left]

```
gait_names{20} = 'B_60_NS_F_L.mat';
```

Gait B Left Follow (sheath off) Trial 2 [no sheath - not following - left]

```
gait_names{21} = 'B_60_NS_F_L_2.mat';
```

Gait B Right (sheath off) Trial 1 [no sheath - not following - right] (not consistent / semi-following)

```
gait_names{22} = 'B_60_NS_NF_R.mat';
```

Gait B Right (sheath off) Trial 2 [no sheath - not following - right] (not consistent / semi-following)

```
gait_names{23} = 'B_60_NS_NF_R_2.mat';
```

Gait B Left (sheath off) Trial 1 [no sheath - not following - left] (not consistent / semi-following)

```
gait_names{24} = 'B_60_NS_NF_L.mat';
```

Gait B Left (sheath off) Trial 2 [no sheath - not following - left] (not consistent / semi-following)

```
gait_names{25} = 'B_60_NS_NF_L_2.mat';
```

Gait B Right Follow (sheath on) [sheath - following - right]

```
gait_names{26} = 'B_60_S_F_R.mat';
```

Gait E Right (sheath off) Trial 1 [no sheath - not following - right] (not consistent / semi-following)

```
gait_names{27} = 'E_60_NS_NF_R.mat';
```

Gait E Right (sheath off) Trial 2 [no sheath - not following - right] (not consistent / semi-following)

```
gait_names{28} = 'E_60_NS_NF_R_2.mat';
```

Gait E Left (sheath off) Trial 1 [no sheath - not following - left] (not consistent / semi-following)

```
gait_names{29} = 'E_60_NS_NF_L_2.mat';
```

Gait E Left (sheath off) Trial 2 [no sheath - not following - left] (not consistent / semi-following)

```
gait_names{30} = 'E_60_NS_NF_L_3.mat';
```

Build Experiment Matrix

```
if isrow(gait_names)
    gait_names = gait_names';
end
n_gaits = length(gait_names);

% Extract characteristics of each gait test.
file_names = split(gait_names, '.');
gait_characteristics = cell(n_gaits, 7);
for i = 1:n_gaits
    gait_characteristics(i, 1) = num2cell(i);
    n_underscores = count(gait_names{i}, '_');
    if n_underscores == 4
        gait_characteristics(i, 2:6) = split(file_names(i,1), '_');
        gait_characteristics(i, 7) = num2cell(1);
    else
        gait_characteristics(i, 2:7) = split(file_names(i,1), '_');
    end
end
characteristics = ["Experiment", "Gait", "#Cycles", "Tether", ...
             "Protocol", "Placement", "Trial"];
% Build table.
experiments = cell2table(gait_characteristics, "VariableNames", ...
             characteristics);
sorted_exps = sortrows(experiments, characteristics([2, 4, 5, 6, 7]))
```

sorted_exps = 30×7 table

	Experiment	Gait	#Cycles	Tether	Protocol	Placement	Trial
1	1	'B'	'120'	'H'	'NF'	'L'	
2	20	'B'	'60'	'NS'	'F'	'L'	
3	21	'B'	'60'	'NS'	'F'	'L'	2
4	18	'B'	'60'	'NS'	'F'	'R'	
5	19	'B'	'60'	'NS'	'F'	'R'	2
6	24	'B'	'60'	'NS'	'NF'	'L'	
7	25	'B'	'60'	'NS'	'NF'	'L'	2
8	22	'B'	'60'	'NS'	'NF'	'R'	
9	23	'B'	'60'	'NS'	'NF'	'R'	2
10	13	'B'	'60'	'S'	'F'	'L'	

	Experiment	Gait	#Cycles	Tether	Protocol	Placement	Trial
11	14	'B'	'60'	'S'	'F'	'Lf'	
12	26	'B'	'60'	'S'	'F'	'R'	
13	15	'B'	'60'	'S'	'NF'	'L'	
14	16	'B'	'60'	'S'	'NF'	'Lf'	
15	5	'B'	'120'	'S'	'NF'	'R'	
16	17	'B'	'60'	'S'	'NF'	'R'	
17	6	'Bs'	'60'	'S'	'F'	'L'	
18	7	'Bs'	'60'	'S'	'F'	'R'	
19	2	'E'	'120'	'H'	'NF'	'L'	
20	3	'E'	'60'	'H'	'NF'	'L'	
21	10	'E'	'60'	'NS'	'NF'	'L'	
22	29	'E'	'60'	'NS'	'NF'	'L'	2
23	30	'E'	'60'	'NS'	'NF'	'L'	3
24	12	'E'	'60'	'NS'	'NF'	'Lf'	
25	27	'E'	'60'	'NS'	'NF'	'R'	
26	28	'E'	'60'	'NS'	'NF'	'R'	2
27	8	'E'	'60'	'S'	'NF'	'L'	
28	9	'E'	'60'	'S'	'NF'	'R'	
29	4	'Es'	'60'	'H'	'NF'	'R'	
30	11	'Es'	'60'	'NS'	'NF'	'R'	

```
% Construct titles.
title_components = string(gait_characteristics);
title_components(strcmp(experiments.Tether, 'H'), 4) = 'heavy sheath';
title_components(strcmp(experiments.Tether, 'S'), 4) = 'light sheath';
title_components(strcmp(experiments.Tether, 'NS'), 4) = 'no sheath';
title_components(strcmp(experiments.Protocol, 'F'), 5) = 'following';
title_components(strcmp(experiments.Protocol, 'NF'), 5) = 'not following';
title_components(strcmp(experiments.Placement, 'L'), 6) = 'left';
title_components(strcmp(experiments.Placement, 'R'), 6) = 'right';
title_components = [repmat(["Experiment" ":" "cycles of Gait" "with" "tether (" "," "),"  "trial
title_matrix = title_components(:, [1,9,2,11,3,10,4,12,5,14,6,13,7,8,15]);
exp_title = join(title_matrix);
% Define experimental parameters after investigating videos.
frame_start_list(1:4) = [382, 234, 141, 109];
frame_start_list(5) = 76;
frame_start_list(6:11) = [210, 168, 131, 131, 55, 983];
frame_start_list(12:16) = [132, 184, 278, 324, 196];
frame_start_list(17:25) = [63, 45, 39, 49, 137, 36, 63, 39, 519];
frame_start_list(26:30) = [88, 51, 64, 50, 57];
```

```
% Find marker order by investigating first frame.
marker_order_list = repmat([1 3 4 2],n_gaits, 1);
marker_order_list([2,3],:) = repmat([1 4 3 2],2,1);
marker_order_list([8,9,10,21,25],:) = repmat([2 4 3 1],5,1);
marker_order_list([12,14,16],:) = repmat([4 2 1 3],3,1);
marker_order_list(26,:) = repmat([2,3,4,1],1,1);

show_markers = false;     % plots first frame of each video

% Initialize the stability experiment struct.
stab_exp = struct('params', [], 'raw_data', []);
gait_sequences = cell(n_gaits,1);

% Extract data.
for i = 1:n_gaits
    % Define robot / experiment parameters.
    stab_exp(i).params.robot_name = 'orange';
    stab_exp(i).params.substrate = 'black mat';
    stab_exp(i).params.n_markers = 4;
    stab_exp(i).params.pixel_length = 1/8.6343;     % cm per pixel

    % Define number of gait cycles run.
    if ismember(i,[1,2,5])
        stab_exp(i).params.n_cycles= 119;
         % accounts for pause after first cycle in some of the videos
    else
        stab_exp(i).params.n_cycles= 59;
    end

    % Extract and store first frame information.
    stab_exp(i).params.frame_1 = frame_start_list(i);
    stab_exp(i).params.marker_order = marker_order_list(i,:);

    % Extract and store raw data from each trial
    filename = gait_names{i};
    stab_exp(i).raw_data = load(filename).tracking_data;
    if experiments.Gait{i} == 'B'
        gait_sequences{i} = [16,7,5,11,14];   % Gait B (rotational)
    elseif experiments.Gait{i} == 'E'
        gait_sequences{i} = [9,16,1];         % Gait E (translational)
    elseif experiments.Gait{i} == 'Bs'
        gait_sequences{i} = [16,7,5,11,14];
    elseif experiments.Gait{i} == 'Es'
        gait_sequences{i} = [9,16,1];
    else
        gait_sequences{i} = [];
    end
end
```

Rotate the data w.r.t. the initial global orientation.

```
% Define first experiment first frame orientation as theta = 0.
markers_x(1, :) = stab_exp(1).raw_data(1, 1:3:stab_exp(1).params.n_markers*3-2);
markers_y(1, :) = stab_exp(1).raw_data(1, 2:3:stab_exp(1).params.n_markers*3-1);
centroid(:, :, 1) = mean([markers_x(1, :); markers_y(1, :)], 2);
% Move initial position to (0,0) origin.
reference_markers = [markers_x(1, [1 3 4 2]); markers_y(1, [1 3 4 2]);]...
                    - centroid(:, :, 1);

% For each trial, rotate data to align first frame global orientations.
 for i = 1:n_gaits
    % Find initial rotation matrix for each trial to have consistent fixed frame.
    markers_x(i, :) = stab_exp(i).raw_data(1, 1:3:stab_exp(i).params.n_markers*3-2);
    markers_y(i, :) = stab_exp(i).raw_data(1, 2:3:stab_exp(i).params.n_markers*3-1);
    centroid(:, :, i) = mean([markers_x(i, :); markers_y(i, :)], 2);

    % Move initial position to (0,0) origin.
    shifted_markers = [markers_x(i, stab_exp(i).params.marker_order);
                       markers_y(i, stab_exp(i).params.marker_order)]...
                       - centroid(:, :, i);

    % Find orientation of subsequent trials w.r.t. first trial GCS.
    [regParams,~,~] = absor(reference_markers, shifted_markers);
    stab_exp(i).params.R_1 = [regParams.R zeros(2,1); 0 0 1];
 end
```

Instantiate GaitTest() objects for each experimental trial.

This analyzes the data from each trial to find motion primitive twist information.

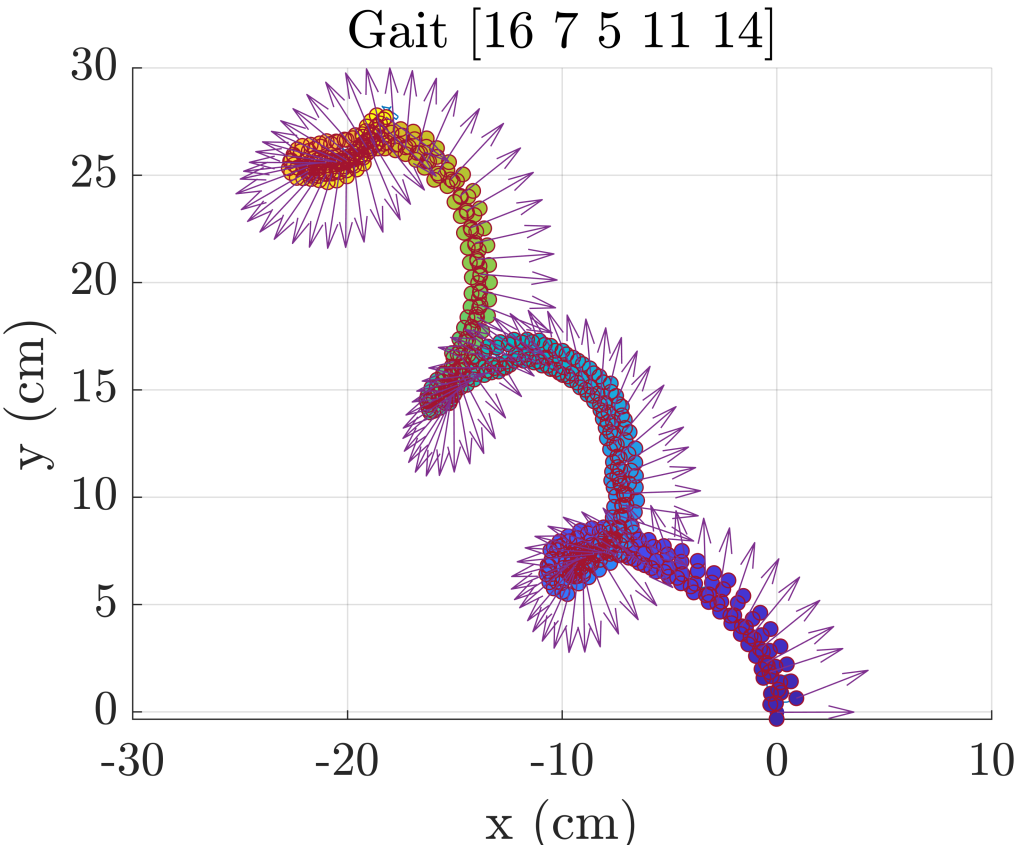
```
% Instantiate objects for each gait tested.
all_gaits = offlineanalysis.GaitTest.empty(0,n_gaits);
gait_defs = gaitdef.Gait.empty(0,n_gaits);
tail_idx = cell(n_gaits, 1);
head_idx = cell(n_gaits, 1);
delta_poses = cell(n_gaits,1);
global_theta = cell(n_gaits,1);
twists = cell(n_gaits,1);
cent_rot = cell(n_gaits,1);
rad_curv = cell(n_gaits,1);

% Calculate twists for each gait experiment.
for i = 1:n_gaits
    all_gaits(i) = offlineanalysis.GaitTest(stab_exp(i).raw_data, ...
                                            gait_sequences{i}(1,:), ...
                                            stab_exp(i).params);
    gait_defs(i) = gaitdef.Gait(all_gaits(i), stab_exp(i).params);
    % Manually find the twists.
```

```
    % Extract gait / experiment details.
    n_cycles = all_gaits(i).n_cycles;
    len_gait = all_gaits(i).len_gait;
     for j = 1:n_cycles - 1
         % Record indexes for each motion primitive tail and head.
         tail_idx{i}(1,j) = all_gaits(i).keyframes(len_gait*(j-1)+2);
         head_idx{i}(1,j) = all_gaits(i).keyframes(len_gait*j+2);

         % Find change in poses w.r.t. global inertial frame for each motion primitive.
         delta_poses{i}(:,j) = all_gaits(i).raw_poses(:, head_idx{i}(1,j)) - all_gaits(i).raw_p
         delta_poses{i}(1:2,j) = all_gaits(i).pixel_length*delta_poses{i}(1:2,j);

         % Find global orientation of robot at each motion primitive tail.
         global_theta{i}(:,j) = all_gaits(i).raw_poses(3, tail_idx{i}(1,j));

         % Convert from global frame to body frame.
         rot_mat =  [cos(global_theta{i}(1,j)) -sin(global_theta{i}(1,j)); sin(global_theta{i}(
         delta_poses{i}(1:2,j) = rot_mat'*delta_poses{i}(1:2,j);

         % Convert to twists.
         twists{i}(:,j)  = gait_defs(i).delta_pose_2_twist(delta_poses{i}(:,j), all_gaits(i).tr

         % Find instantaneous center of rotation.
         cent_rot{i}(:,j) = (eye(2) - rot_mat)\delta_poses{i}(1:2, j);

         % Find instantaneous radius of curvature.
         rad_curv{i}(1,j) = norm(cent_rot{i}(:,j));
     end

end
```

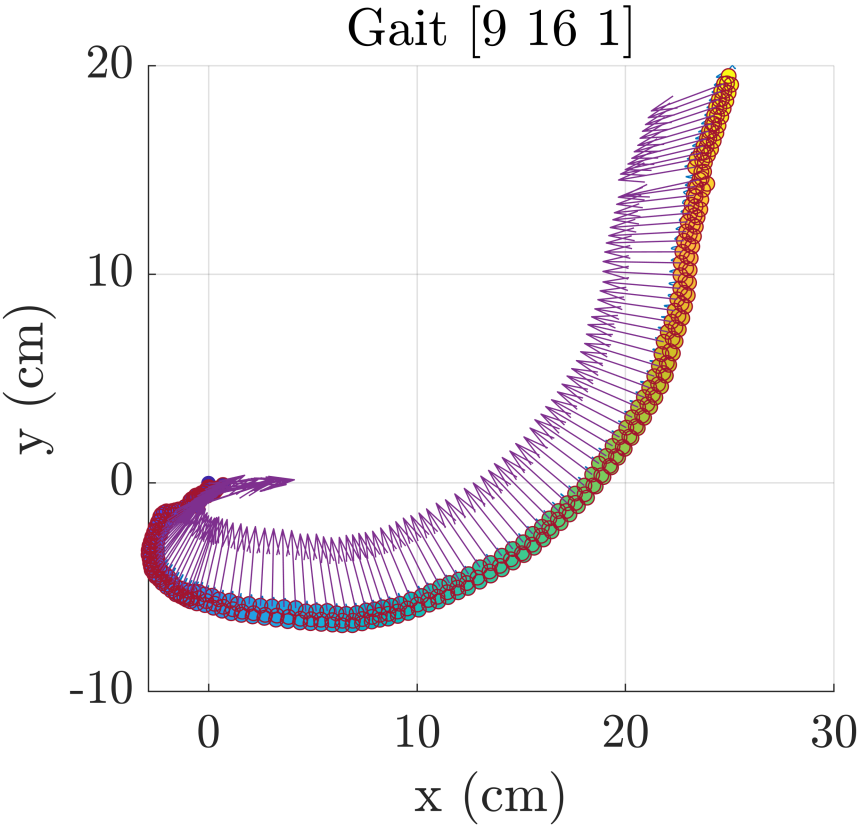
Plot the full motion data for each experiment.

```
for i = 1:n_gaits
    disp(exp_title(i))
    figure
    all_gaits(i).plot;
end
```

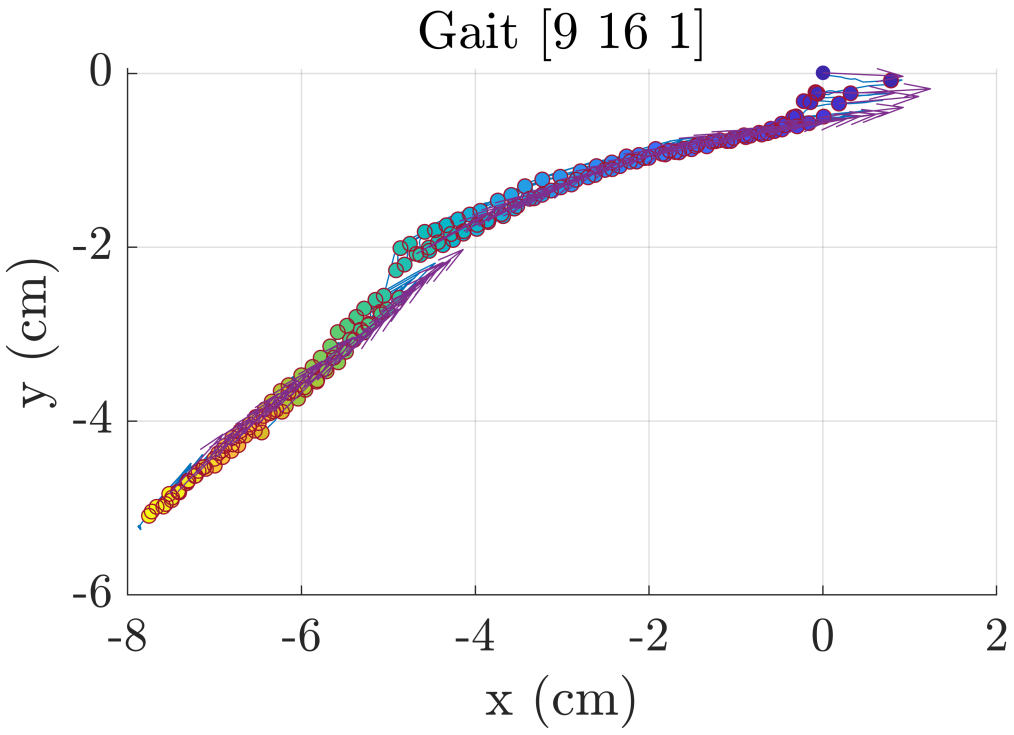
```
Experiment 1 : 120 cycles of Gait B with heavy sheath tether ( left , not following ), trial 1
```



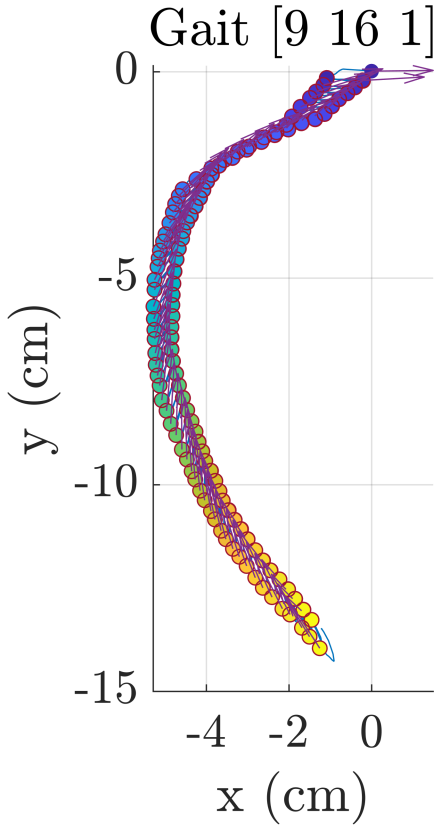
Experiment 2 : 120 cycles of Gait E with heavy sheath tether (left , not following), trial 1



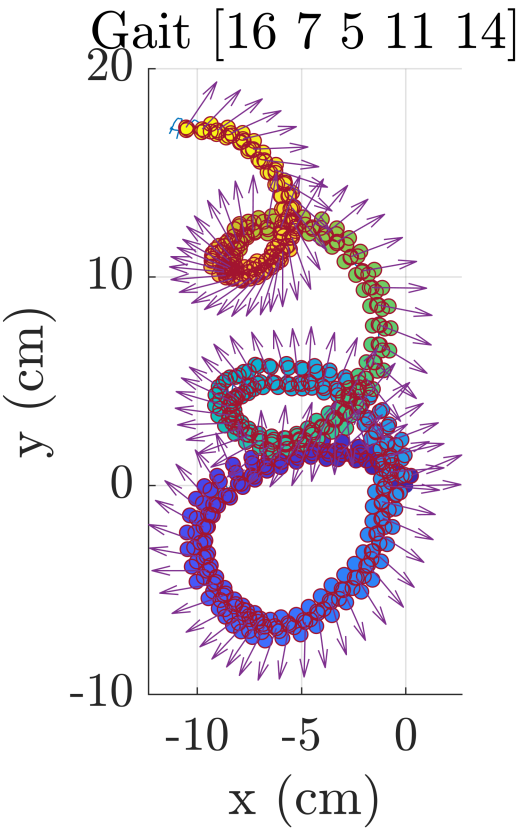
Experiment 3 : 60 cycles of Gait E with heavy sheath tether (left , not following), trial 1



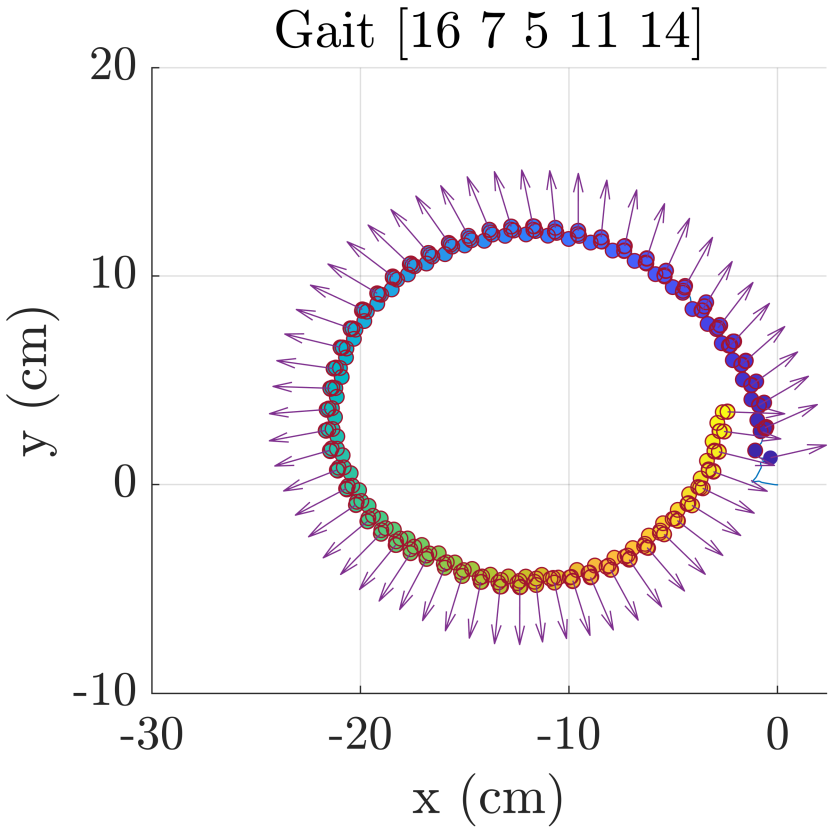
Experiment 4 : 60 cycles of Gait Es with heavy sheath tether (right , not following), trial 1



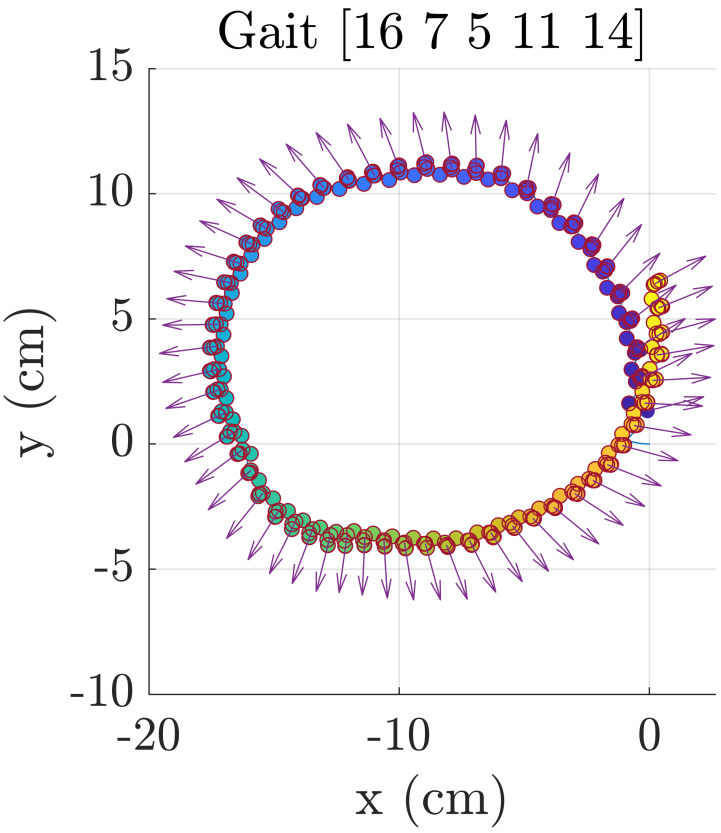
Experiment 5 : 120 cycles of Gait B with light sheath tether (right , not following), trial 1



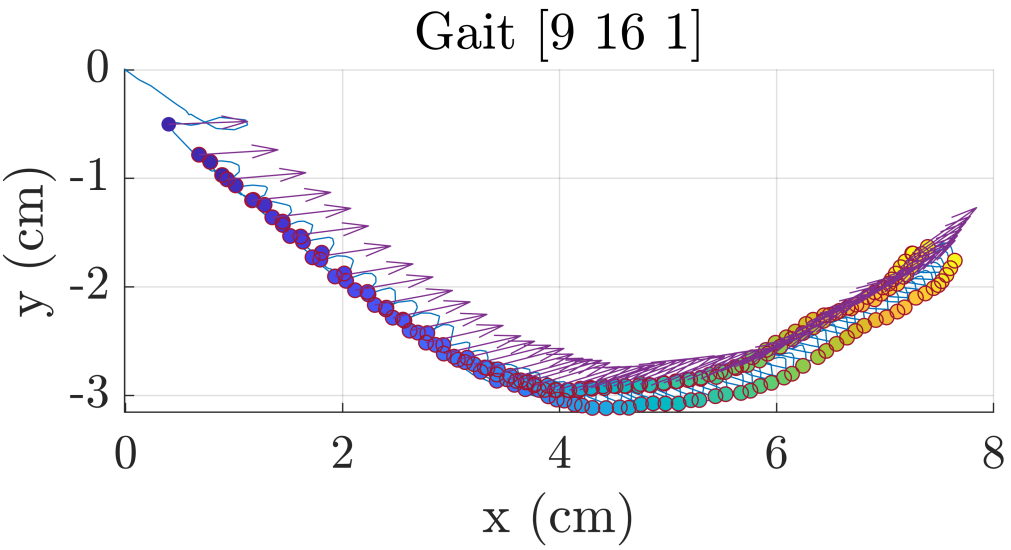
Experiment 6 : 60 cycles of Gait Bs with light sheath tether (left , following), trial 1



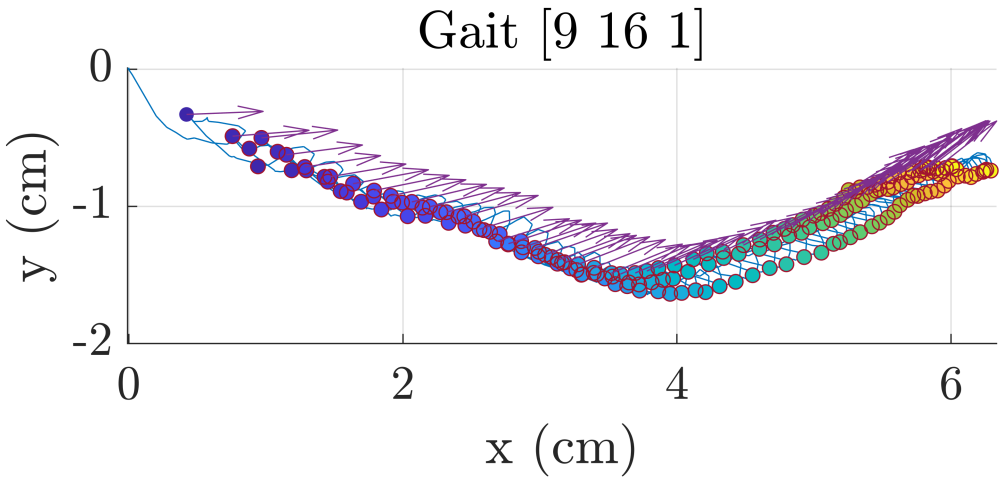
Experiment 7 : 60 cycles of Gait Bs with light sheath tether (right , following), trial 1



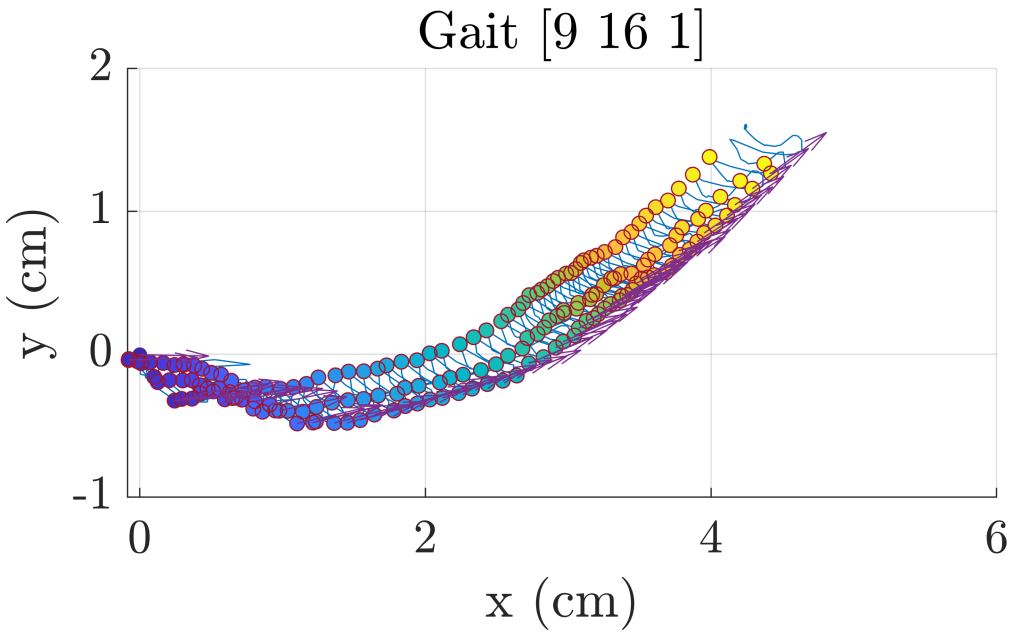
Experiment 8 : 60 cycles of Gait E with light sheath tether (left , not following), trial 1



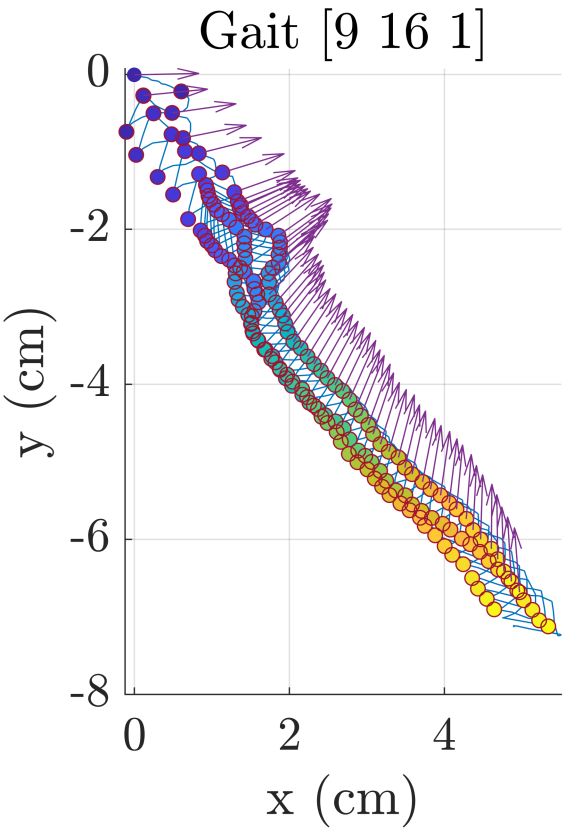
Experiment 9 : 60 cycles of Gait E with light sheath tether (right , not following), trial 1



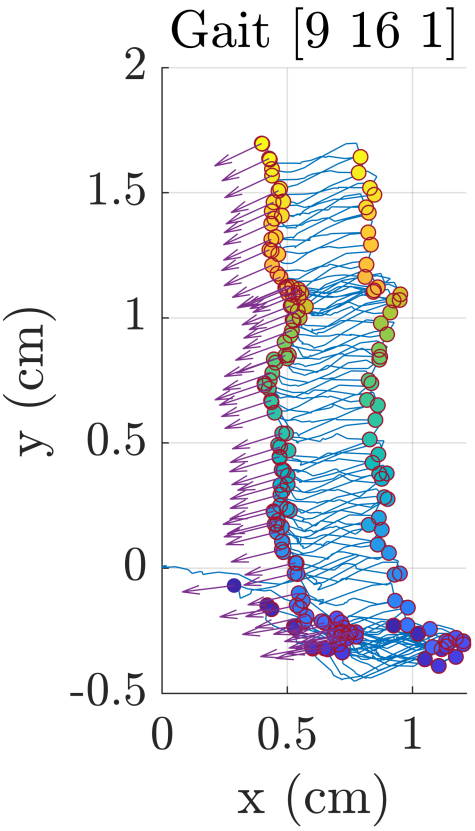
Experiment 10 : 60 cycles of Gait E with no sheath tether (left , not following), trial 1



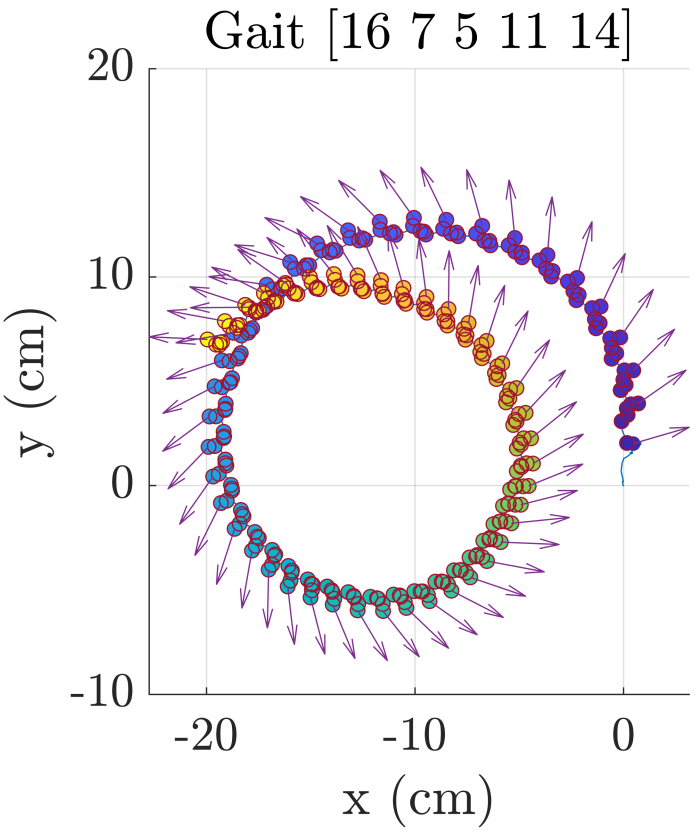
Experiment 11 : 60 cycles of Gait Es with no sheath tether (right , not following), trial 1



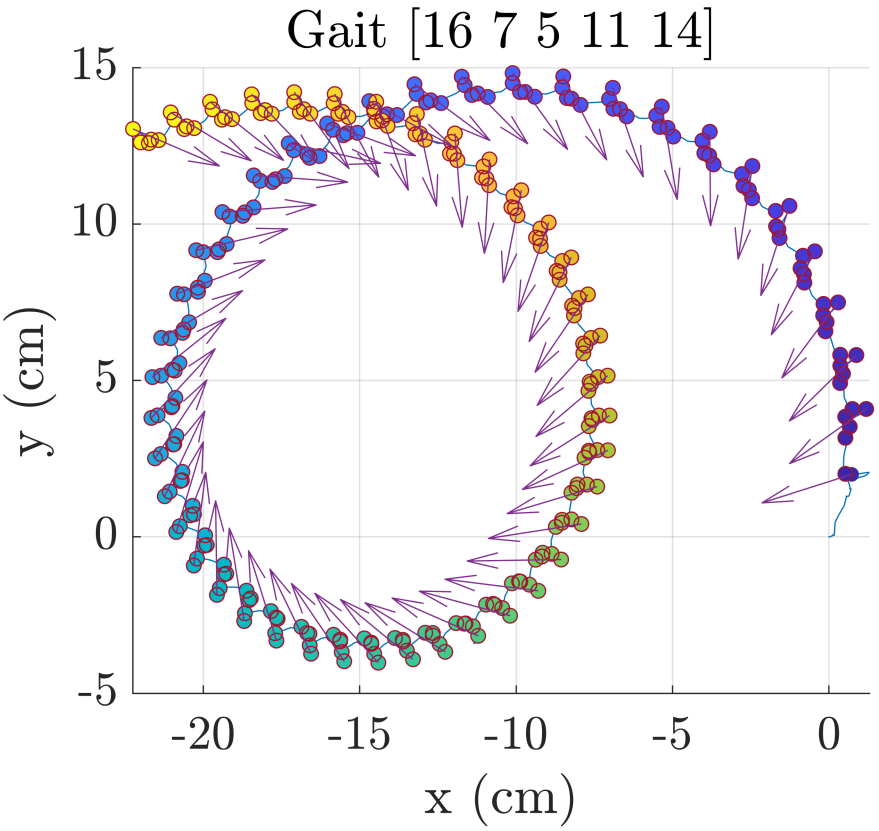
Experiment 12 : 60 cycles of Gait E with no sheath tether (Lf , not following), trial 1



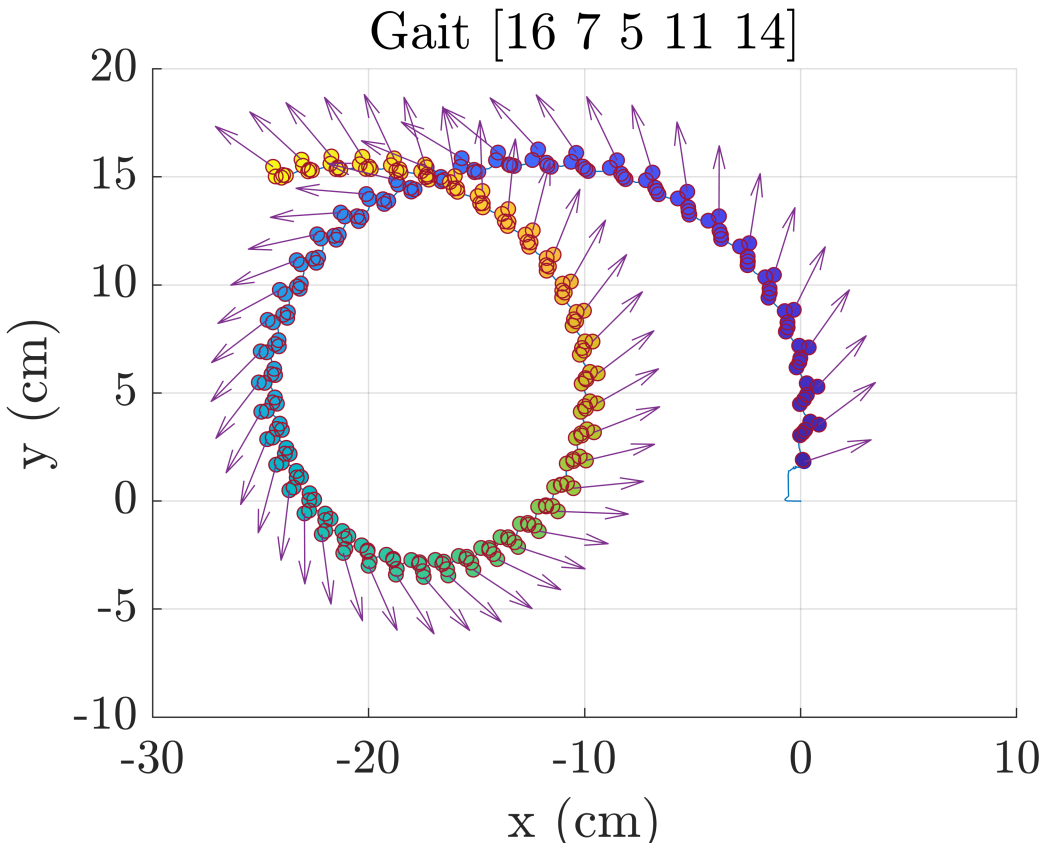
Experiment 13 : 60 cycles of Gait B with light sheath tether (left , following), trial 1



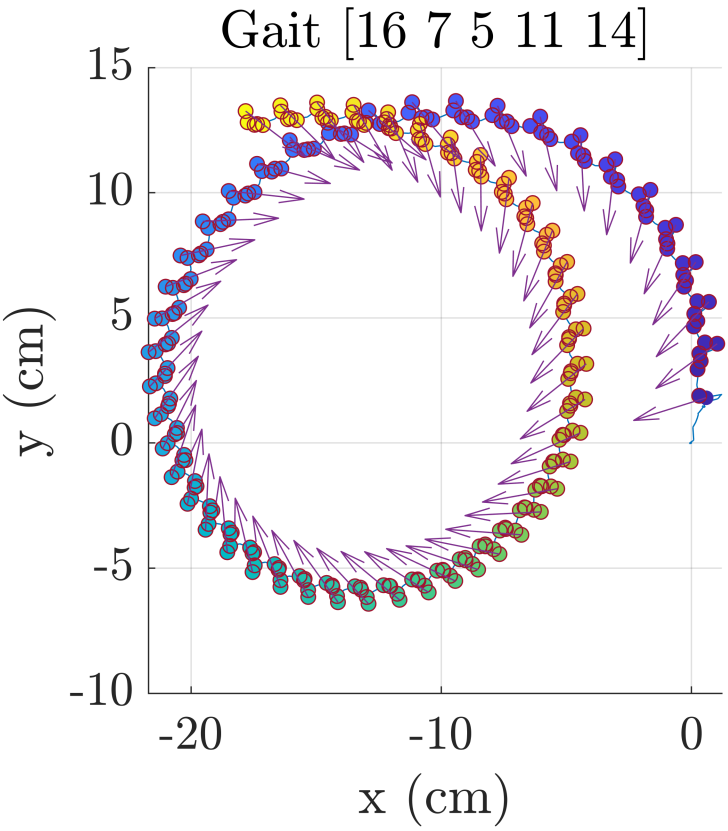
Experiment 14 : 60 cycles of Gait B with light sheath tether (Lf , following), trial 1



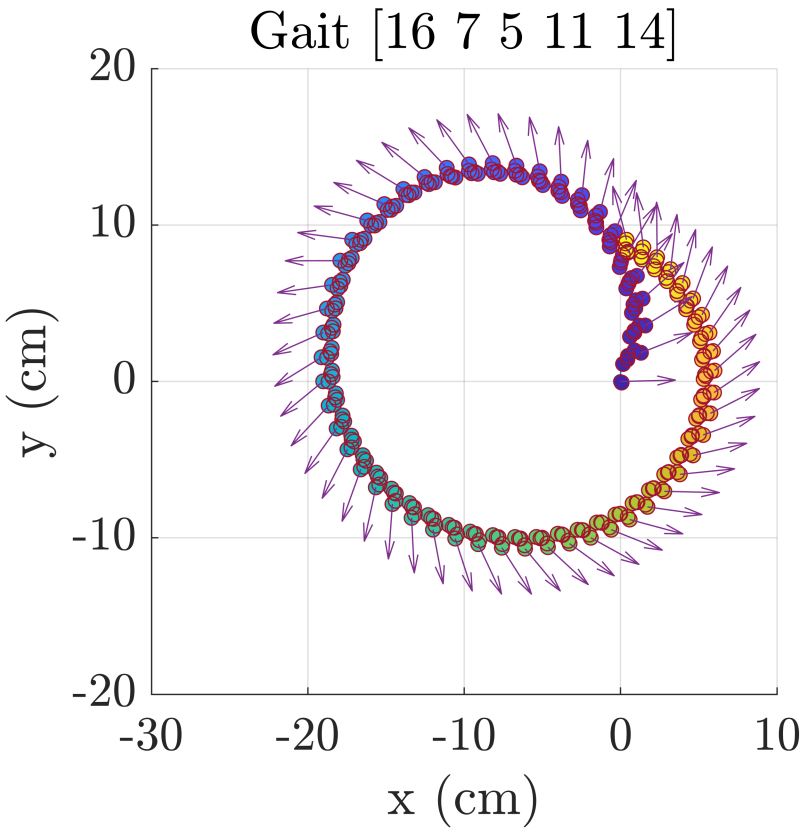
Experiment 15 : 60 cycles of Gait B with light sheath tether (left , not following), trial 1



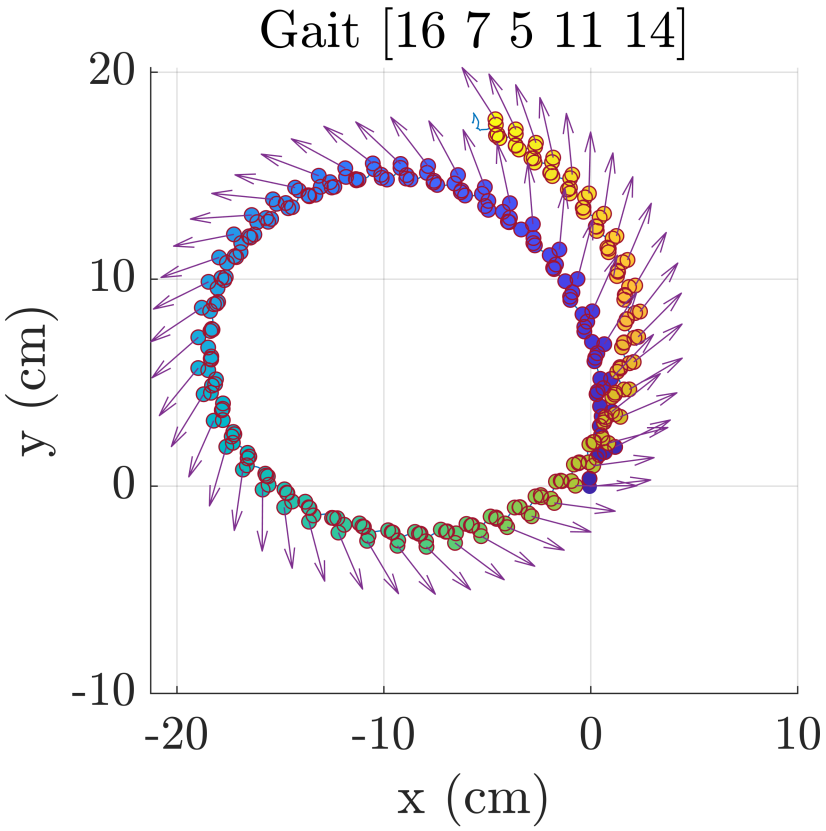
Experiment 16 : 60 cycles of Gait B with light sheath tether (Lf , not following), trial 1



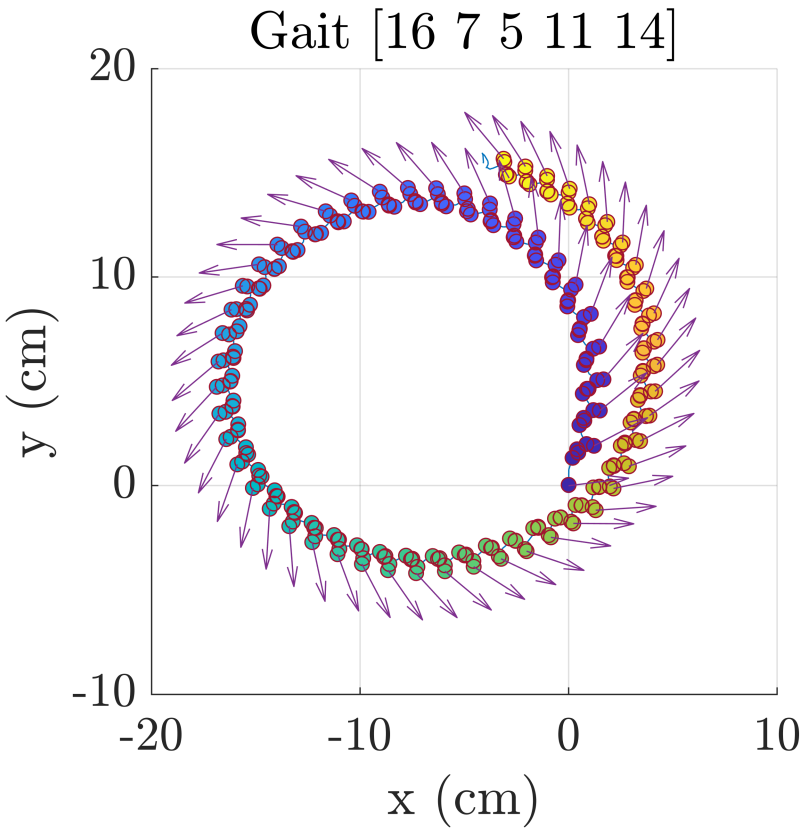
Experiment 17 : 60 cycles of Gait B with light sheath tether (right , not following), trial 1



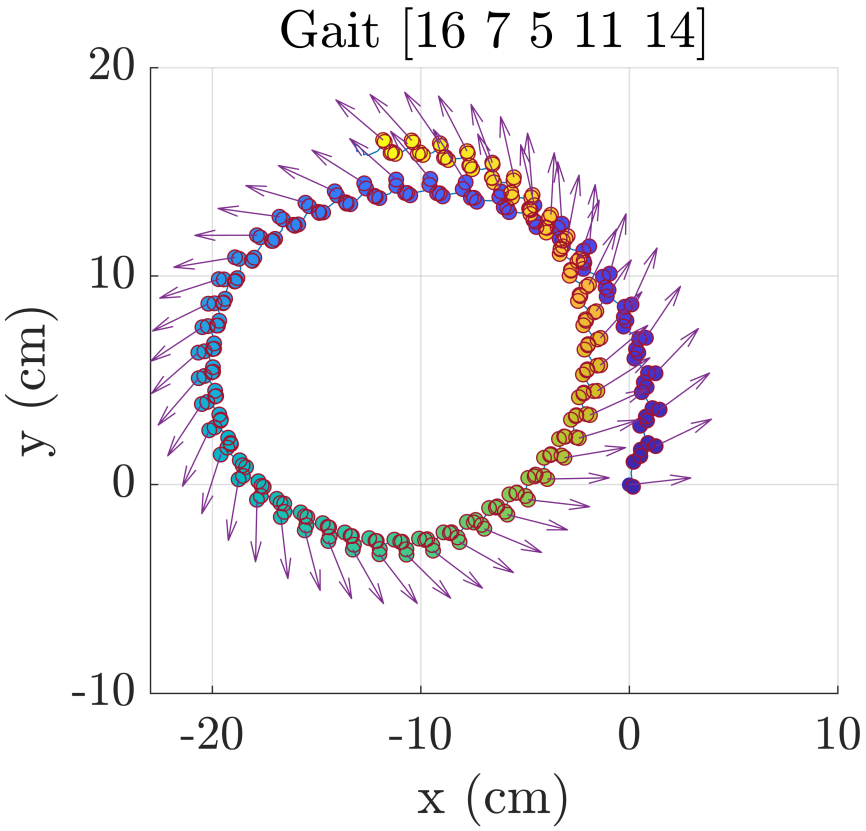
Experiment 18 : 60 cycles of Gait B with no sheath tether (right , following), trial 1



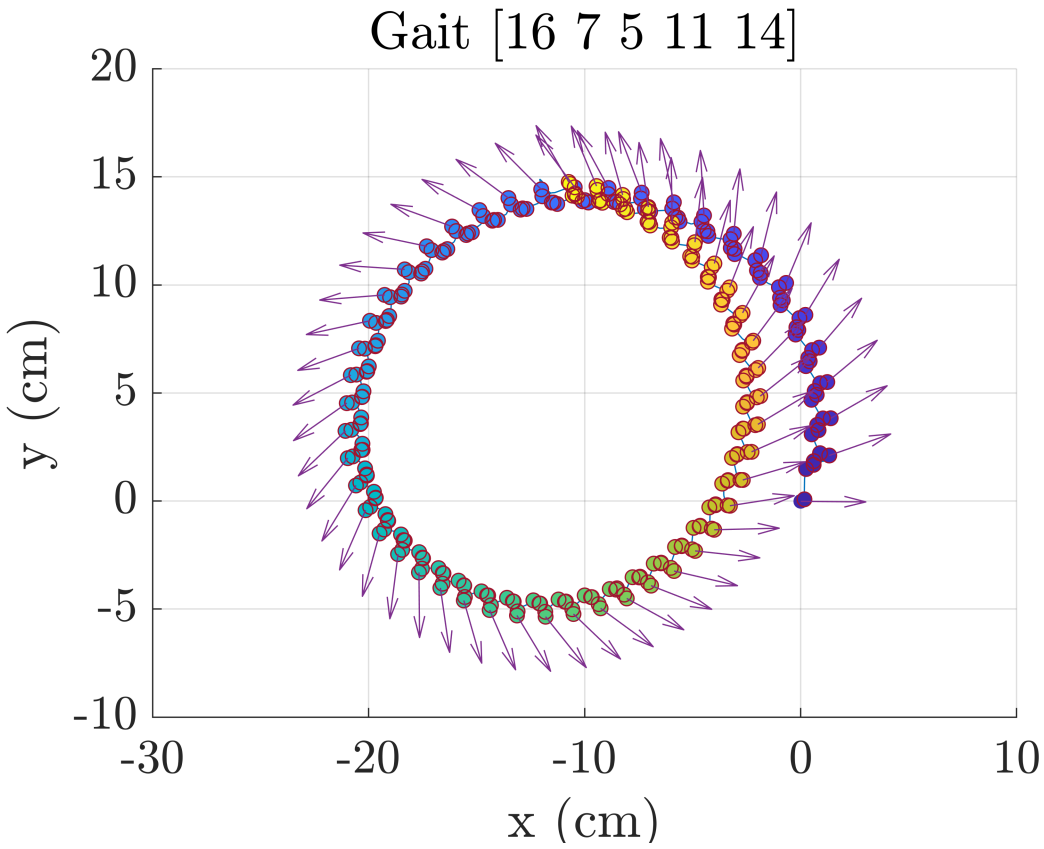
Experiment 19 : 60 cycles of Gait B with no sheath tether (right , following), trial 2



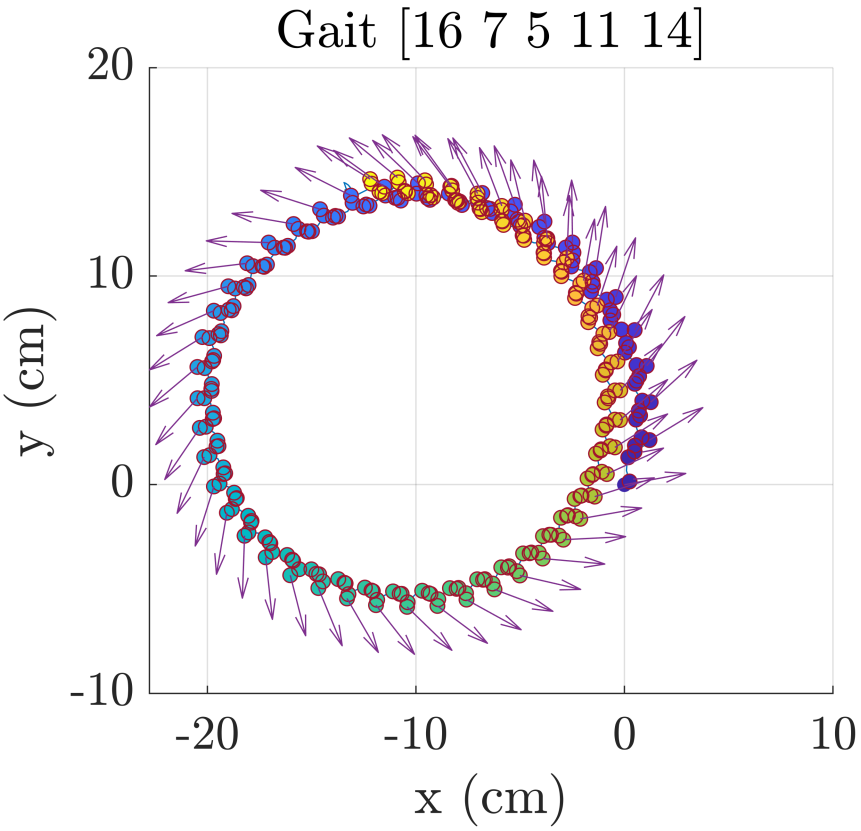
Experiment 20 : 60 cycles of Gait B with no sheath tether (left , following), trial 1



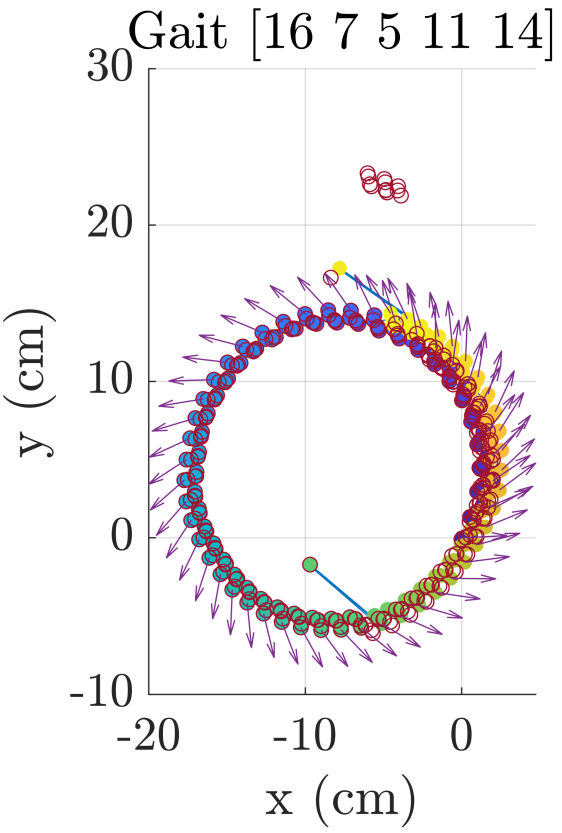
Experiment 21 : 60 cycles of Gait B with no sheath tether (left , following), trial 2



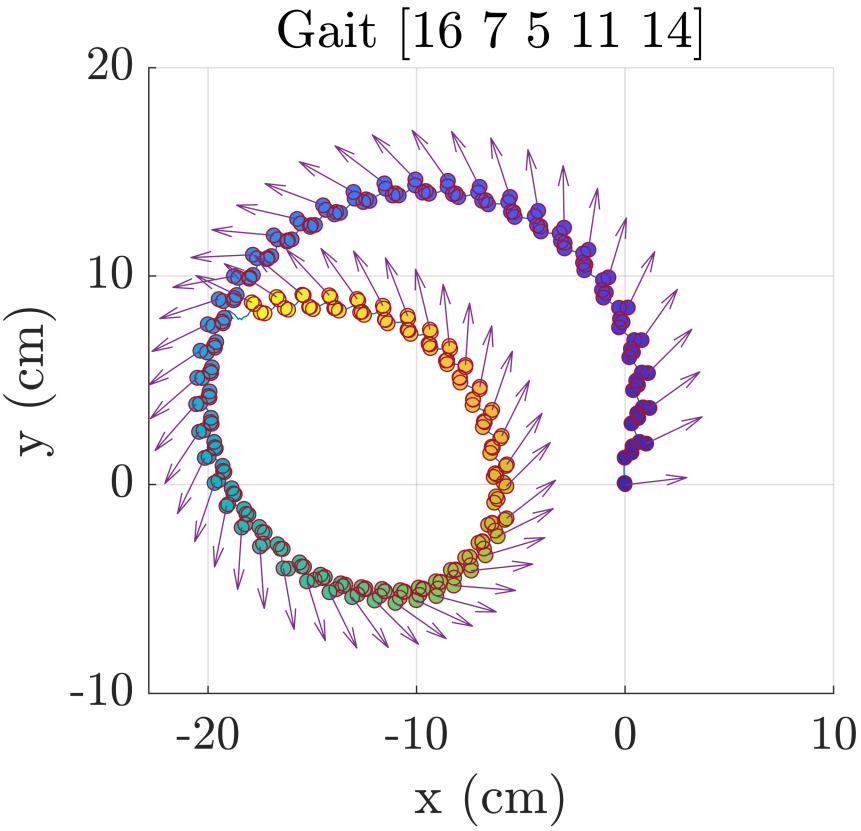
Experiment 22 : 60 cycles of Gait B with no sheath tether (right , not following), trial 1



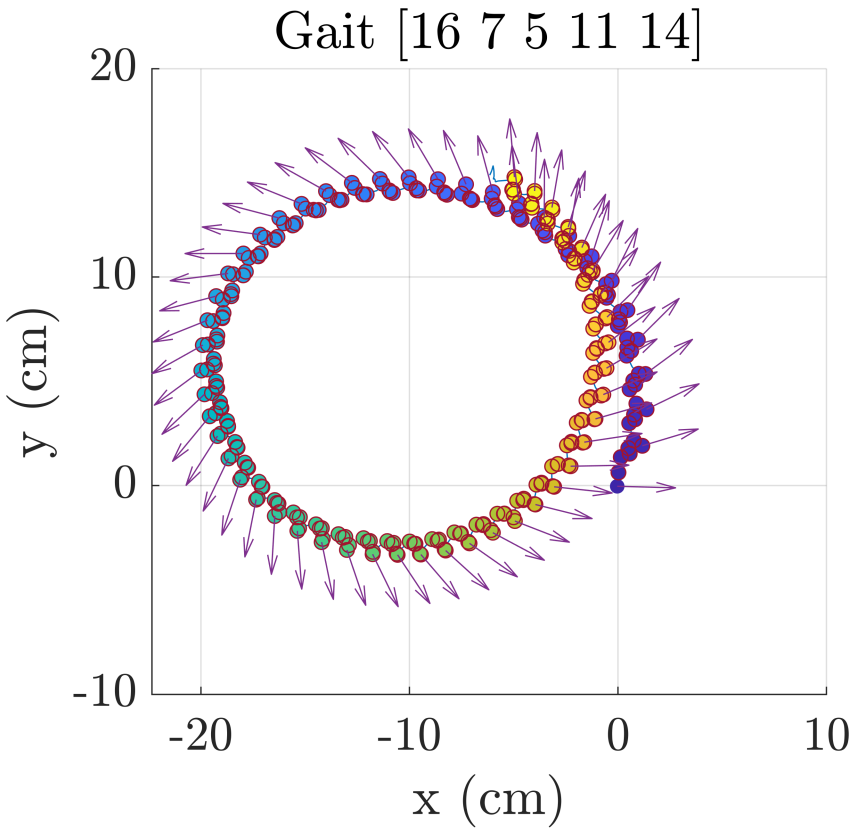
Experiment 23 : 60 cycles of Gait B with no sheath tether (right , not following), trial 2



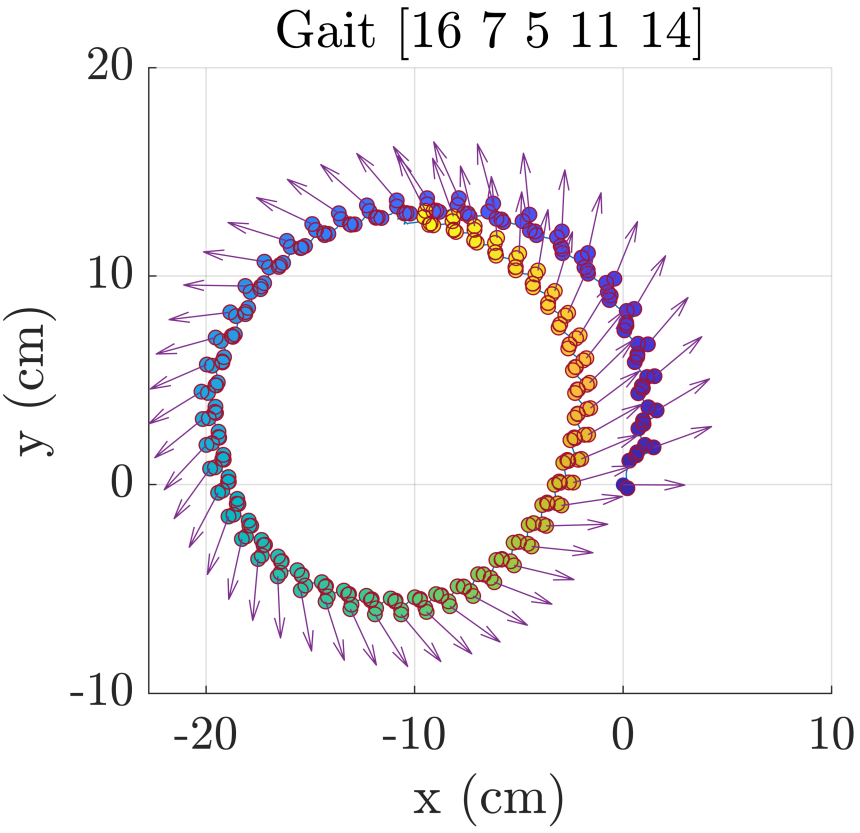
Experiment 24 : 60 cycles of Gait B with no sheath tether (left , not following), trial 1



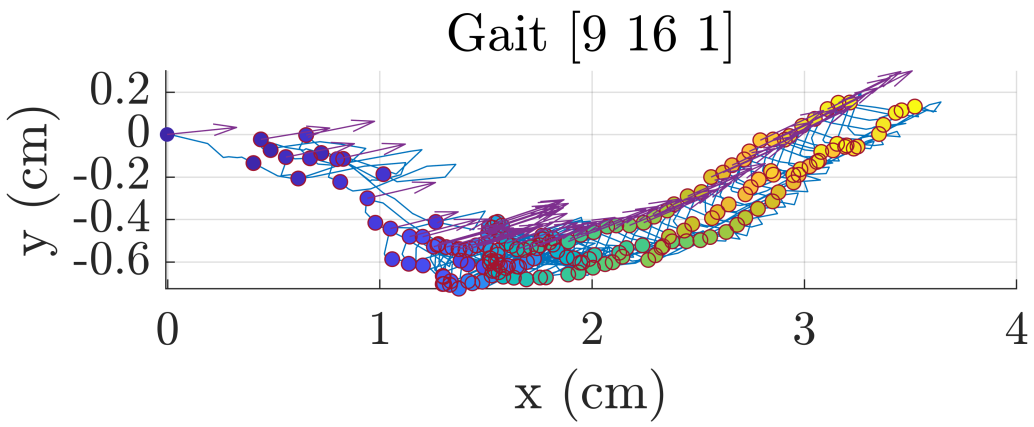
Experiment 25 : 60 cycles of Gait B with no sheath tether (left , not following), trial 2



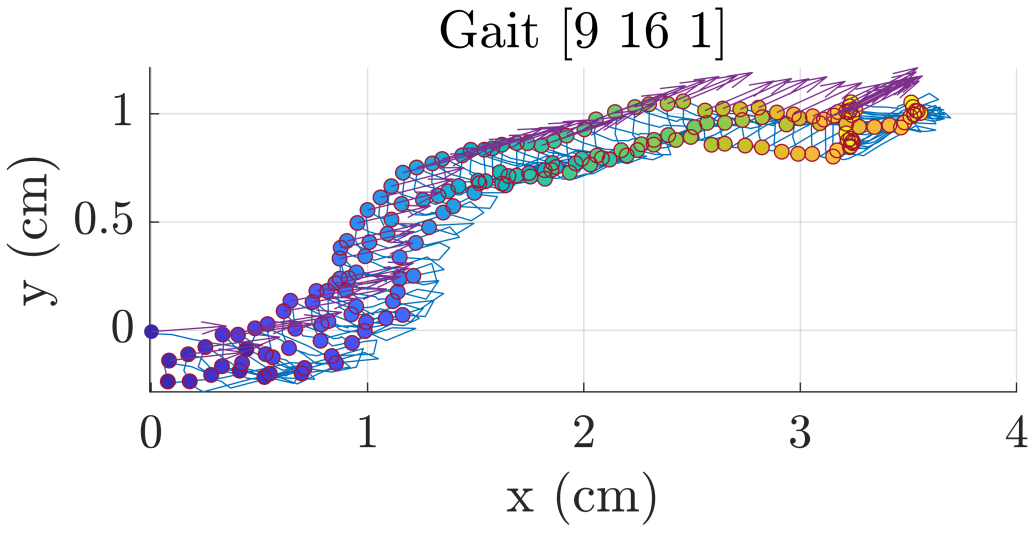
Experiment 26 : 60 cycles of Gait B with light sheath tether (right , following), trial 1



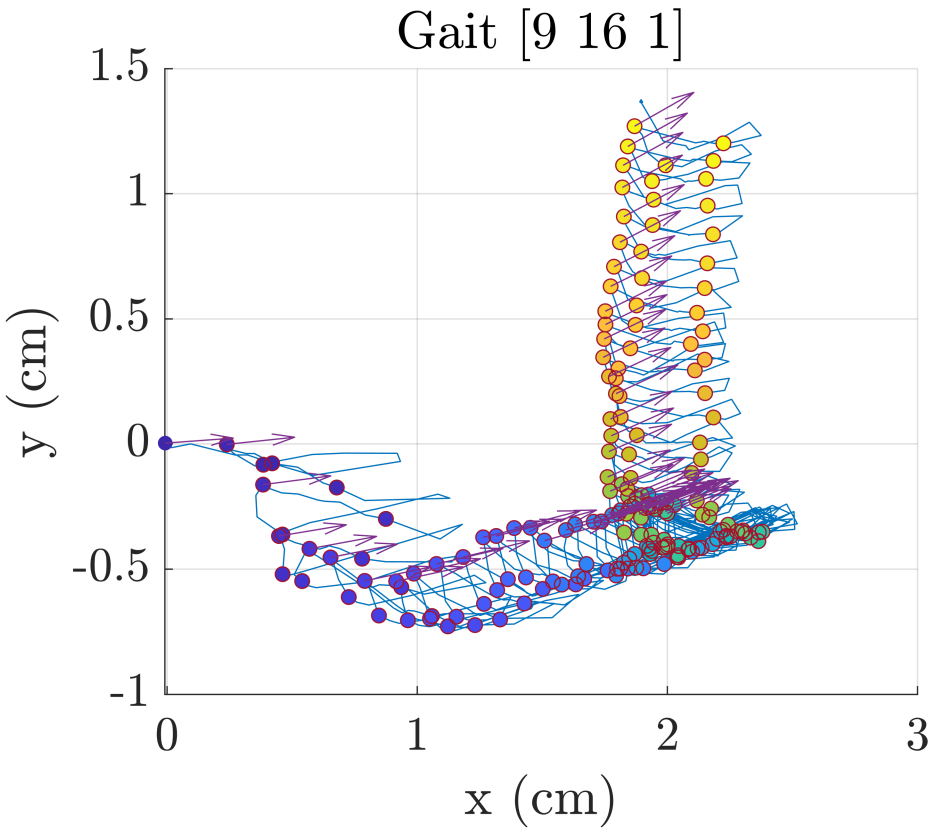
Experiment 27 : 60 cycles of Gait E with no sheath tether (right , not following), trial 1



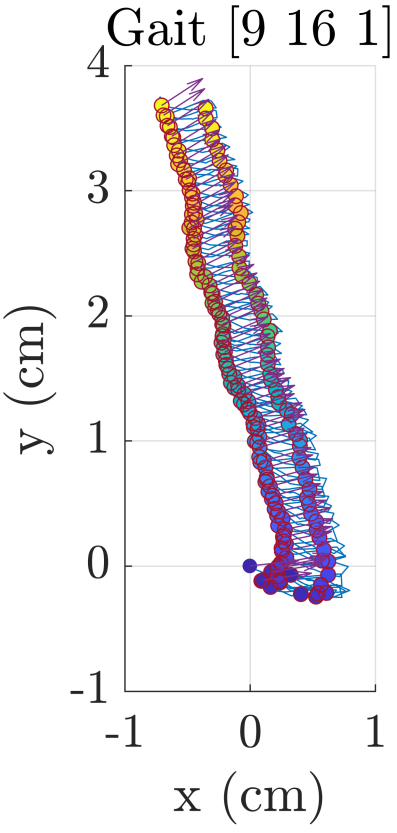
Experiment 28 : 60 cycles of Gait E with no sheath tether (right , not following), trial 2



Experiment 29 : 60 cycles of Gait E with no sheath tether (left , not following), trial 2

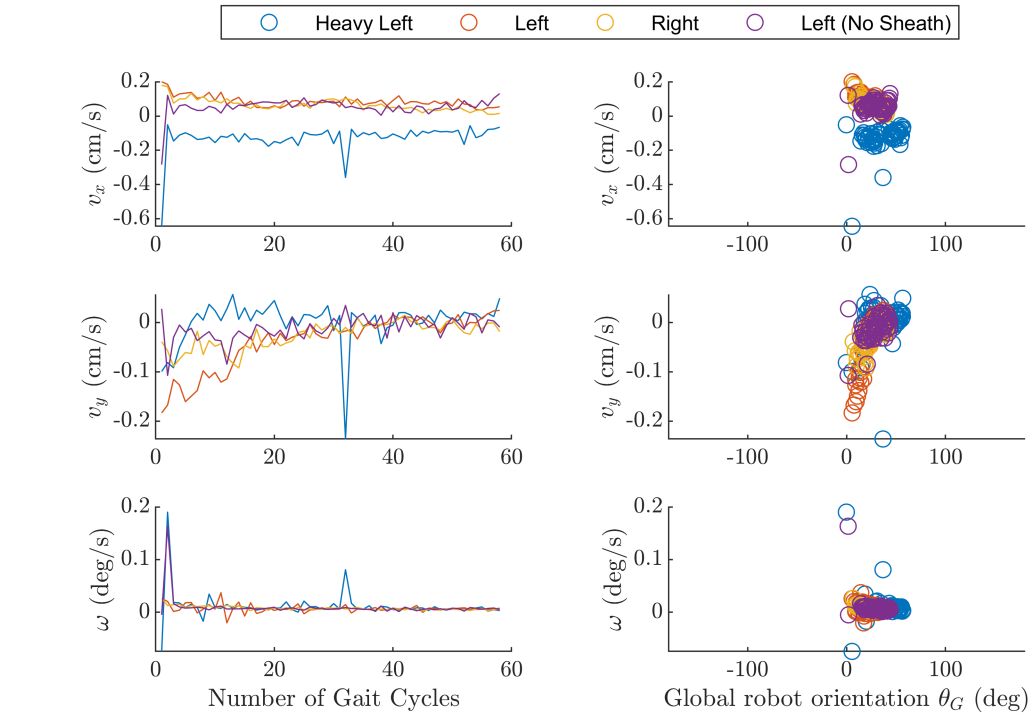


Experiment 30 : 60 cycles of Gait E with no sheath tether (left , not following), trial 3



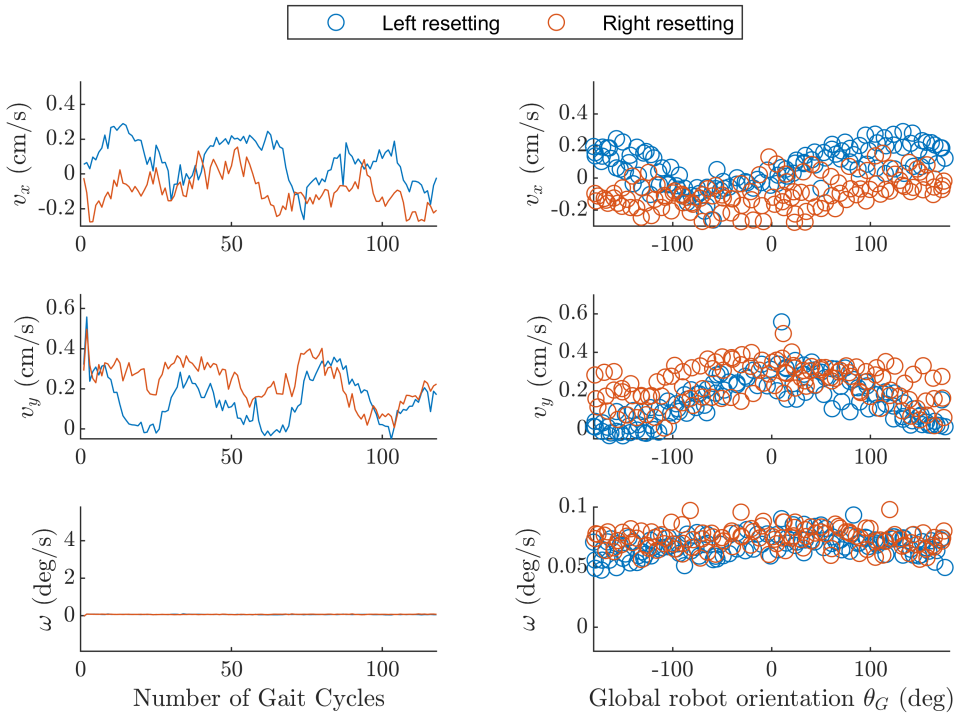
Plot the twist data for each experiment.

```
figure
trial_labels = {'Heavy Left','Left', 'Right', 'Left (No Sheath)'};
t_4 = plot_twists(twists, global_theta, [3,8,9,10], trial_labels,[]);
title(t_4, 'Twist (body velocity) for two experiments of Gait E [9,16,1]', 'FontSize',22)
```

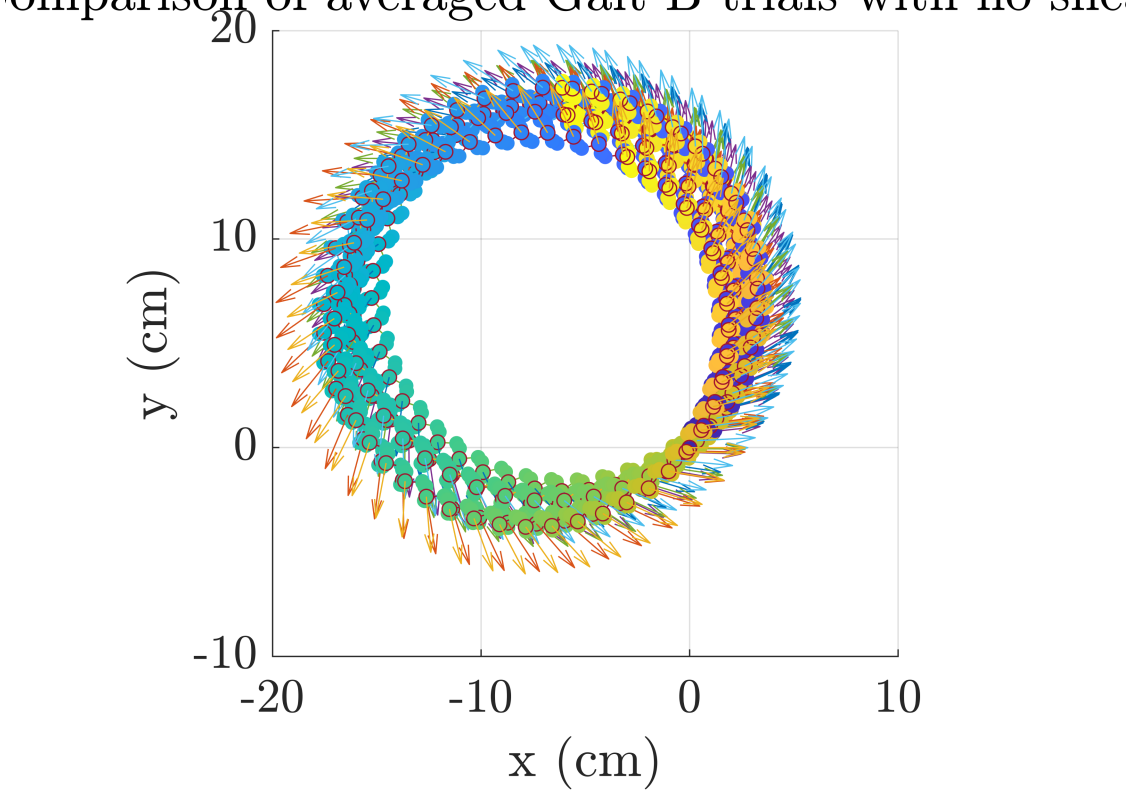



```
figure
ylims = [-.3 .53;
        -.05 .67;
        -1.9 5.8;
        -0.3 0.6;
        -0.05 0.7;
        -0.02 0.1];
trial_labels = {'Left resetting', 'Right resetting'};
t_5 = plot_twists(twists, global_theta, [1,5], trial_labels,ylims);
title(t_5,'Twist (body velocity) for two experiments of Gait B [16,7,5,11,14] with different te
```

for two experiments of Gait B [16,7,5,11,14




```
figure
hold on;
j =1;
for i = [18,19,20,21,22,25]
    gait_library_NS(j) = gaitdef.Gait(all_gaits(i), stab_exp(i).params);
    gait_library_NS(j).gait_name = num2str(i);

    gait_library_NS(j).plot(60)
    j = j+1;
end
title('Comparison of averaged Gait B trials with no sheath')
```




```
if show_markers
    figure
    tiledlayout(5,6)
    for i = 1:n_gaits

    pic_name = gait_names{i};
    pic_name(end-3:end) = [];
    % Plot first image of experiment video.
    file_name = ['data/visualtracking/firstframe_', pic_name, '.jpg'];
    pic = imread(file_name);
    nexttile;
    imshow(pic)
    hold on

    % Plot labeled (i.e., numbered) markers on top of image.
    markers_x(i, :) = stab_exp(i).raw_data(1, 1:3:stab_exp(i).params.n_markers*3-2);
    markers_y(i, :) = stab_exp(i).raw_data(1, 2:3:stab_exp(i).params.n_markers*3-1);
%     if iTrial ~=5
    for m = 1:stab_exp(i).params.n_markers
        text(markers_x(i, m), 1080-markers_y(i, m), num2str(m),'Color','white','FontSize',14);
%     end
    end

    % Adjust size.
     xlim([min(markers_x(i, :))-200, max(markers_x(i, :))+200]);
```

```
     ylim([min(1080 - markers_y(i, :)) - 200, max(1080 - markers_y(i, :))+200]);


end
end
```

```
function twists_plot = plot_twists(twists, global_theta, trial_nums, trial_labels, ylims)

    twists_plot = tiledlayout(3,2);
    ylabels = {'$v_x$ (cm/s)', '$v_y$ (cm/s)', '$\omega$ (deg/s)'};

    % Plot twist components vs number of gait cycles.
    for i = 1:3
        nexttile(2*i - 1);
        hold on;
        for j = 1:length(trial_nums)
            plot(twists{trial_nums(j)}(i,:))
        end
        ylabel(ylabels{i})
        if ~isempty(ylims)
            ylim(ylims(i,:))
        end
    end
    xlabel('Number of Gait Cycles')

    % Plot twist components vs global robot orientation.
    for i = 1:3
        nexttile(2*i);
        hold on;
        for j = 1:length(trial_nums)
            scatter(rad2deg(global_theta{trial_nums(j)}(1,:)),twists{trial_nums(j)}(i,:))
        end
        xlim([-180 180])
        ylabel(ylabels{i})
        if ~isempty(ylims)
            ylim(ylims(3+i,:))
        end
    end
    xlabel('Global robot orientation $\theta_G$ (deg)')

    if length(trial_labels) > 1
        lgd = legend(trial_labels);
        lgd.Orientation = 'horizontal';
        lgd.Layout.Tile = 'north';
    else
        plot_title = ['Twist (body velocity) for', trial_labels];
        title(twists_plot,plot_title, 'FontSize',22)
    end
```

```
end
```