```
% CaveX Honours Project 2023
% Riley Groome, Tyler Groome & Luka Moran
% Preliminary Testing results

% angles corresponding to different blocks of wood
angles = [6.47, 7.65, 8.24, 8.83]; % deg [0 blocks, 2, 3 4]
% extract opt-track csv data
% Tripod Gait data
Tripod_Decline_0 = csvread("CaveX - Tripod - Decline Test (0 blocks of
 wood).csv");
Tripod_Decline_2 = csvread("CaveX - Tripod - Decline Test (2 blocks of
 wood).csv");
Tripod_Decline_3 = csvread("CaveX - Tripod - Decline Test (3 blocks of
 wood).csv");
Tripod_Decline_4 = csvread("CaveX - Tripod - Decline Test (4 blocks of
 wood).csv");

Tripod_Incline_0 = csvread("CaveX - Tripod - Incline Test (0 blocks of
 wood).csv");
Tripod_Incline_2 = csvread("CaveX - Tripod - Incline Test (2 blocks of
 wood).csv");
Tripod_Incline_3 = csvread("CaveX - Tripod - Incline Test (3 blocks of
 wood).csv");
Tripod_Incline_4 = csvread("CaveX - Tripod - Incline Test (4 blocks of
 wood).csv");

% Ripple Gait Data
Ripple_Decline_0 = csvread("CaveX - Ripple - Decline Test (0 blocks of
 wood).csv");
Ripple_Decline_2 = csvread("CaveX - Ripple - Decline Test (2 blocks of
 wood).csv");
Ripple_Decline_3 = csvread("CaveX - Ripple - Decline Test (3 blocks of
 wood).csv");
Ripple_Decline_4 = csvread("CaveX - Ripple - Decline Test (4 blocks of
 wood).csv");

Ripple_Incline_0 = csvread("CaveX - Ripple - Incline Test (0 blocks of
 wood).csv");
Ripple_Incline_2 = csvread("CaveX - Ripple - Incline Test (2 blocks of
 wood).csv");
Ripple_Incline_3 = csvread("CaveX - Ripple - Incline Test (3 blocks of
 wood).csv");
Ripple_Incline_4 = csvread("CaveX - Ripple - Incline Test (4 blocks of
 wood).csv");

% Wave Gait Data
Wave_Decline_0 = csvread("CaveX - Wave - Decline Test (0 blocks of
 wood).csv");
Wave_Decline_2 = csvread("CaveX - Wave - Decline Test (2 blocks of
 wood).csv");
Wave_Decline_3 = csvread("CaveX - Wave - Decline Test (3 blocks of
 wood).csv");
```

```
Wave_Decline_4 = csvread("CaveX - Wave - Decline Test (4 blocks of
 wood).csv");

Wave_Incline_0 = csvread("CaveX - Wave - Incline Test (0 blocks of
 wood).csv");
Wave_Incline_2 = csvread("CaveX - Wave - Incline Test (2 blocks of
 wood).csv");
Wave_Incline_3 = csvread("CaveX - Wave - Incline Test (3 blocks of
 wood).csv");
Wave_Incline_4 = csvread("CaveX - Wave - Incline Test (4 blocks of
 wood).csv");

% Decline Results for each gait and each angle
% work out average velocity for Tripod initial angle (0 blocks)
Time_Tripod_0 = Tripod_Decline_0(:,1); % s
X_Tripod_0 = Tripod_Decline_0(:,2); % m
Y_Tripod_0 = Tripod_Decline_0(:,3); % m
Z_Tripod_0 = Tripod_Decline_0(:,4); % m
delta_t_tripod_0 = Time_Tripod_0(length(Time_Tripod_0))-
Time_Tripod_0(1); % s
delta_x_tripod_0 = X_Tripod_0(length(X_Tripod_0))-X_Tripod_0(1); % m
delta_y_tripod_0 = Y_Tripod_0(length(Y_Tripod_0))-Y_Tripod_0(1); % m
delta_z_tripod_0 = Z_Tripod_0(length(Z_Tripod_0))-Z_Tripod_0(1); % m
Vx_ave_Tripod_0 = delta_x_tripod_0/delta_t_tripod_0; % m/s [average x
 velocity]
Vy_ave_Tripod_0 = delta_y_tripod_0/delta_t_tripod_0; % m/s [average x
 velocity]
Vz_ave_Tripod_0 = delta_z_tripod_0/delta_t_tripod_0; % m/s [average x
 velocity]
V_ave_tripod_0 =
 sqrt((Vx_ave_Tripod_0)^2+(Vy_ave_Tripod_0)^2+(Vz_ave_Tripod_0)^2);

% work out average velocity for Tripod second angle (2 blocks)
Time_Tripod_2 = Tripod_Decline_2(:,1); % s
X_Tripod_2 = Tripod_Decline_2(:,2); % m
Y_Tripod_2 = Tripod_Decline_2(:,3); % m
Z_Tripod_2 = Tripod_Decline_2(:,4); % m
delta_t_tripod_2 = Time_Tripod_2(length(Time_Tripod_2))-
Time_Tripod_2(1); % s
delta_x_tripod_2 = X_Tripod_2(length(X_Tripod_2))-X_Tripod_2(1); % m
delta_y_tripod_2 = Y_Tripod_2(length(Y_Tripod_2))-Y_Tripod_2(1); % m
delta_z_tripod_2 = Z_Tripod_2(length(Z_Tripod_2))-Z_Tripod_2(1); % m
Vx_ave_Tripod_2 = delta_x_tripod_2/delta_t_tripod_2; % m/s [average x
 velocity]
Vy_ave_Tripod_2 = delta_y_tripod_2/delta_t_tripod_2; % m/s [average x
 velocity]
Vz_ave_Tripod_2 = delta_z_tripod_2/delta_t_tripod_2; % m/s [average x
 velocity]
V_ave_tripod_2 =
 sqrt((Vx_ave_Tripod_2)^2+(Vy_ave_Tripod_2)^2+(Vz_ave_Tripod_2)^2);

% work out average velocity for Tripod third angle (3 blocks)
Time_Tripod_3 = Tripod_Decline_3(:,1); % s
X_Tripod_3 = Tripod_Decline_3(:,2); % m
```

```
Y_Tripod_3 = Tripod_Decline_3(:,3); % m
Z_Tripod_3 = Tripod_Decline_3(:,4); % m
delta_t_tripod_3 = Time_Tripod_3(length(Time_Tripod_3))-
Time_Tripod_3(1); % s
delta_x_tripod_3 = X_Tripod_3(length(X_Tripod_3))-X_Tripod_3(1); % m
delta_y_tripod_3 = Y_Tripod_3(length(Y_Tripod_3))-Y_Tripod_3(1); % m
delta_z_tripod_3 = Z_Tripod_3(length(Z_Tripod_3))-Z_Tripod_3(1); % m
Vx_ave_Tripod_3 = delta_x_tripod_3/delta_t_tripod_3; % m/s [average x
 velocity]
Vy_ave_Tripod_3 = delta_y_tripod_3/delta_t_tripod_3; % m/s [average x
 velocity]
Vz_ave_Tripod_3 = delta_z_tripod_3/delta_t_tripod_3; % m/s [average x
 velocity]
V_ave_tripod_3 =
 sqrt((Vx_ave_Tripod_3)^2+(Vy_ave_Tripod_3)^2+(Vz_ave_Tripod_3)^2);

% work out average velocity for Tripod fourth angle (4 blocks)
Time_Tripod_4 = Tripod_Decline_4(:,1); % s
X_Tripod_4 = Tripod_Decline_4(:,2); % m
Y_Tripod_4 = Tripod_Decline_4(:,3); % m
Z_Tripod_4 = Tripod_Decline_4(:,4); % m
delta_t_tripod_4 = Time_Tripod_4(length(Time_Tripod_4))-
Time_Tripod_4(1); % s
delta_x_tripod_4 = X_Tripod_4(length(X_Tripod_4))-X_Tripod_4(1); % m
delta_y_tripod_4 = Y_Tripod_4(length(Y_Tripod_4))-Y_Tripod_4(1); % m
delta_z_tripod_4 = Z_Tripod_4(length(Z_Tripod_4))-Z_Tripod_4(1); % m
Vx_ave_Tripod_4 = delta_x_tripod_4/delta_t_tripod_4; % m/s [average x
 velocity]
Vy_ave_Tripod_4 = delta_y_tripod_4/delta_t_tripod_4; % m/s [average x
 velocity]
Vz_ave_Tripod_4 = delta_z_tripod_4/delta_t_tripod_4; % m/s [average x
 velocity]
V_ave_tripod_4 =
 sqrt((Vx_ave_Tripod_4)^2+(Vy_ave_Tripod_4)^2+(Vz_ave_Tripod_4)^2);

% work out average velocity for Ripple initial angle (0 blocks)
Time_Ripple_0 = Ripple_Decline_0(:,1); % s
X_Ripple_0 = Ripple_Decline_0(:,2); % m
Y_Ripple_0 = Ripple_Decline_0(:,3); % m
Z_Ripple_0 = Ripple_Decline_0(:,4); % m
delta_t_ripple_0 = Time_Ripple_0(length(Time_Ripple_0))-
Time_Ripple_0(1); % s
delta_x_ripple_0 = X_Ripple_0(length(X_Ripple_0))-X_Ripple_0(1); % m
delta_y_ripple_0 = Y_Ripple_0(length(Y_Ripple_0))-Y_Ripple_0(1); % m
delta_z_ripple_0 = Z_Ripple_0(length(Z_Ripple_0))-Z_Ripple_0(1); % m
Vx_ave_Ripple_0 = delta_x_ripple_0/delta_t_ripple_0; % m/s [average x
 velocity]
Vy_ave_Ripple_0 = delta_y_ripple_0/delta_t_ripple_0; % m/s [average x
 velocity]
Vz_ave_Ripple_0 = delta_z_ripple_0/delta_t_ripple_0; % m/s [average x
 velocity]
V_ave_ripple_0 =
 sqrt((Vx_ave_Ripple_0)^2+(Vy_ave_Ripple_0)^2+(Vz_ave_Ripple_0)^2);
```

```
% work out average velocity for Ripple second angle (2 blocks)
Time_Ripple_2 = Ripple_Decline_2(:,1); % s
X_Ripple_2 = Ripple_Decline_2(:,2); % m
Y_Ripple_2 = Ripple_Decline_2(:,3); % m
Z_Ripple_2 = Ripple_Decline_2(:,4); % m
delta_t_ripple_2 = Time_Ripple_2(length(Time_Ripple_2))-
Time_Ripple_2(1); % s
delta_x_ripple_2 = X_Ripple_2(length(X_Ripple_2))-X_Ripple_2(1); % m
delta_y_ripple_2 = Y_Ripple_2(length(Y_Ripple_2))-Y_Ripple_2(1); % m
delta_z_ripple_2 = Z_Ripple_2(length(Z_Ripple_2))-Z_Ripple_2(1); % m
Vx_ave_Ripple_2 = delta_x_ripple_2/delta_t_ripple_2; % m/s [average x
 velocity]
Vy_ave_Ripple_2 = delta_y_ripple_2/delta_t_ripple_2; % m/s [average x
 velocity]
Vz_ave_Ripple_2 = delta_z_ripple_2/delta_t_ripple_2; % m/s [average x
 velocity]
V_ave_ripple_2 =
 sqrt((Vx_ave_Ripple_2)^2+(Vy_ave_Ripple_2)^2+(Vz_ave_Ripple_2)^2);

% work out average velocity for Ripple third angle (2 blocks)
Time_Ripple_3 = Ripple_Decline_3(:,1); % s
X_Ripple_3 = Ripple_Decline_3(:,2); % m
Y_Ripple_3 = Ripple_Decline_3(:,3); % m
Z_Ripple_3 = Ripple_Decline_3(:,4); % m
delta_t_ripple_3 = Time_Ripple_3(length(Time_Ripple_3))-
Time_Ripple_3(1); % s
delta_x_ripple_3 = X_Ripple_3(length(X_Ripple_3))-X_Ripple_3(1); % m
delta_y_ripple_3 = Y_Ripple_3(length(Y_Ripple_3))-Y_Ripple_3(1); % m
delta_z_ripple_3 = Z_Ripple_3(length(Z_Ripple_3))-Z_Ripple_3(1); % m
Vx_ave_Ripple_3 = delta_x_ripple_3/delta_t_ripple_3; % m/s [average x
 velocity]
Vy_ave_Ripple_3 = delta_y_ripple_3/delta_t_ripple_3; % m/s [average x
 velocity]
Vz_ave_Ripple_3 = delta_z_ripple_3/delta_t_ripple_3; % m/s [average x
 velocity]
V_ave_ripple_3 =
 sqrt((Vx_ave_Ripple_3)^2+(Vy_ave_Ripple_3)^2+(Vz_ave_Ripple_3)^2);

% work out average velocity for Ripple fourth angle (4 blocks)
Time_Ripple_4 = Ripple_Decline_4(:,1); % s
X_Ripple_4 = Ripple_Decline_4(:,2); % m
Y_Ripple_4 = Ripple_Decline_4(:,3); % m
Z_Ripple_4 = Ripple_Decline_4(:,4); % m
delta_t_ripple_4 = Time_Ripple_4(length(Time_Ripple_4))-
Time_Ripple_4(1); % s
delta_x_ripple_4 = X_Ripple_4(length(X_Ripple_4))-X_Ripple_4(1); % m
delta_y_ripple_4 = Y_Ripple_4(length(Y_Ripple_4))-Y_Ripple_4(1); % m
delta_z_ripple_4 = Z_Ripple_4(length(Z_Ripple_4))-Z_Ripple_4(1); % m
Vx_ave_Ripple_4 = delta_x_ripple_4/delta_t_ripple_4; % m/s [average x
 velocity]
Vy_ave_Ripple_4 = delta_y_ripple_4/delta_t_ripple_4; % m/s [average x
 velocity]
Vz_ave_Ripple_4 = delta_z_ripple_4/delta_t_ripple_4; % m/s [average x
 velocity]
```

```
V_ave_ripple_4 =
 sqrt((Vx_ave_Ripple_4)^2+(Vy_ave_Ripple_4)^2+(Vz_ave_Ripple_4)^2);

% work out average velocity for Wave gait initial angle (0 blocks)
Time_Wave_0 = Wave_Decline_0(:,1); % s
X_Wave_0 = Wave_Decline_0(:,2); % m
Y_Wave_0 = Wave_Decline_0(:,3); % m
Z_Wave_0 = Wave_Decline_0(:,4); % m
delta_t_wave_0 = Time_Wave_0(length(Time_Wave_0))-Time_Wave_0(1); % s
delta_x_wave_0 = X_Wave_0(length(X_Wave_0))-X_Wave_0(1); % m
delta_y_wave_0 = Y_Wave_0(length(Y_Wave_0))-Y_Wave_0(1); % m
delta_z_wave_0 = Z_Wave_0(length(Z_Wave_0))-Z_Wave_0(1); % m
Vx_ave_Wave_0 = delta_x_wave_0/delta_t_wave_0; % m/s [average x
 velocity]
Vy_ave_Wave_0 = delta_y_wave_0/delta_t_wave_0; % m/s [average x
 velocity]
Vz_ave_Wave_0 = delta_z_wave_0/delta_t_wave_0; % m/s [average x
 velocity]
V_ave_wave_0 =
 sqrt((Vx_ave_Wave_0)^2+(Vy_ave_Wave_0)^2+(Vz_ave_Wave_0)^2);

% work out average velocity for Wave gait second angle (2 blocks)
Time_Wave_2 = Wave_Decline_2(:,1); % s
X_Wave_2 = Wave_Decline_2(:,2); % m
Y_Wave_2 = Wave_Decline_2(:,3); % m
Z_Wave_2 = Wave_Decline_2(:,4); % m
delta_t_wave_2 = Time_Wave_2(length(Time_Wave_2))-Time_Wave_2(1); % s
delta_x_wave_2 = X_Wave_2(length(X_Wave_2))-X_Wave_2(1); % m
delta_y_wave_2 = Y_Wave_2(length(Y_Wave_2))-Y_Wave_2(1); % m
delta_z_wave_2 = Z_Wave_2(length(Z_Wave_2))-Z_Wave_2(1); % m
Vx_ave_Wave_2 = delta_x_wave_2/delta_t_wave_2; % m/s [average x
 velocity]
Vy_ave_Wave_2 = delta_y_wave_2/delta_t_wave_2; % m/s [average x
 velocity]
Vz_ave_Wave_2 = delta_z_wave_2/delta_t_wave_2; % m/s [average x
 velocity]
V_ave_wave_2 =
 sqrt((Vx_ave_Wave_2)^2+(Vy_ave_Wave_2)^2+(Vz_ave_Wave_2)^2);

% work out average velocity for Wave gait third angle (3 blocks)
Time_Wave_3 = Wave_Decline_3(:,1); % s
X_Wave_3 = Wave_Decline_3(:,2); % m
Y_Wave_3 = Wave_Decline_3(:,3); % m
Z_Wave_3 = Wave_Decline_3(:,4); % m
delta_t_wave_3 = Time_Wave_3(length(Time_Wave_3))-Time_Wave_3(1); % s
delta_x_wave_3 = X_Wave_3(length(X_Wave_3))-X_Wave_3(1); % m
delta_y_wave_3 = Y_Wave_3(length(Y_Wave_3))-Y_Wave_3(1); % m
delta_z_wave_3 = Z_Wave_3(length(Z_Wave_3))-Z_Wave_3(1); % m
Vx_ave_Wave_3 = delta_x_wave_3/delta_t_wave_3; % m/s [average x
 velocity]
Vy_ave_Wave_3 = delta_y_wave_3/delta_t_wave_3; % m/s [average x
 velocity]
Vz_ave_Wave_3 = delta_z_wave_3/delta_t_wave_3; % m/s [average x
 velocity]
```

```
V_ave_wave_3 =
 sqrt((Vx_ave_Wave_3)^2+(Vy_ave_Wave_3)^2+(Vz_ave_Wave_3)^2);

% work out average velocity for Wave gait fourth angle (4 blocks)
Time_Wave_4 = Wave_Decline_4(:,1); % s
X_Wave_4 = Wave_Decline_4(:,2); % m
Y_Wave_4 = Wave_Decline_4(:,3); % m
Z_Wave_4 = Wave_Decline_4(:,4); % m
delta_t_wave_4 = Time_Wave_4(length(Time_Wave_4))-Time_Wave_4(1); % s
delta_x_wave_4 = X_Wave_4(length(X_Wave_4))-X_Wave_4(1); % m
delta_y_wave_4 = Y_Wave_4(length(Y_Wave_4))-Y_Wave_4(1); % m
delta_z_wave_4 = Z_Wave_4(length(Z_Wave_4))-Z_Wave_4(1); % m
Vx_ave_Wave_4 = delta_x_wave_4/delta_t_wave_4; % m/s [average x
 velocity]
Vy_ave_Wave_4 = delta_y_wave_4/delta_t_wave_4; % m/s [average x
 velocity]
Vz_ave_Wave_4 = delta_z_wave_4/delta_t_wave_4; % m/s [average x
 velocity]
V_ave_wave_4 =
 sqrt((Vx_ave_Wave_4)^2+(Vy_ave_Wave_4)^2+(Vz_ave_Wave_4)^2);

% plot Decline results for each angle
figure
plot(angles, [V_ave_tripod_0, V_ave_tripod_2, V_ave_tripod_3,
 V_ave_tripod_4], '-bx', 'LineWidth', 1)
ylim([0 0.14])
hold on
plot(angles, [V_ave_ripple_0, V_ave_ripple_2, V_ave_ripple_3,
 V_ave_ripple_4], '-rx', 'LineWidth', 1)
hold on
plot(angles, [V_ave_wave_0, V_ave_wave_2, V_ave_wave_3,
 V_ave_wave_4], '-gx', 'LineWidth', 1)
grid on
legend('Tripod','Ripple','Wave','interpreter','latex', 'Location', 'NorthWest')
xlabel('Angle of Decline $\theta$ $^{\circ}$','interpreter','latex')
ylabel('Average Velocity $V_{ave}$ [m/s]','interpreter','latex')
title('Gait velocity comparison for CaveX robot over various
 declination angles','interpreter','latex')

% Incline Results for each gait and each angle
% work out average velocity for Tripod initial angle (0 blocks)
Time_Tripod_0 = Tripod_Incline_0(:,1); % s
X_Tripod_0 = Tripod_Incline_0(:,2); % m
Y_Tripod_0 = Tripod_Incline_0(:,3); % m
Z_Tripod_0 = Tripod_Incline_0(:,4); % m
delta_t_tripod_0 = Time_Tripod_0(length(Time_Tripod_0))-
Time_Tripod_0(1); % s
delta_x_tripod_0 = X_Tripod_0(length(X_Tripod_0))-X_Tripod_0(1); % m
delta_y_tripod_0 = Y_Tripod_0(length(Y_Tripod_0))-Y_Tripod_0(1); % m
delta_z_tripod_0 = Z_Tripod_0(length(Z_Tripod_0))-Z_Tripod_0(1); % m
Vx_ave_Tripod_0 = delta_x_tripod_0/delta_t_tripod_0; % m/s [average x
 velocity]
Vy_ave_Tripod_0 = delta_y_tripod_0/delta_t_tripod_0; % m/s [average x
 velocity]
```

```
Vz_ave_Tripod_0 = delta_z_tripod_0/delta_t_tripod_0; % m/s [average x
 velocity]
V_ave_tripod_0 =
 sqrt((Vx_ave_Tripod_0)^2+(Vy_ave_Tripod_0)^2+(Vz_ave_Tripod_0)^2);

% work out average velocity for Tripod second angle (2 blocks)
Time_Tripod_2 = Tripod_Incline_2(:,1); % s
X_Tripod_2 = Tripod_Incline_2(:,2); % m
Y_Tripod_2 = Tripod_Incline_2(:,3); % m
Z_Tripod_2 = Tripod_Incline_2(:,4); % m
delta_t_tripod_2 = Time_Tripod_2(length(Time_Tripod_2))-
Time_Tripod_2(1); % s
delta_x_tripod_2 = X_Tripod_2(length(X_Tripod_2))-X_Tripod_2(1); % m
delta_y_tripod_2 = Y_Tripod_2(length(Y_Tripod_2))-Y_Tripod_2(1); % m
delta_z_tripod_2 = Z_Tripod_2(length(Z_Tripod_2))-Z_Tripod_2(1); % m
Vx_ave_Tripod_2 = delta_x_tripod_2/delta_t_tripod_2; % m/s [average x
 velocity]
Vy_ave_Tripod_2 = delta_y_tripod_2/delta_t_tripod_2; % m/s [average x
 velocity]
Vz_ave_Tripod_2 = delta_z_tripod_2/delta_t_tripod_2; % m/s [average x
 velocity]
V_ave_tripod_2 =
 sqrt((Vx_ave_Tripod_2)^2+(Vy_ave_Tripod_2)^2+(Vz_ave_Tripod_2)^2);

% work out average velocity for Tripod third angle (3 blocks)
Time_Tripod_3 = Tripod_Incline_3(:,1); % s
X_Tripod_3 = Tripod_Incline_3(:,2); % m
Y_Tripod_3 = Tripod_Incline_3(:,3); % m
Z_Tripod_3 = Tripod_Incline_3(:,4); % m
delta_t_tripod_3 = Time_Tripod_3(length(Time_Tripod_3))-
Time_Tripod_3(1); % s
delta_x_tripod_3 = X_Tripod_3(length(X_Tripod_3))-X_Tripod_3(1); % m
delta_y_tripod_3 = Y_Tripod_3(length(Y_Tripod_3))-Y_Tripod_3(1); % m
delta_z_tripod_3 = Z_Tripod_3(length(Z_Tripod_3))-Z_Tripod_3(1); % m
Vx_ave_Tripod_3 = delta_x_tripod_3/delta_t_tripod_3; % m/s [average x
 velocity]
Vy_ave_Tripod_3 = delta_y_tripod_3/delta_t_tripod_3; % m/s [average x
 velocity]
Vz_ave_Tripod_3 = delta_z_tripod_3/delta_t_tripod_3; % m/s [average x
 velocity]
V_ave_tripod_3 =
 sqrt((Vx_ave_Tripod_3)^2+(Vy_ave_Tripod_3)^2+(Vz_ave_Tripod_3)^2);

% work out average velocity for Tripod initial angle (0 blocks)
Time_Tripod_4 = Tripod_Incline_4(:,1); % s
X_Tripod_4 = Tripod_Incline_4(:,2); % m
Y_Tripod_4 = Tripod_Incline_4(:,3); % m
Z_Tripod_4 = Tripod_Incline_4(:,4); % m
delta_t_tripod_4 = Time_Tripod_4(length(Time_Tripod_4))-
Time_Tripod_4(1); % s
delta_x_tripod_4 = X_Tripod_4(length(X_Tripod_4))-X_Tripod_4(1); % m
delta_y_tripod_4 = Y_Tripod_4(length(Y_Tripod_4))-Y_Tripod_4(1); % m
delta_z_tripod_4 = Z_Tripod_4(length(Z_Tripod_4))-Z_Tripod_4(1); % m
```

```
Vx_ave_Tripod_4 = delta_x_tripod_4/delta_t_tripod_4; % m/s [average x
 velocity]
Vy_ave_Tripod_4 = delta_y_tripod_4/delta_t_tripod_4; % m/s [average x
 velocity]
Vz_ave_Tripod_4 = delta_z_tripod_4/delta_t_tripod_4; % m/s [average x
 velocity]
V_ave_tripod_4 =
 sqrt((Vx_ave_Tripod_4)^2+(Vy_ave_Tripod_4)^2+(Vz_ave_Tripod_4)^2);

% work out average velocity for Ripple initial angle (0 blocks)
Time_Ripple_0 = Ripple_Incline_0(:,1); % s
X_Ripple_0 = Ripple_Incline_0(:,2); % m
Y_Ripple_0 = Ripple_Incline_0(:,3); % m
Z_Ripple_0 = Ripple_Incline_0(:,4); % m
delta_t_ripple_0 = Time_Ripple_0(length(Time_Ripple_0))-
Time_Ripple_0(1); % s
delta_x_ripple_0 = X_Ripple_0(length(X_Ripple_0))-X_Ripple_0(1); % m
delta_y_ripple_0 = Y_Ripple_0(length(Y_Ripple_0))-Y_Ripple_0(1); % m
delta_z_ripple_0 = Z_Ripple_0(length(Z_Ripple_0))-Z_Ripple_0(1); % m
Vx_ave_Ripple_0 = delta_x_ripple_0/delta_t_ripple_0; % m/s [average x
 velocity]
Vy_ave_Ripple_0 = delta_y_ripple_0/delta_t_ripple_0; % m/s [average x
 velocity]
Vz_ave_Ripple_0 = delta_z_ripple_0/delta_t_ripple_0; % m/s [average x
 velocity]
V_ave_ripple_0 =
 sqrt((Vx_ave_Ripple_0)^2+(Vy_ave_Ripple_0)^2+(Vz_ave_Ripple_0)^2);

% work out average velocity for Ripple second angle (2 blocks)
Time_Ripple_2 = Ripple_Incline_2(:,1); % s
X_Ripple_2 = Ripple_Incline_2(:,2); % m
Y_Ripple_2 = Ripple_Incline_2(:,3); % m
Z_Ripple_2 = Ripple_Incline_2(:,4); % m
delta_t_ripple_2 = Time_Ripple_2(length(Time_Ripple_2))-
Time_Ripple_2(1); % s
delta_x_ripple_2 = X_Ripple_2(length(X_Ripple_2))-X_Ripple_2(1); % m
delta_y_ripple_2 = Y_Ripple_2(length(Y_Ripple_2))-Y_Ripple_2(1); % m
delta_z_ripple_2 = Z_Ripple_2(length(Z_Ripple_2))-Z_Ripple_2(1); % m
Vx_ave_Ripple_2 = delta_x_ripple_2/delta_t_ripple_2; % m/s [average x
 velocity]
Vy_ave_Ripple_2 = delta_y_ripple_2/delta_t_ripple_2; % m/s [average x
 velocity]
Vz_ave_Ripple_2 = delta_z_ripple_2/delta_t_ripple_2; % m/s [average x
 velocity]
V_ave_ripple_2 =
 sqrt((Vx_ave_Ripple_2)^2+(Vy_ave_Ripple_2)^2+(Vz_ave_Ripple_2)^2);

% work out average velocity for Ripple third angle (3 blocks)
Time_Ripple_3 = Ripple_Incline_3(:,1); % s
X_Ripple_3 = Ripple_Incline_3(:,2); % m
Y_Ripple_3 = Ripple_Incline_3(:,3); % m
Z_Ripple_3 = Ripple_Incline_3(:,4); % m
delta_t_ripple_3 = Time_Ripple_3(length(Time_Ripple_3))-
Time_Ripple_3(1); % s
```

```
delta_x_ripple_3 = X_Ripple_3(length(X_Ripple_3))-X_Ripple_3(1); % m
delta_y_ripple_3 = Y_Ripple_3(length(Y_Ripple_3))-Y_Ripple_3(1); % m
delta_z_ripple_3 = Z_Ripple_3(length(Z_Ripple_3))-Z_Ripple_3(1); % m
Vx_ave_Ripple_3 = delta_x_ripple_3/delta_t_ripple_3; % m/s [average x
 velocity]
Vy_ave_Ripple_3 = delta_y_ripple_3/delta_t_ripple_3; % m/s [average x
 velocity]
Vz_ave_Ripple_3 = delta_z_ripple_3/delta_t_ripple_3; % m/s [average x
 velocity]
V_ave_ripple_3 =
 sqrt((Vx_ave_Ripple_3)^2+(Vy_ave_Ripple_3)^2+(Vz_ave_Ripple_3)^2);

% work out average velocity for Ripple fourth angle (4 blocks)
Time_Ripple_4 = Ripple_Incline_4(:,1); % s
X_Ripple_4 = Ripple_Incline_4(:,2); % m
Y_Ripple_4 = Ripple_Incline_4(:,3); % m
Z_Ripple_4 = Ripple_Incline_4(:,4); % m
delta_t_ripple_4 = Time_Ripple_4(length(Time_Ripple_4))-
Time_Ripple_4(1); % s
delta_x_ripple_4 = X_Ripple_4(length(X_Ripple_4))-X_Ripple_4(1); % m
delta_y_ripple_4 = Y_Ripple_4(length(Y_Ripple_4))-Y_Ripple_4(1); % m
delta_z_ripple_4 = Z_Ripple_4(length(Z_Ripple_4))-Z_Ripple_4(1); % m
Vx_ave_Ripple_4 = delta_x_ripple_4/delta_t_ripple_4; % m/s [average x
 velocity]
Vy_ave_Ripple_4 = delta_y_ripple_4/delta_t_ripple_4; % m/s [average x
 velocity]
Vz_ave_Ripple_4 = delta_z_ripple_4/delta_t_ripple_4; % m/s [average x
 velocity]
V_ave_ripple_4 =
 sqrt((Vx_ave_Ripple_4)^2+(Vy_ave_Ripple_4)^2+(Vz_ave_Ripple_4)^2);

% work out average velocity for Wave gait initial angle (0 blocks)
Time_Wave_0 = Wave_Incline_0(:,1); % s
X_Wave_0 = Wave_Incline_0(:,2); % m
Y_Wave_0 = Wave_Incline_0(:,3); % m
Z_Wave_0 = Wave_Incline_0(:,4); % m
delta_t_wave_0 = Time_Wave_0(length(Time_Wave_0))-Time_Wave_0(1); % s
delta_x_wave_0 = X_Wave_0(length(X_Wave_0))-X_Wave_0(1); % m
delta_y_wave_0 = Y_Wave_0(length(Y_Wave_0))-Y_Wave_0(1); % m
delta_z_wave_0 = Z_Wave_0(length(Z_Wave_0))-Z_Wave_0(1); % m
Vx_ave_Wave_0 = delta_x_wave_0/delta_t_wave_0; % m/s [average x
 velocity]
Vy_ave_Wave_0 = delta_y_wave_0/delta_t_wave_0; % m/s [average x
 velocity]
Vz_ave_Wave_0 = delta_z_wave_0/delta_t_wave_0; % m/s [average x
 velocity]
V_ave_wave_0 =
 sqrt((Vx_ave_Wave_0)^2+(Vy_ave_Wave_0)^2+(Vz_ave_Wave_0)^2);

% work out average velocity for Wave gait second angle (2 blocks)
Time_Wave_2 = Wave_Incline_2(:,1); % s
X_Wave_2 = Wave_Incline_2(:,2); % m
Y_Wave_2 = Wave_Incline_2(:,3); % m
Z_Wave_2 = Wave_Incline_2(:,4); % m
```

```
delta_t_wave_2 = Time_Wave_2(length(Time_Wave_2))-Time_Wave_2(1); % s
delta_x_wave_2 = X_Wave_2(length(X_Wave_2))-X_Wave_2(1); % m
delta_y_wave_2 = Y_Wave_2(length(Y_Wave_2))-Y_Wave_2(1); % m
delta_z_wave_2 = Z_Wave_2(length(Z_Wave_2))-Z_Wave_2(1); % m
Vx_ave_Wave_2 = delta_x_wave_2/delta_t_wave_2; % m/s [average x
 velocity]
Vy_ave_Wave_2 = delta_y_wave_2/delta_t_wave_2; % m/s [average x
 velocity]
Vz_ave_Wave_2 = delta_z_wave_2/delta_t_wave_2; % m/s [average x
 velocity]
V_ave_wave_2 =
 sqrt((Vx_ave_Wave_2)^2+(Vy_ave_Wave_2)^2+(Vz_ave_Wave_2)^2);

% work out average velocity for Wave gait third angle (3 blocks)
Time_Wave_3 = Wave_Incline_3(:,1); % s
X_Wave_3 = Wave_Incline_3(:,2); % m
Y_Wave_3 = Wave_Incline_3(:,3); % m
Z_Wave_3 = Wave_Incline_3(:,4); % m
delta_t_wave_3 = Time_Wave_3(length(Time_Wave_3))-Time_Wave_3(1); % s
delta_x_wave_3 = X_Wave_3(length(X_Wave_3))-X_Wave_3(1); % m
delta_y_wave_3 = Y_Wave_3(length(Y_Wave_3))-Y_Wave_3(1); % m
delta_z_wave_3 = Z_Wave_3(length(Z_Wave_3))-Z_Wave_3(1); % m
Vx_ave_Wave_3 = delta_x_wave_3/delta_t_wave_3; % m/s [average x
 velocity]
Vy_ave_Wave_3 = delta_y_wave_3/delta_t_wave_3; % m/s [average x
 velocity]
Vz_ave_Wave_3 = delta_z_wave_3/delta_t_wave_3; % m/s [average x
 velocity]
V_ave_wave_3 =
 sqrt((Vx_ave_Wave_3)^2+(Vy_ave_Wave_3)^2+(Vz_ave_Wave_3)^2);

% work out average velocity for Wave gait fourth angle (4 blocks)
Time_Wave_4 = Wave_Incline_4(:,1); % s
X_Wave_4 = Wave_Incline_4(:,2); % m
Y_Wave_4 = Wave_Incline_4(:,3); % m
Z_Wave_4 = Wave_Incline_4(:,4); % m
delta_t_wave_4 = Time_Wave_4(length(Time_Wave_4))-Time_Wave_4(1); % s
delta_x_wave_4 = X_Wave_4(length(X_Wave_4))-X_Wave_4(1); % m
delta_y_wave_4 = Y_Wave_4(length(Y_Wave_4))-Y_Wave_4(1); % m
delta_z_wave_4 = Z_Wave_4(length(Z_Wave_4))-Z_Wave_4(1); % m
Vx_ave_Wave_4 = delta_x_wave_4/delta_t_wave_4; % m/s [average x
 velocity]
Vy_ave_Wave_4 = delta_y_wave_4/delta_t_wave_4; % m/s [average x
 velocity]
Vz_ave_Wave_4 = delta_z_wave_4/delta_t_wave_4; % m/s [average x
 velocity]
V_ave_wave_4 =
 sqrt((Vx_ave_Wave_4)^2+(Vy_ave_Wave_4)^2+(Vz_ave_Wave_4)^2);

% plot Incline results for each angle and gait
figure
plot(angles, [V_ave_tripod_0, V_ave_tripod_2, V_ave_tripod_3,
 V_ave_tripod_4], '-bx', 'LineWidth', 1)
hold on
```

```
plot(angles, [V_ave_ripple_0, V_ave_ripple_2, V_ave_ripple_3,
 V_ave_ripple_4], '-rx', 'LineWidth', 1)
hold on
plot(angles, [V_ave_wave_0, V_ave_wave_2, V_ave_wave_3,
 V_ave_wave_4], '-gx', 'LineWidth', 1)
grid on
legend('Tripod','Ripple','Wave','interpreter','latex', 'Location', 'NorthEast')
xlabel('Angle of Inclination $\theta$
 $^{\circ}$','interpreter','latex')
ylabel('Average Velocity $V_{ave}$ [m/s]','interpreter','latex')
title('Gait velocity comparison for CaveX robot over various
 inclination angles','interpreter','latex')
```

Published with MATLAB® R2018b