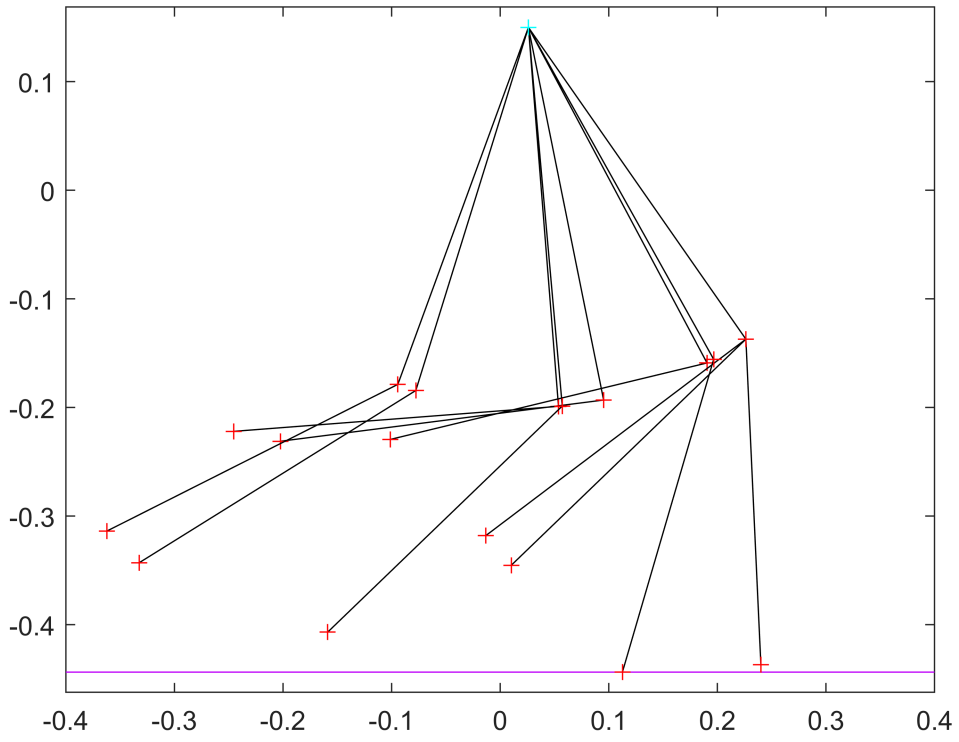
```
%%File: CV-CODE-Project-R2PT-Biped-2linkLEG-LSQ-EXP-4.mlx  :   Robot Arm (2-link) To PoinT (in
% Trajectory Generation of the Equivalent End Effector _EE for a 2-link robot arm
% Source location: QRIS> C:\Users\USER-PC\QRIS\MATLAB Code
% ------------------------------------------------------------------------
% Notes:
% 1. Use the EE trajectory from :  Project-R2PT-Biped-2linkLEG-ANALYSIS-I.mlx as truth data fo
% 2. Compute the angle vector for a 2-link leg (Biped) that realises the trajectory in (1)
% 3. Other reference file: Project-R2PT-LSQsampleCODE-2link_BL.mlx
% 4. Consider the tunable variables in the optimisation and the effects on the solution angle v
% 5. SET : TRR algorithm, lb = [-2*pi;-2*pi] , ub = [2*pi;2*pi], ex0 = [0;0] ===> LSQ configura
% 6. See pp.60-62 in QRIS-PROJECTS I
% 7. Consider: r1 = k*r2, adjust COM = [xCOM=0.026, yCOM=0.15] --> [xCOM=0.0, yCOM=0.0] , minir
% ------------------------------------------------------------------------------------------
% ------------------------------------------------------------------------------------------
clc;clf ; clear all;
clear global
% ------------------------------------------------------------------------
% CODE execution START:
% ------------------------------------------------------------------------
global nConfigs numConfigs  % declare all the global variables
nConfigs = 11;
numConfigs = 99;
n = 2 ; % # links
nj = n+1; % # joints
% ------------------------------ THETA ------------------------------------------------------
theta1 = [1.65;1.77;2.06;2.18;2.18;2.18;2.08;1.66;1.27;1.22]; % [rad] <--- data obtained by vis
theta2 = [4.71;4.64;4.46;4.75;4.87;5.72;5.49;5.39;5.57;5.53]; % [rad] <--- data obtained by vis
% ------------------------------------------------------------------------------------------
r1 = 0.35 ; r2 = 0.3 ; % link lengths [m] <--- original settings
%r1 = 0.3 ; r2 = 0.35 ; % link lengths [m]
% ------------------------------------------------------
x0 = -0.026 ; y0 = -0.15; z0 = 0.0055; % <---- assumed CoM 3D coordinates from LSQ run
CoM = [x0;y0;z0];
% ------------------------------------------------------
x0 = x0.*ones(10,1);
y0 = y0.*ones(10,1);
z0 = z0.*ones(10,1);
x1 = x0 + r1.*cos(theta1);
y1 = y0 + r1.*sin(theta1);
z1 = z0 ; % 2D planar configuration <-- CoM fixed
x2 = x0 + r1.*cos(theta1) + r2.*cos(theta1 + theta2);
y2 = y0 + r1.*sin(theta1) + r2.*sin(theta1 + theta2);
z2 = z0 ; % 2D planar configuration <-- CoM fixed
Xpoints = [x0 x1 x2]; % matrix of X coordinates --> [CoM joint1 joint2] < --- column data
Ypoints = [y0 y1 y2]; % matrix of Y coordinates --> [CoM joint1 joint2] < --- column data
Zpoints = [z0 z1 z2]; % matrix of Y coordinates --> [CoM joint1 joint2] < --- column data
% 3D world coordinates of the system
j=1;
%XYZ0 = [Xpoints(i,:);Ypoints(i,:);Zpoints(i,:)] % data points for image# = i
XYZtotal = zeros(30,3);
for i=1:10
    XYZi = [Xpoints(i,:);Ypoints(i,:);Zpoints(i,:)]; % data points for image# = i
```

```
    XYZtotal(j:j+2,:) = XYZi;
    j=j+3;
end
XYZactual = XYZtotal;
% ------------------------------------
% color CODE vectors ---->
cW = [1 1 1]; % white
cA = [1 0 0]; % red
cB = [0 0 1]; % blue
cC = [0 1 1]; % cyan
cD = [1 0 1]; % magenta
cE = [1 1 0]; % yellow
cJ = [0 1 0]; % green
cK = [0 0 0]; % black
% --- hybrids ----------
cF = [0.75 0 0.99]; % purple
cG = [0 0.4 0.3]; % dark green
cH = [0.6 0.98 0]; % light green
cI = [0.99 0.5 0]; % orange
% ------------------------------------

% ------------------------------------------------------------------------------------------
%                                ANIMATION OF 2-LINK LEG BIPED ===> motion direction
% ------------------------------------------------------------------------------------------
cnfg = 11; % configuration number out of 30 : gait 2 assumed range -->[11,20]
w=1;
% ----------- set up ANIMATION ---------------------->[]
loops = 1;
movColr(1:loops) = struct('cdata', [],'colormap',[]);
i_movie=1;

XYZa = XYZactual;
%XYZm = Robot3D;
%figure % ACTUAL XY plane coordinates evolution Z=0
j=[];
j=1;
EE_GGcontact = -0.4436;
GNDLL = [[-0.4,-0.2,0,0,0,0,0,0.2,0.3,0.4];EE_GGcontact*ones(1,10)];
% ------------------------------------------------------------------------------------------
figure
% set background colour
fig = gcf;
fig.Color = [1 1 1]; % white = [1 1 1]
colordef white
% ----------------------------
plot(GNDLL(1,:),GNDLL(2,:), 'Color',cF,'LineStyle','-'); % GROUND level >--GG--<
hold on
EE_data = 9*ones(2,10);
R3D = XYZtotal;
for k=1:10 % loop for the 10 configurations
%figure % plot JOINTS & LINKS ------ Cartesian coordinates ------------
line(-1.*XYZa(j,:),-1.*XYZa(j+1,:), 'Color','k','LineStyle','-')
%hold on
```

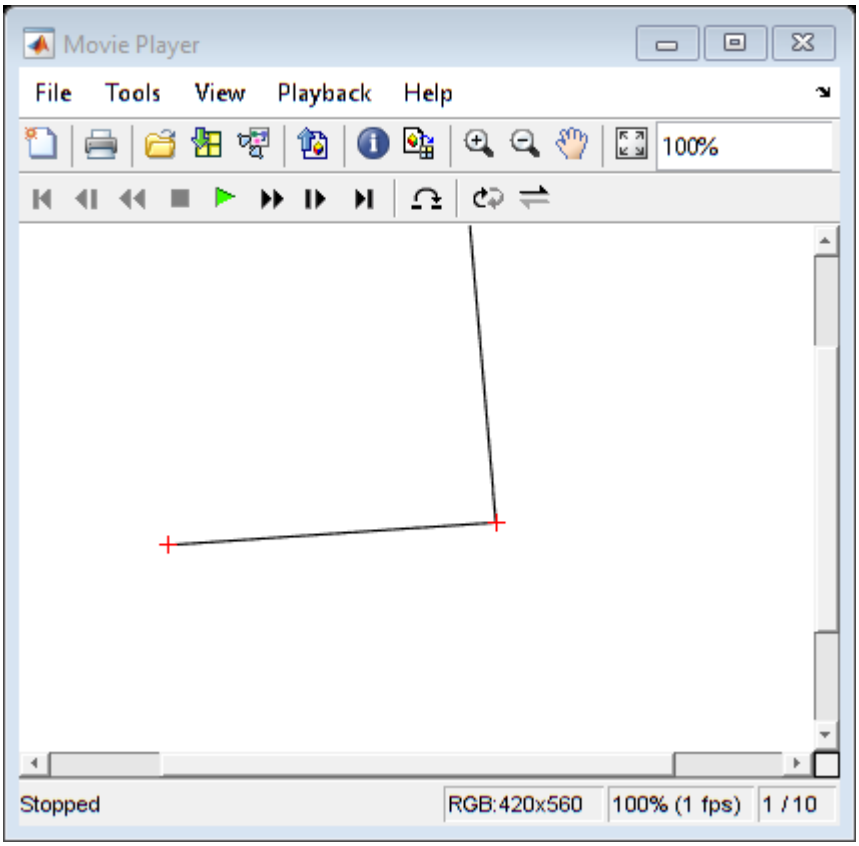
```
plot(-1.*XYZa(j,1),-1.*XYZa(j+1,1),'c', 'Marker','+') % reference marker = CoM
plot(-1.*XYZa(j,2),-1.*XYZa(j+1,2),'r','Marker','+')
plot(-1.*XYZa(j,3),-1.*XYZa(j+1,3),'r','Marker','+')
EE_data(1,w) = -1.*XYZa(j,3); % X data of EE
EE_data(2,w) = -1.*XYZa(j+1,3); % Y data of EE

j=j+3; % next set of 3D coordinates start on every 3rd row
movColr(i_movie) = getframe(gcf); % <--- store the current frame
i_movie = i_movie + 1;
w = w + 1;
end
EE_GGcontact = min(EE_data(2,:));
EE_GGcontact
```

```
EE_GGcontact = -0.4436
```

```
%GNDLL = [[-0.3,-0.2,0,0,0,0,0,0.2,0.3,0.4];EE_GGcontact*ones(1,10)];
%plot(GNDLL(1,:),GNDLL(2,:), 'Color',cF,'LineStyle','-'); % GROUND level >--GG--<

hold off
axis equal  % <---------------- SET axes equal for plot
```

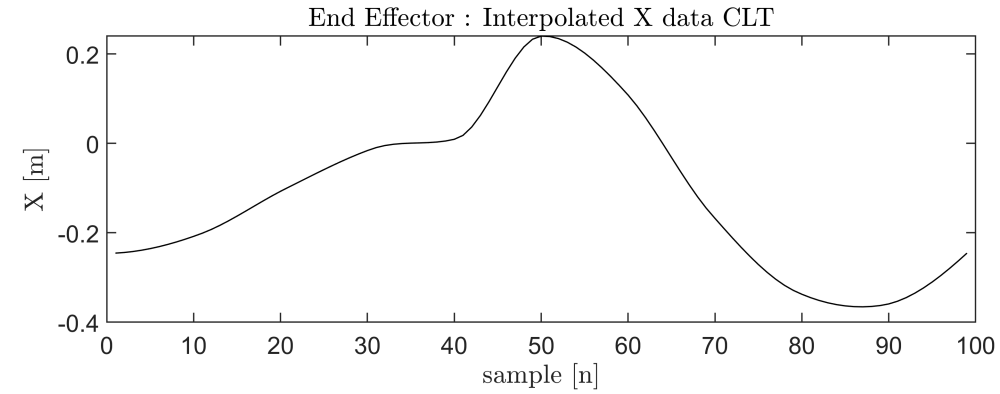


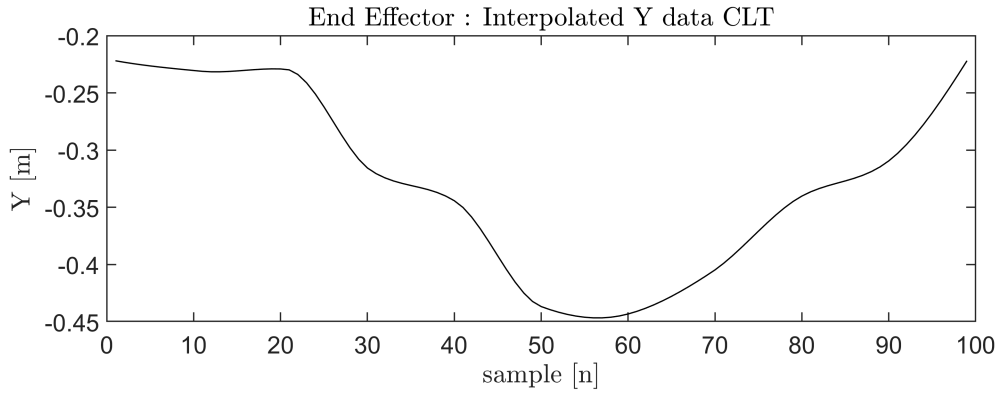

```
fps = 1; % <--- shows a frame every 1 [s]
implay(movColr,fps); % specify the frames per second to show in the animation
```



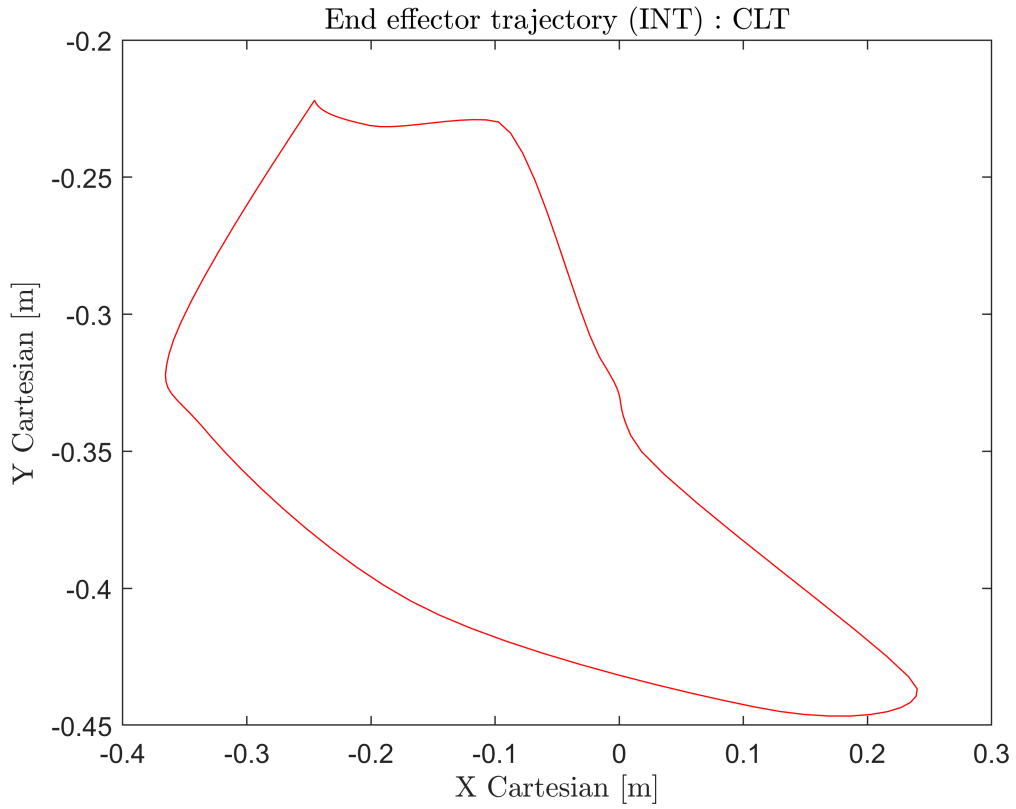

```
clear movColr
% ------------------------------------
XYZa = XYZactual;
j=[];
j=1;
w=1;
for k=1:10 % loop for the 10 configurations
EE_data(1,w) = -1.*XYZa(j,3); % X data of EE
EE_data(2,w) = -1.*XYZa(j+1,3); % Y data of EE

j=j+3; % next set of 3D coordinates start on every 3rd row
w = w + 1;
end
% ------------------------------------------------------------------------------------------------
figure
% set background colour
fig = gcf;
fig.Color = [1 1 1]; % white = [1 1 1]
colordef white
% ----------------------------
% ----------------------------
% ------------------------------------------------------------------------------------------------
X_EEdata = EE_data(1,:);
Y_EEdata = EE_data(2,:);

Xdata_CLT = [[X_EEdata],[EE_data(1,1)]];
Ydata_CLT = [[Y_EEdata],[EE_data(2,1)]];
```

```
% CLT = Closed Loop Trajectory
% ------------------------------------------------------------------------------------------------
% ------------------------------------------------------------------------------------------------
% ------------------------------------------------------------------------------------------------
%                                  INTERPOLATE DATA
% ------------------------------------------------------------------------------------------------
% ------------------------------------------------------------------------------------------------
% Interpolate data to 300 ==> smooth trajectories & high sample rates (i.e accuracy) for servo
% Interpolate angle data : 11 ---> 99 data points
EEx = Xdata_CLT'; % [11x1]
EEy = Ydata_CLT'; % [11x1]
EExy_CLT = [EEx,EEy]; % [11x2]
fr=30;T=1/fr;
%fr=11;T=1/fr;
xx1 = linspace(0,(nConfigs-1)*T,nConfigs);%  time line for all configurations
xx0 = linspace(xx1(1),xx1(end),9*nConfigs);% interpolated data x axis time line
EExyINT_CLT = zeros(99,2);
Y1=[];YY1=[];
for k=1:2
    Y1=[];YY1=[];
    Y1 = EExy_CLT(:,k)';
    % INTERPOLATION ------------->
    %YY1 = interp1(xx1,Y1,xx0,'spline','extrap'); % <--- 2 discontinuities
    %YY1 = interp1(xx1,Y1,xx0,'pchip','extrap');
    YY1 = interp1(xx1,Y1,xx0,'makima','extrap');

    EExyINT_CLT(:,k) = YY1;
    clearvars Y1 YY1
end

figure
subplot(2,1,1),plot(EExyINT_CLT(:,1), 'Color',cK)
title(' End Effector : Interpolated X data  CLT', 'Interpreter','latex')
xlabel('sample  [n]', 'Interpreter','latex')
ylabel('X [m]', 'Interpreter','latex')
subplot(2,1,2),plot( EExyINT_CLT(:,2), 'Color',cK)
title(' End Effector : Interpolated Y data  CLT', 'Interpreter','latex')
xlabel('sample  [n]', 'Interpreter','latex')
ylabel('Y [m]', 'Interpreter','latex')
```






```
% ----------------------------
figure
plot(EExyINT_CLT(:,1),EExyINT_CLT(:,2), 'Color',cA)
title('End effector trajectory (INT)  : CLT', 'Interpreter','latex')
xlabel('X Cartesian [m]', 'Interpreter','latex')
ylabel('Y Cartesian [m]', 'Interpreter','latex')
```

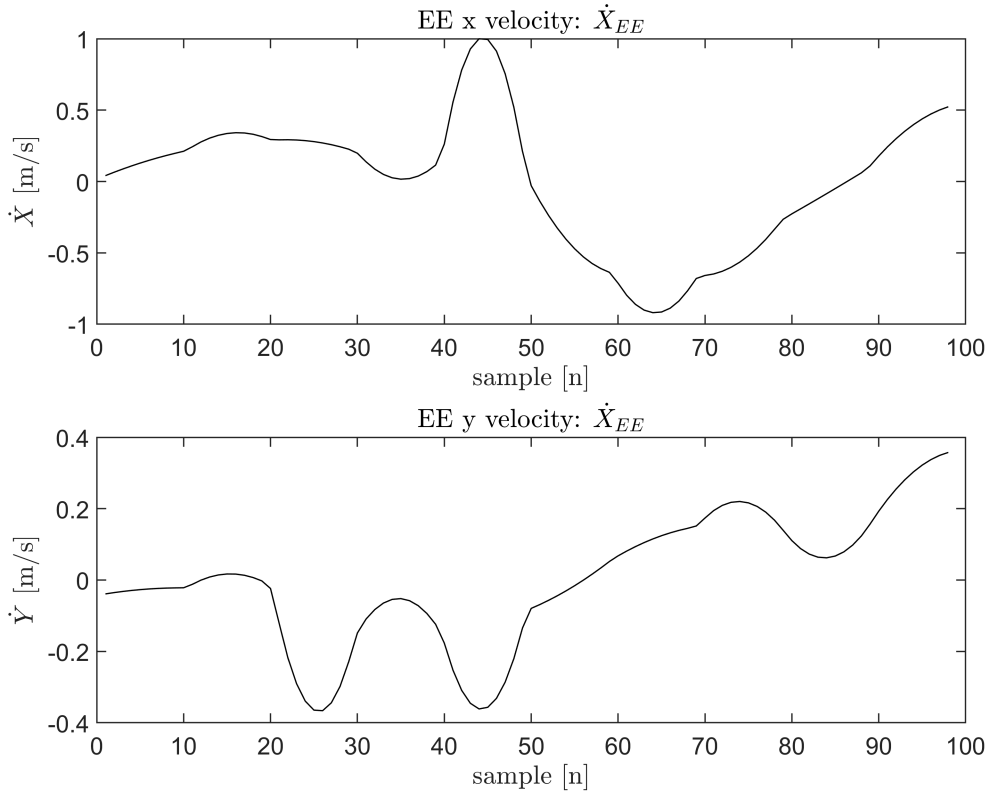



```
% ------------------------------------------------------------------------------------------------
% ------------------------------------------------------------------------------------------------
% COMPUTE END EFFECTOR Cartesian Velocities
% ------------------------------------------------------------------------------------------------
xEEplane = EExyINT_CLT(:,1);
yEEplane = EExyINT_CLT(:,2);
Ts = T;
Ts
```

```
Ts = 0.0333
```

```
EExd = zeros((numConfigs-1),1); % stores x velocity points of EE
EEyd = zeros((numConfigs-1),1); % stores y velocity points of EE
for i=1:(numConfigs-1)
 EExd(i) = ((xEEplane(i+1)-xEEplane(i))/Ts); % [m/s]
 EEyd(i) = ((yEEplane(i+1)-yEEplane(i))/Ts); % [m/s]
end
% ------------------------------------------------------------------------------------------------
figure
subplot(2,1,1),plot(EExd,'Color', cK);
title('EE x velocity: ${{\dot{X}}}_{{EE}}$', 'Interpreter','latex')
xlabel('sample [n]', 'Interpreter','latex')
ylabel(' ${{\dot{X}}}$ [m/s]', 'Interpreter','latex')
subplot(2,1,2),plot(EEyd,'Color', cK);
%title('LSQ estimated EE trajectory: y', 'Interpreter','latex')
title('EE y velocity: ${{\dot{X}}}_{{EE}}$', 'Interpreter','latex')
```

```
xlabel('sample [n]', 'Interpreter','latex')
ylabel(' ${{\dot{Y}}}$ [m/s]', 'Interpreter','latex')
```



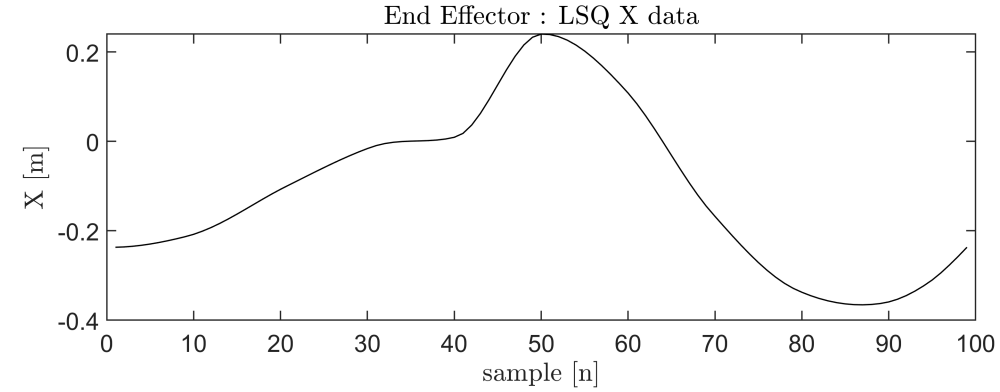

```
% ------------------------------------------------------------------------------------------
% ------------------------------------------------------------------------------------------
%                                  Non-linear LEAST SQUARES OPTIMISATION
% ------------------------------------------------------------------------------------------
% Compute the angle data for all 99 configurations using the EE coordinates as truth data :
% ------------------------------------------------------------------------------------------
xEEspace = EExyINT_CLT(:,1);  % [99x1]
yEEspace = EExyINT_CLT(:,2);  % [99x1]
%r1 = 0.459; r2 = 0.611; % link lengths of 2-link back leg (BL) <----------- original estimates
%r1 = 0.459; r2 = 0.511; % link lengths of 2-link back leg (BL) <----------- original estimates

%r1 = 0.35 ; r2 = 0.3 ; % link lengths [m]
%maxExt = 0.82; sFac = 0.45; % <--- GOOD results for th1_LSQ & th2_LSQ
%r1 = sFac*maxExt; r2 = 0.79 - r1; % link lengths of 2-link  % <--- GOOD results for th1_LSQ &

%r1 = 0.35 ; r2 = 0.3 ; % link lengths [m] <--- original settings
%r2 = 0.3 ; % r1 = k*r2 with r2=0.3 link lengths [m]
%r1 = 0.425; r2 = 0.375;
r1 = 0.425; r2 = 0.425; % link lengths [m] : r1 = r2 = 0.425
%r1 = 0.3 ; r2 = 0.35 ; % link lengths [m]
%r1 = 0.35 ; r2 = 0.35 ; % link lengths [m] : r1 = r2 = 0.35
maxDIS = 0.5085; % see: Project-R2PT-Biped-2linkLEG-LSQ.mlx
%r1 = 0.65/2; r2 = r1; % <---- link lengths equal
%r1 = maxDIS/2; r2 = r1; % <---- link lengths equal : <--- LIMIT on link lengths
% Check link length TOTAL --->
```

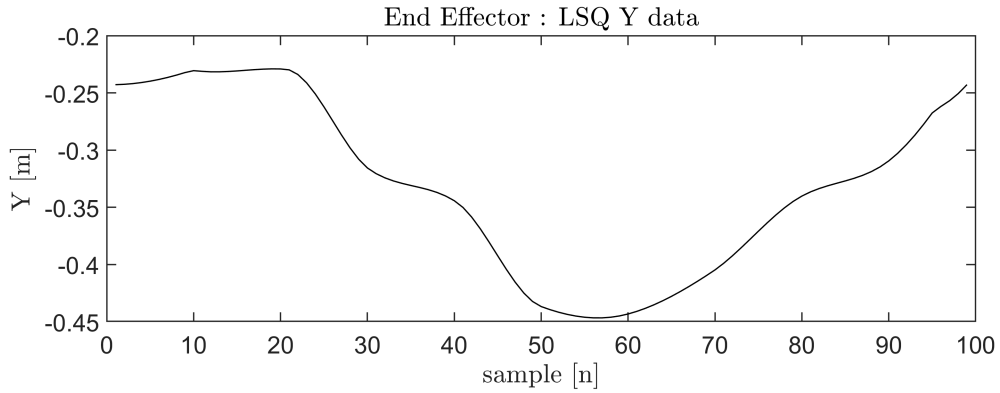
```
sum_r1r2 = r1 + r2;
sum_r1r2
```

```
sum_r1r2 = 0.8500
```

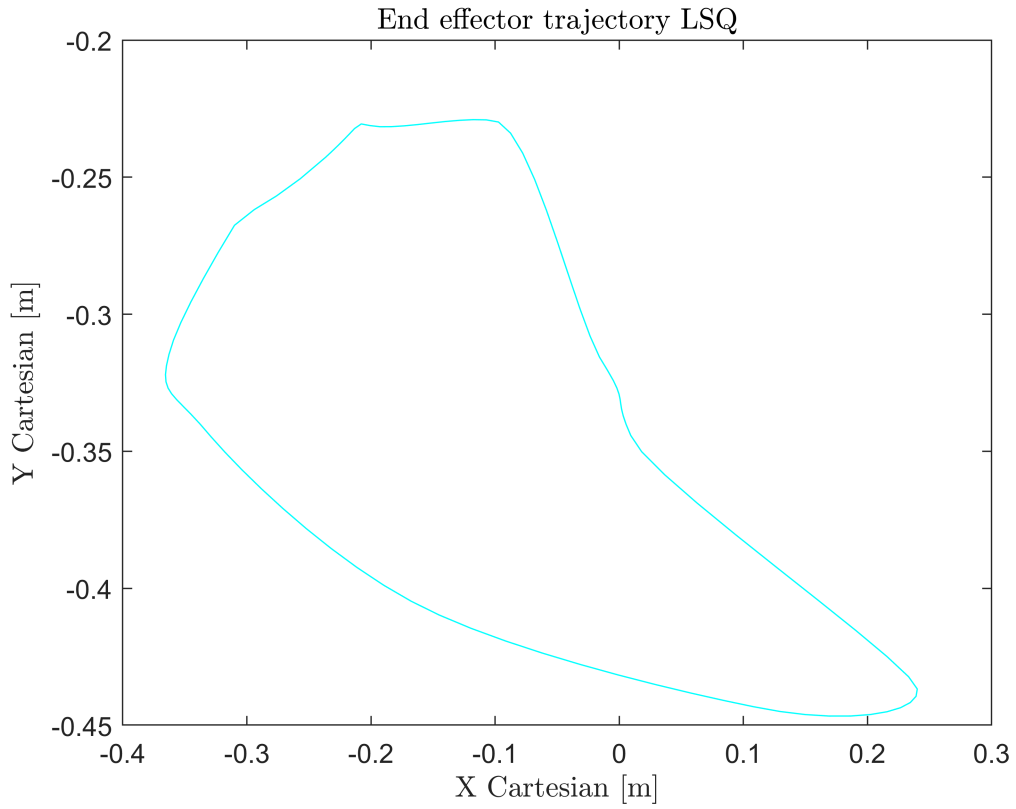
```
% ------------------------------------------------------
%x0 = 0.0 ; y0 = 0.0; % <---- assumed CoM 2D coordinates from LSQ run
%x0 = -0.026 ; y0 = -0.15;  % <---- assumed CoM 3D coordinates from LSQ run
x0 = 0.026 ; y0 = 0.15;  % <---- assumed CoM 3D coordinates from LSQ run

CoM = [x0;y0];
% ------------------------------------------------------------------------------------------
xo = x0; yo = y0; % fixed reference joint J#0 coordinates
%ex0 = [0.5;0.5;0.6;0.45]; % guesstimates of unknowns [rad] : [th1;th2] <===> [x(1);x(2)] = x
%ex0 = [0.5;0.5]; % guesstimates of unknowns [rad] : [th1;th2] <===> [x(1);x(2)] = x

%lb = [-pi;-pi;0.2;0.2]; ub = [pi;pi;0.7;0.65]; % set the bounds for x solution variables
%lb=[];ub=[]; % LVM algorithm
%lb = [-pi;-pi]; ub = [pi;pi]; % set the bounds for x solution variables TRR algorithm
X_EEinpt = [];
X_EEinptDATA = [xEEspace';yEEspace'];
LSQ_thetaDATA = 9*ones(2,99); % [2x99] <--- store all of the LSQ angle estimates
%LSQ_thetaDATA = 9*ones(3,99); % [3x99] <--- store all of the LSQ angle estimates

% EXECUTE non-linear Least Squares Optimisation Algorithm :
% ------------------------------------------------------->>>>>
for i = 1:99
% INITIAL VALUE SETTINGS :
% ------------------------
%ex0 = [0.0;0.0]; % guesstimates of unknowns : [th1;th2] <===> [x(1);x(2)] = x
%ex0 = [0.0;0.0;1.1]; % guesstimates of unknowns : [th1;th2;k] <===> [x(1);x(2);x(3)] = x
%ex0 = [0.5;0.5]; % guesstimates of unknowns : [th1;th2] <===> [x(1);x(2)] = x
ex0 = [-1.5;1.0]; % guesstimates of unknowns : [th1;th2] <===> [x(1);x(2)] = x    <------ good
%ex0 = [-2.9;1.5]; % guesstimates of unknowns : [th1;th2] <===> [x(1);x(2)] = x
%ex0 = [1.65;4.71]; % guesstimates of unknowns : [th1;th2] <===> [x(1);x(2)] = x  <--- initial
% NOTE: ex0 = [1.65;4.71] <--- POOR results : angle and EE trajectory singularities !!!
% ------------------------

% SET BOUNDS for search :
% ----------------------
% SET bounds for unknown solution elements of x : <--- ONLY applicable for : Trust-region-refle
lb = [-pi;-pi]; ub = [pi;pi]; % set the bounds for x solution variables <----- original setting
%lb = [-2*pi;-2*pi]; ub = [2*pi;2*pi];
%lb = [-3*pi;-3*pi]; ub = [3*pi;3*pi];
%lb = [-pi/3]; ub = [pi/23]; % set the bounds for x solution variables

%lb=[]; ub=[]; % <--- ONLY applicable for : Levenberg-Marquardt (LVM) Algorithm
% ----------------------

X_EEinpt = [X_EEinptDATA(1,i);X_EEinptDATA(2,i)];
statenlLSQ = sol_theta(xo,yo,r1,r2,ex0,X_EEinpt,lb,ub);  % <------------- call LSQ defined func
%statenlLSQ = sol_theta(xo,yo,r2,ex0,X_EEinpt,lb,ub);  % <------------- call LSQ defined functi
th1_LSQ = statenlLSQ(1);
```

```
th2_LSQ = statenlLSQ(2);
%k_LSQ = statenlLSQ(3);
LSQ_thetaDATA(1,i) = th1_LSQ;
LSQ_thetaDATA(2,i) = th2_LSQ;
%LSQ_thetaDATA(3,i) = k_LSQ;
end

% ------------------------------------------------------------------------------------------
% SAVE the DATA |TEMPLATE|  :::::::>>> [PSD1] size of DATA matrix:[90x9]
%save('C:\Users\Casey\Projects-MATLAB\DATA\PSD1.txt','PSD1','-ascii')
save('C:\Users\USER-PC\QRIS\MATLAB Code\DATA\LSQ_thetaDATA_biped.txt','LSQ_thetaDATA','-ascii')
% ------------------------------------------------------------------------------------------
% ------------------------------------------------------------------------------------------
% Compute the configuration in space o----o---->
xCOM = xo; yCOM = yo;
%r2 = (mean(r1_LSQ))*1.33;
%r1 = mean(r1_LSQ);
%r1 = mean(k_LSQ)*r2; %r1 = x(3)*r2
%r1 = 0.35; r2 = 0.3;  % <----- original setting
%r1 = 0.425; r2 = 0.375;
r1 = 0.425; r2 = 0.425; % link lengths [m] : r1 = r2 = 0.425
%r1 = 0.35 ; r2 = 0.35 ; % link lengths [m] : r1 = r2 = 0.35
p1x = xCOM.*ones(1,99) + r1.*cos(LSQ_thetaDATA(1,:));
p1y = yCOM.*ones(1,99) + r1.*sin(LSQ_thetaDATA(1,:));
p2x = p1x+r2.*cos(LSQ_thetaDATA(1,:)+LSQ_thetaDATA(2,:));
p2y = p1y+r2.*sin(LSQ_thetaDATA(1,:)+LSQ_thetaDATA(2,:));

Xx1=p1x;Xy1=p1y;
Xx2=p2x;Xy2=p2y;
Xx1SP=p1x;Xy1SP=p1y;
Xx2SP=p2x;Xy2SP=p2y;
% plot a configuration
Xdata=[Xx1; Xx2 ];
%Ydata=-1.*[Xy1, Xy2 ];
Ydata=[Xy1; Xy2 ];
% ----------------------------PLOT []----------------------------------->
%figure
%subplot(2,1,1),plot(LSQ_thetaDATA(3,:), 'Color',cK)
%title('LSQ estimates for k', 'Interpreter','latex')
%xlabel('sample  [n]', 'Interpreter','latex')
%ylabel('k', 'Interpreter','latex')
figure
subplot(2,1,1),plot(Xdata(2,:), 'Color',cK)
title('End Effector : LSQ X data', 'Interpreter','latex')
xlabel('sample  [n]', 'Interpreter','latex')
ylabel('X [m]', 'Interpreter','latex')

subplot(2,1,2),plot(Ydata(2,:), 'Color',cK)
title('End Effector : LSQ Y data', 'Interpreter','latex')
xlabel('sample  [n]', 'Interpreter','latex')
ylabel('Y [m]', 'Interpreter','latex')
```

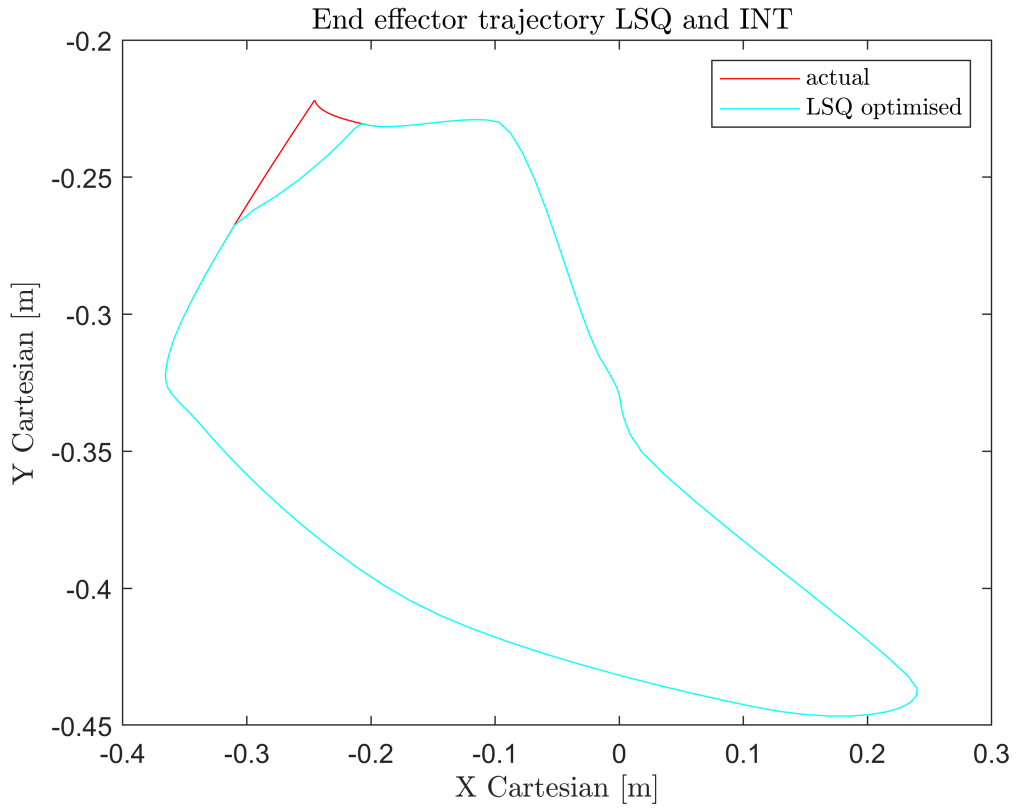





```
EE_GGcontact_auto = min(Ydata(2,:));
% ----------------------------
figure
plot(Xdata(2,:),Ydata(2,:), 'Color',cC)
title('End effector trajectory LSQ', 'Interpreter','latex')
xlabel('X Cartesian [m]', 'Interpreter','latex')
ylabel('Y Cartesian [m]', 'Interpreter','latex')
```




```
figure
p1=plot(EExyINT_CLT(:,1),EExyINT_CLT(:,2), 'Color',cA);
hold on
p2=plot(Xdata(2,:),Ydata(2,:), 'Color',cC);
hold off
title('End effector trajectory LSQ and INT ', 'Interpreter','latex')
legend([p1,p2],{'actual','LSQ optimised'},'Location','northeast','NumColumns',1, 'Interpreter'
xlabel('X Cartesian [m]', 'Interpreter','latex')
ylabel('Y Cartesian [m]', 'Interpreter','latex')
```




```
% ------------------------------------------------------------------------------------
%             ORIGINAL ANIMATION OF 2-LINK LEG BIPED ===> motion direction
% ------------------------------------------------------------------------------------
cnfg = 11; % configuration number out of 30 : gait 2 assumed range -->[11,20]
w=1;
% ----------- set up ANIMATION ---------------------->[]
loops = 1;
movColr(1:loops) = struct('cdata', [],'colormap',[]);
i_movie=1;

XYZa = XYZactual;
%XYZm = Robot3D;
%figure % ACTUAL XY plane coordinates evolution Z=0
j=[];
j=1;
%EE_GGcontact = -0.4436;
EE_GGcontact_auto
```

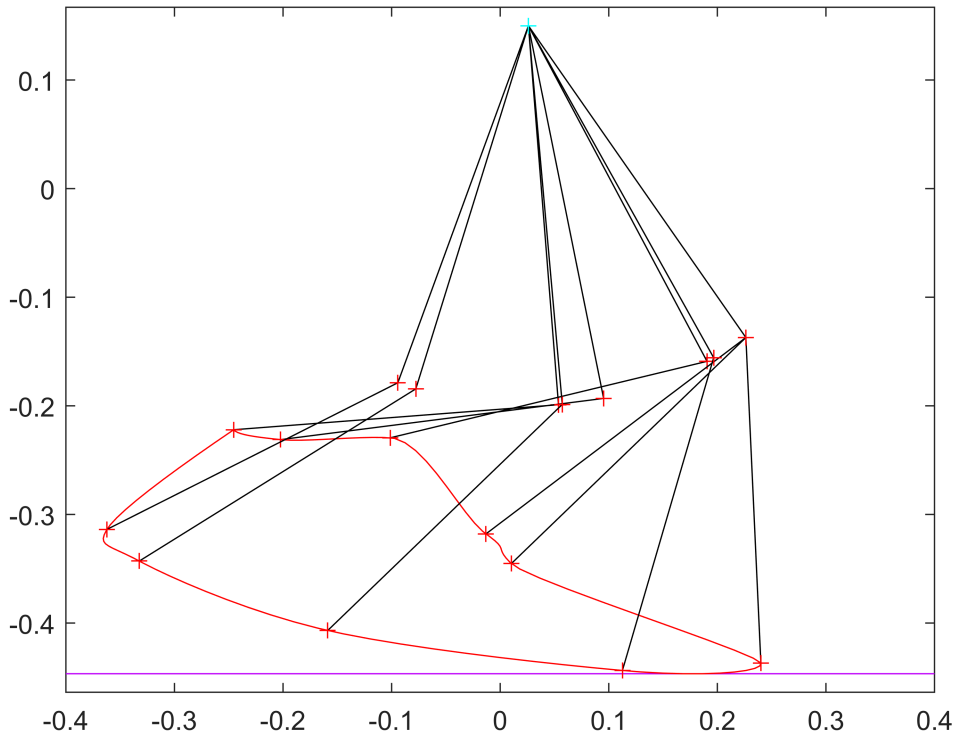
```
EE_GGcontact_auto = -0.4466
```

```
EE_GGcontact = EE_GGcontact_auto;
GNDLL = [[-0.4,-0.2,0,0,0,0,0,0,0.2,0.3,0.4];EE_GGcontact*ones(1,10)];
% ------------------------------------------------------------------------------------
figure
% set background colour
fig = gcf;
```

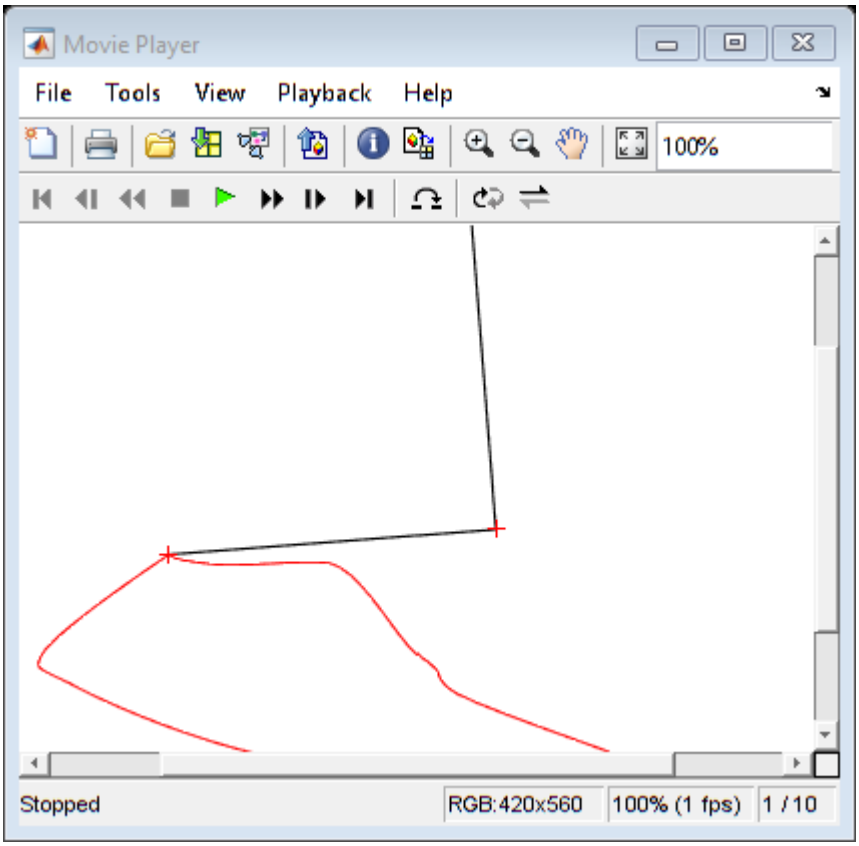
```
fig.Color = [1 1 1]; % white = [1 1 1]
colordef white
% ----------------------------
plot(GNDLL(1,:),GNDLL(2,:), 'Color',cF,'LineStyle','-'); % GROUND level >--GG--<
hold on
plot(EExyINT_CLT(:,1),EExyINT_CLT(:,2), 'Color',cA)
axis equal  % <---------------- SET axes equal for plot
EE_data = 9*ones(2,10);
R3D = XYZtotal;
for k=1:10 % loop for the 10 configurations
%figure % plot JOINTS & LINKS ------ Cartesian coordinates ------------
line(-1.*XYZa(j,:),-1.*XYZa(j+1,:), 'Color','k','LineStyle','-')
%hold on
plot(-1.*XYZa(j,1),-1.*XYZa(j+1,1),'c', 'Marker','+') % reference marker = CoM
plot(-1.*XYZa(j,2),-1.*XYZa(j+1,2),'r','Marker','+')
plot(-1.*XYZa(j,3),-1.*XYZa(j+1,3),'r','Marker','+')
EE_data(1,w) = -1.*XYZa(j,3); % X data of EE
EE_data(2,w) = -1.*XYZa(j+1,3); % Y data of EE

j=j+3; % next set of 3D coordinates start on every 3rd row
movColr(i_movie) = getframe(gcf); % <--- store the current frame
i_movie = i_movie + 1;
w = w + 1;
end
%EE_GGcontact = min(EE_data(2,:));
EE_GGcontact = EE_GGcontact_auto;
EE_GGcontact
```

```
EE_GGcontact = -0.4466
```

```
%GNDLL = [[-0.3,-0.2,0,0,0,0,0,0,0.2,0.3,0.4];EE_GGcontact*ones(1,10)];
%plot(GNDLL(1,:),GNDLL(2,:), 'Color',cF,'LineStyle','-'); % GROUND level >--GG--<

hold off
axis equal  % <---------------- SET axes equal for plot
```




```
fps = 1; % <--- shows a frame every 1 [s]
implay(movColr,fps); % specify the frames per second to show in the animation
```

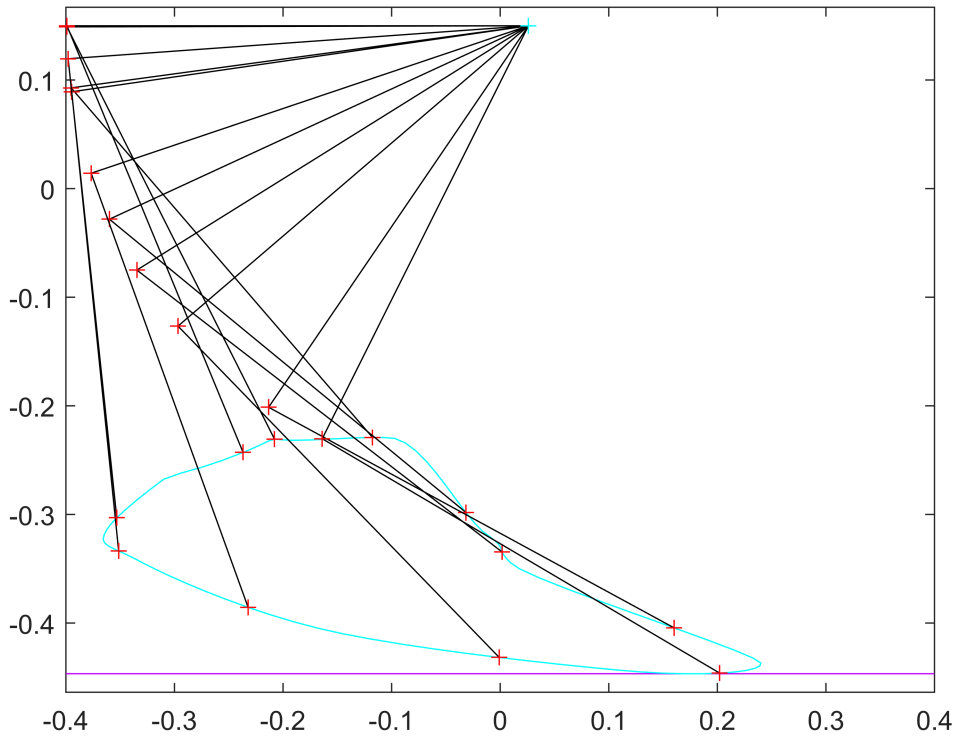



```
clear movColr
% ------------------------------------

% ------------------------------------------------------------------------------------------
%       NON-LINEAR LEAST SQUARES OPTIMISATION - ANIMATION OF 2-LINK LEG BIPED
% ------------------------------------------------------------------------------------------
XYZa = [[xCOM.*ones(1,99)]', [p1x'],[p2x'] ];
XYZb = [[yCOM.*ones(1,99)]', [p1y'],[p2y'] ];
%cnfg = 11; % configuration number out of 30 : gait 2 assumed range -->[11,20]
w=1;
% ----------- set up ANIMATION ---------------------->[]
loops = 1;
movColr(1:loops) = struct('cdata', [],'colormap',[]);
i_movie=1;


j=[];
j=1;
%EE_GGcontact = -0.4436;
EE_GGcontact = EE_GGcontact_auto;
GNDLL = [[-0.4,-0.2,0,0,0,0,0,0.2,0.3,0.4];EE_GGcontact*ones(1,10)];
% ------------------------------------------------------------------------------------------
figure
% set background colour
fig = gcf;
fig.Color = [1 1 1]; % white = [1 1 1]
colordef white
```

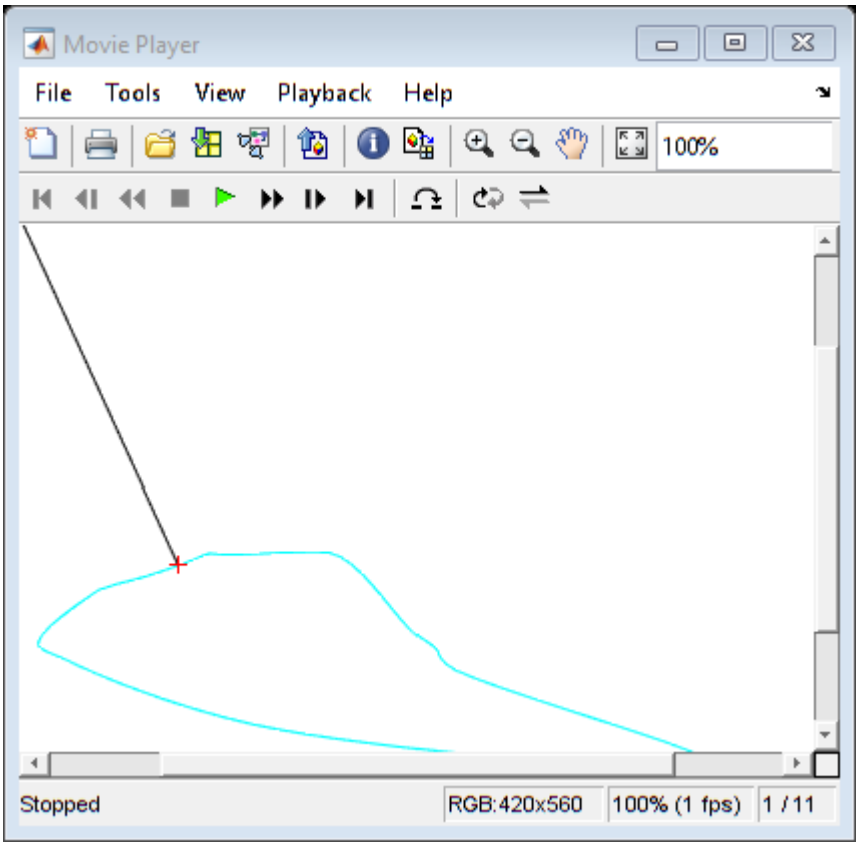
```
% ----------------------------
plot(GNDLL(1,:),GNDLL(2,:), 'Color',cF,'LineStyle','-'); % GROUND level >--GG--<
hold on
plot(Xdata(2,:),Ydata(2,:), 'Color',cC);
EE_data = 9*ones(2,10);
R3D = XYZtotal;
for k=1:11 % loop for the 10 configurations
%figure % plot JOINTS & LINKS ------ Cartesian coordinates ------------
line(XYZa(j,:),XYZb(j,:), 'Color','k','LineStyle','-')
%hold on
plot(XYZa(j,1),XYZb(j,1),'c', 'Marker','+') % reference marker = CoM
plot(XYZa(j,2),XYZb(j,2),'r','Marker','+')
plot(XYZa(j,3),XYZb(j,3),'r','Marker','+')
EE_data(1,w) = XYZa(j,3); % X data of EE
EE_data(2,w) = XYZb(j,3); % Y data of EE

j=j+9; % next set of 2D coordinates start on next row
movColr(i_movie) = getframe(gcf); % <--- store the current frame
i_movie = i_movie + 1;
w = w + 1;
end
%EE_GGcontact = min(EE_data(2,:));
EE_GGcontact = EE_GGcontact_auto;
EE_GGcontact
```

```
EE_GGcontact = -0.4466
```

```
%GNDLL = [[-0.3,-0.2,0,0,0,0,0,0,0.2,0.3,0.4];EE_GGcontact*ones(1,10)];
%plot(GNDLL(1,:),GNDLL(2,:), 'Color',cF,'LineStyle','-'); % GROUND level >--GG--<

hold off
axis equal  % <---------------- SET axes equal for plot
```

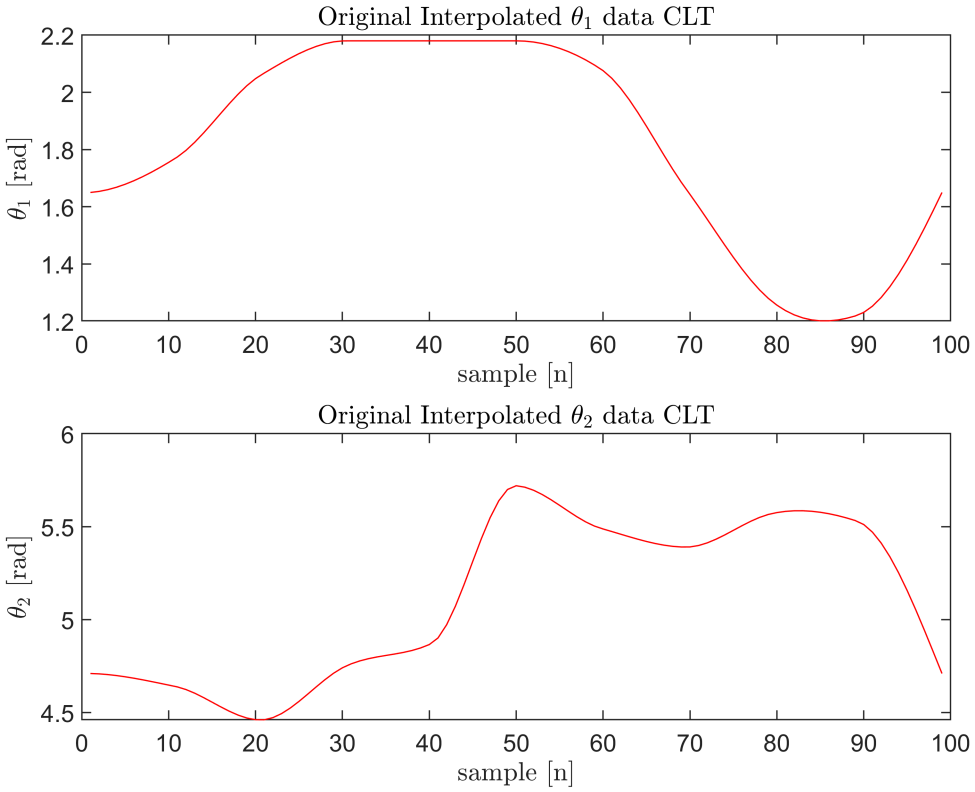



```
fps = 1; % <--- shows a frame every 1 [s]
implay(movColr,fps); % specify the frames per second to show in the animation
```



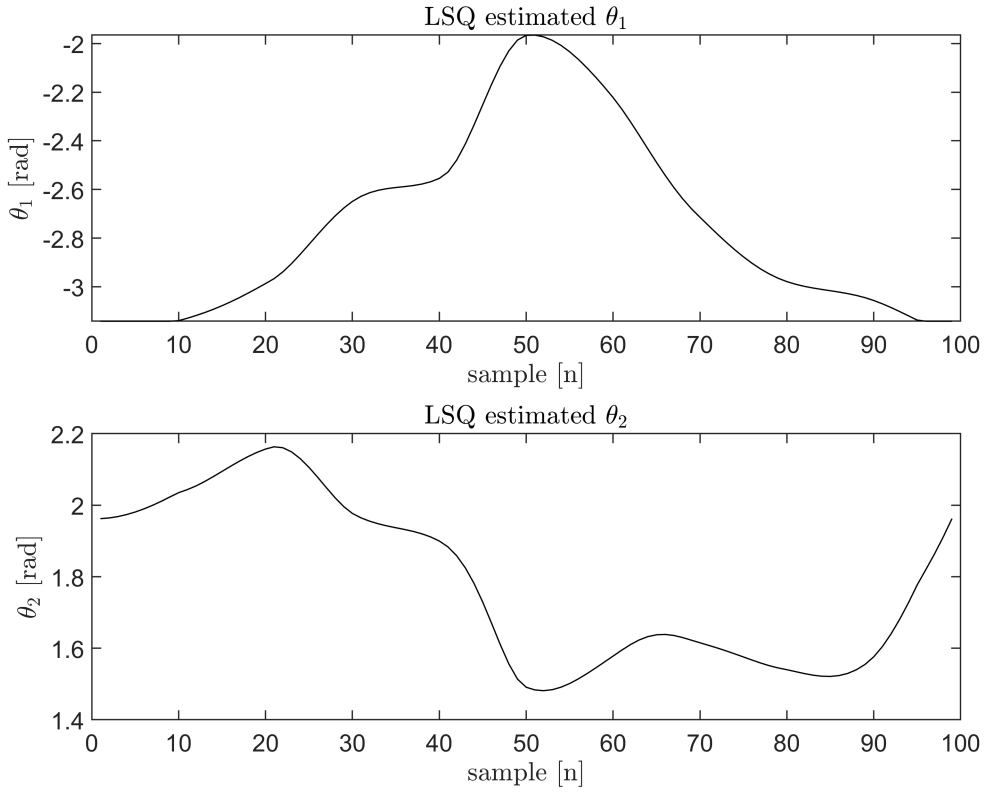

```
clear movColr
% ------------------------------------

% ---------------------------------------------------------------------------------------------
% ---------------------------------------------------------------------------------------------
%                                   INTERPOLATE ANGLE DATA
% ---------------------------------------------------------------------------------------------
% ---------------------------------------------------------------------------------------------
% ------------------------------ THETA ----------------------------------------------------
theta1 = [1.65;1.77;2.06;2.18;2.18;2.18;2.08;1.66;1.27;1.22]; % [rad] <--- data obtained by vis
theta2 = [4.71;4.64;4.46;4.75;4.87;5.72;5.49;5.39;5.57;5.53]; % [rad] <--- data obtained by vis
% ---------------------------------------------------------------------------------------------
theta1_CL = [1.65;1.77;2.06;2.18;2.18;2.18;2.08;1.66;1.27;1.22;1.65]; % [rad] <--- data obtaine
theta2_CL = [4.71;4.64;4.46;4.75;4.87;5.72;5.49;5.39;5.57;5.53;4.71]; % [rad] <--- data obtaine

% Interpolate data to 300 ==> smooth trajectories & high sample rates (i.e accuracy) for servo
% Interpolate angle data : 11 ---> 99 data points
TH1 = theta1_CL; % [11x1]
TH2 = theta2_CL; % [11x1]
TH12_CLT = [TH1,TH2]; % [11x2]
fr=30;T=1/fr;
%fr=11;T=1/fr;
xx1 = linspace(0,(nConfigs-1)*T,nConfigs);%  time line for all configurations
xx0 = linspace(xx1(1),xx1(end),9*nConfigs);% interpolated data x axis time line
thetaINT_CLT = zeros(99,2);
Y1=[];YY1=[];
for k=1:2
```

```
    Y1=[];YY1=[];
    Y1 = TH12_CLT(:,k)';
    % INTERPOLATION ------------->
    %YY1 = interp1(xx1,Y1,xx0,'spline','extrap'); % <--- 2 discontinuities
    %YY1 = interp1(xx1,Y1,xx0,'pchip','extrap');
    YY1 = interp1(xx1,Y1,xx0,'makima','extrap');

   thetaINT_CLT(:,k) = YY1;
   clearvars Y1 YY1
end

figure
subplot(2,1,1),plot(thetaINT_CLT(:,1), 'Color',cA)
title('Original Interpolated $\theta_{1}$ data  CLT', 'Interpreter','latex')
xlabel('sample  [n]', 'Interpreter','latex')
ylabel('$\theta_{1}$ [rad]', 'Interpreter','latex')
subplot(2,1,2),plot(thetaINT_CLT(:,2), 'Color',cA)
title('Original Interpolated $\theta_{2}$ data  CLT', 'Interpreter','latex')
xlabel('sample  [n]', 'Interpreter','latex')
ylabel('$\theta_{2}$ [rad]', 'Interpreter','latex')
```

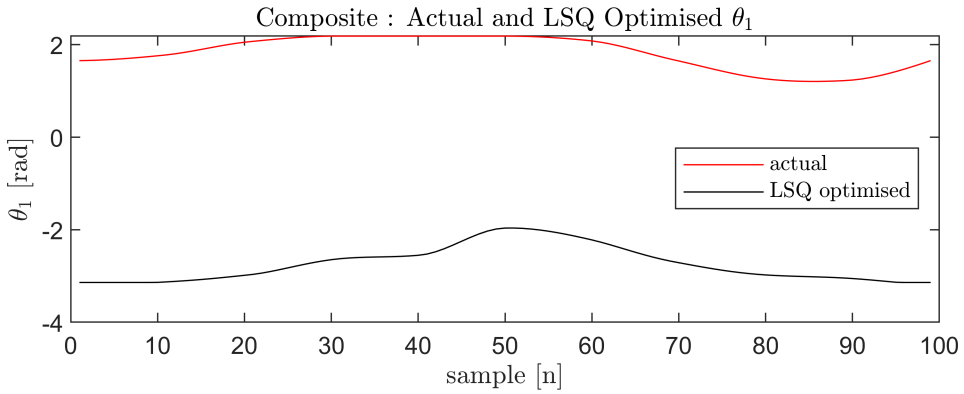


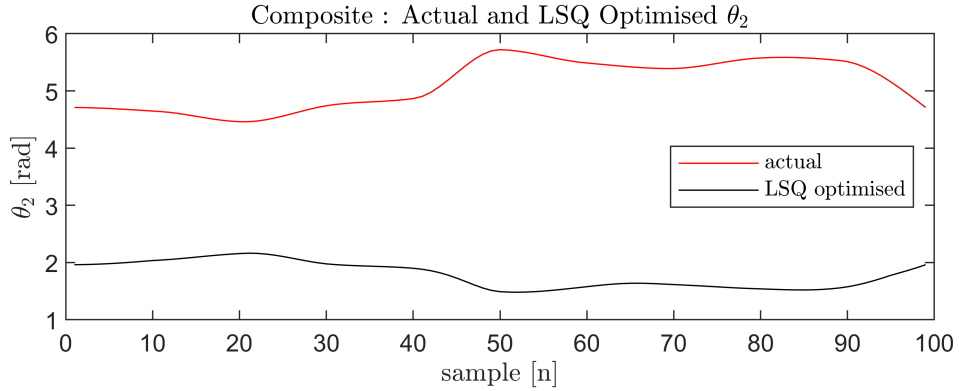

```
% ----------------------------
figure
subplot(2,1,1),plot(LSQ_thetaDATA(1,:),'Color', cK);
title('LSQ estimated $\theta_{1}$', 'Interpreter','latex')
xlabel('sample [n]', 'Interpreter','latex')
ylabel('$\theta_{1}$ [rad]', 'Interpreter','latex')
subplot(2,1,2),plot(LSQ_thetaDATA(2,:),'Color', cK);
```

```
title('LSQ estimated $\theta_{2}$', 'Interpreter','latex')
xlabel('sample [n]', 'Interpreter','latex')
ylabel('$\theta_{2}$ [rad]', 'Interpreter','latex')
```




```
% ------------------------------

figure
subplot(2,1,1),p2=plot(LSQ_thetaDATA(1,:),'Color', cK);
hold on
p1=plot(thetaINT_CLT(:,1), 'Color',cA);
hold off
title('Composite : Actual and LSQ Optimised $\theta_{1}$', 'Interpreter','latex')
legend([p1,p2],{'actual','LSQ optimised'},'Location','east','NumColumns',1, 'Interpreter','late
xlabel('sample [n]', 'Interpreter','latex')
ylabel('$\theta_{1}$ [rad]', 'Interpreter','latex')

subplot(2,1,2),p2=plot(LSQ_thetaDATA(2,:),'Color', cK);
hold on
p1=plot(thetaINT_CLT(:,2), 'Color',cA);
hold off
title('Composite : Actual and LSQ Optimised $\theta_{2}$', 'Interpreter','latex')
legend([p1,p2],{'actual','LSQ optimised'},'Location','east','NumColumns',1, 'Interpreter','late
xlabel('sample [n]', 'Interpreter','latex')
ylabel('$\theta_{2}$ [rad]', 'Interpreter','latex')
```






```
% ------------------------------------------------------------------
% Non-linear Least Squares (LSQ) Optimisation function definition :
% ------------------------------------------------------------------
```

```
function statenlLSQ = sol_theta(xo,yo,r1,r2,ex0,X_EEinpt,lb,ub) % <---- original function
%function statenlLSQ = sol_theta(xo,yo,r2,ex0,X_EEinpt,lb,ub) % option (i) : find k = x(3)
%options = optimoptions(@lsqnonlin,'Algorithm','levenberg-marquardt');
%options = optimoptions(@lsqnonlin,'Algorithm','levenberg-marquardt', 'Display','off'); % <----
options = optimoptions(@lsqnonlin,'Algorithm','trust-region-reflective', 'Display','off'); % <-

%fctn = @(x)( [                    ]    -      [   ]);
%fctn = @(x)( [       LHS          ]    -      [ RHS ]);
% LHS = [xEE;yEE]; <--- model with parameters
% RHS = [xEE_actual; yEE_actual]  <--- truth data

% xEE = xo + r1*cos(th1) + r2*cos(th1+th2); %  x coordinate of EE
% yEE = yo + r1*sin(th1) + r2*sin(th1+th2); %  y coordinate of EE

% xEE = xo + r1*cos(x(1)) + r2*cos(x(1)+x(2)); %  x coordinate of EE
% yEE = yo + r1*sin(x(1)) + r2*sin(x(1)+x(2)); %  y coordinate of EE
```

```
%fctn = @(x)([...
%    [xEE];[yEE]       ]   - ...
%    [  [xEEinpt];[yEEinpt] ...
%    ]);

%fctn = @(x)([...
%    [xo + x(3)*cos(x(1)) + 1.33*x(3)*cos(x(1)+x(2))];[(yo + x(3)*sin(x(1)) + 1.33*x(3)*sin(x(1
%    [  [X_EEinpt(1)];[X_EEinpt(2)] ...
%    ]);

%fctn = @(x)([...
%    [xo + x(3)*cos(x(1)) + 1.33*x(3)*cos(x(1)+x(2))];[-1.*(yo + x(3)*sin(x(1)) + 1.33*x(3)*sin
%    [  [X_EEinpt(1)];[X_EEinpt(2)] ...
%    ]);
% ------------------------------------------------------------------------------------------
% ------> original function
fctn = @(x)([...
    [xo + r1*cos(x(1)) + r2*cos(x(1)+x(2))];[(yo + r1*sin(x(1)) + r2*sin(x(1)+x(2)))]       ]
    [  [X_EEinpt(1)];[X_EEinpt(2)] ...
    ]);
% ------------------------------------------------------------------------------------------
% Consider these options:
% ------------------------
% (i) r1 = k*r2 with r2=0.3 <--- solve for k = x(3) : r1 = x(3)*r2
%fctn = @(x)([...
%    [xo + x(3)*r2*cos(x(1)) + r2*cos(x(1)+x(2))];[(yo + x(3)*r2*sin(x(1)) + r2*sin(x(1)+x(2)))
%    [  [X_EEinpt(1)];[X_EEinpt(2)] ...
%    ]);

% ------------------------------------------------------------------------------------------


%x = lsqnonlin(fctn,ex0,lb,ub,options) % <---- PRINT optimisation variable x values i.e [state
x = lsqnonlin(fctn,ex0,lb,ub,options); % <---- Do NOT PRINT optimisation variable x values i.e

%statenlLSQ = [x(1);x(2);x(3)];
statenlLSQ = [x(1);x(2)];
end
```