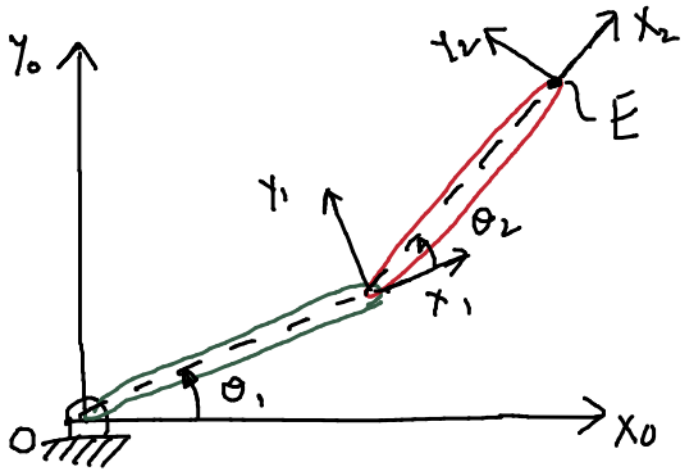
Table of Contents


```
clc
clear all
```

This example shows how to do the inverse kinematics for a 2 link manipulator and do an animation.



Specify DH parameters

```
global a1 alpha1 d1 a2 alpha2 d2 %the function end_effector will use these
                                 %variables so we make them globals
global x_des y_des z_des %where you want the end-effector

%D-H for links. Theta's are not set, because we need to find them.
a1 = 1; alpha1 = 0; d1=0;
a2 = 1; alpha2 = 0; d2=0;

%Location where we want the end-effector to be
x_des = 0.5; y_des = 0; z_des = 0;
```

```
%Try both these. They give different solutions.
%theta1 = 0.5; theta2 = 0.5; %initial guess for angles
%theta1 = -0.5; theta2 = -0.5; %initial guess for angles
%theta1 = -pi/2; theta2 = -pi/2; %initial guess for angles
theta1 = 0.1; theta2 = 0;
```

Solve for the values of theta that give the required end-effector pose.

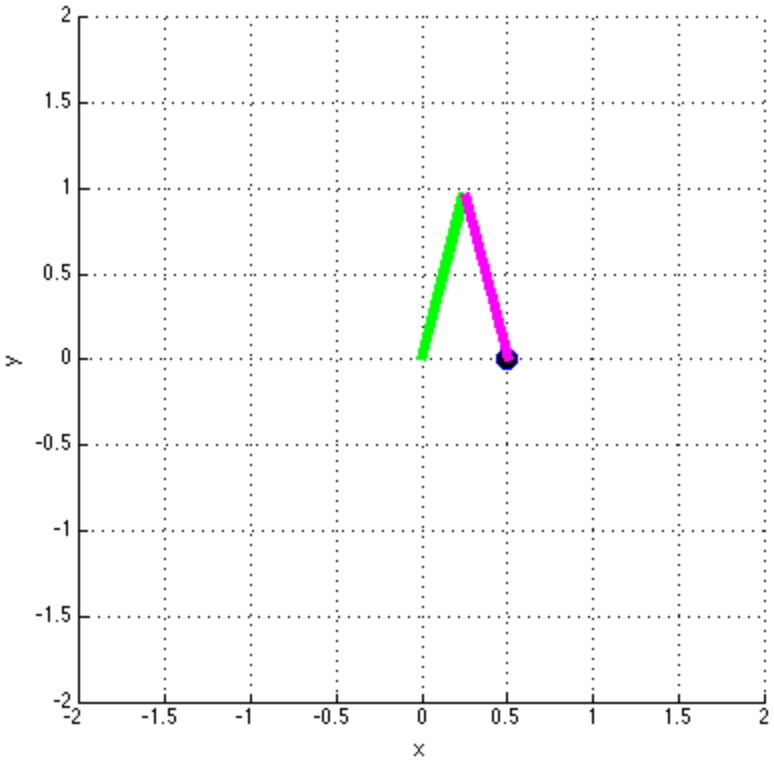
```
%fsolve solves for the roots for the equation X-XDES
[X,FVAL,EXITFLAG] = fsolve('end_effector_position',[theta1,theta2]);
theta1 = X(1);
theta2 = X(2);
disp(['Exitflag after running fsolve = ', num2str(EXITFLAG) ]) %Tells if fsolve co
              %1 means converged else not converged
              %Type help fsolve to know more about what different
              %EXITFLAG mean.

disp(['theta1 = ',num2str(theta1),'; theta2 = ',num2str(theta2)]);
```

```
Warning: Trust-region-dogleg algorithm of FSOLVE cannot handle
non-square systems; using Levenberg-Marquardt algorithm
instead.

Equation solved, fsolve stalled.

fsolve stopped because the relative size of the current step is less than
default value of the step size tolerance and the vector of function values
is near zero as measured by the default value of the function tolerance.



Exitflag after running fsolve = 4
theta1 = 1.3181; theta2 = -2.6362
```

Now visualize the results

```
close all %close all plots before starting

% Now let us plot the results.
A01 = DH(a1,alpha1,d1,theta1); %A^0_1
A12 = DH(a2,alpha2,d2,theta2); %A^1_2

%Location of joint 1
endOfLink1 = A01(1:3,4);

%Location of joint 2
A02 = A01*A12;
endOfLink2 = A02(1:3,4);

%Plot the point where we want the end-effector
```

```
plot3(x_des,y_des,z_des,'o','MarkerSize',10,'MarkerFaceColor','black');
hold on; %Ensures that dot on screen does not dissappear


%Draw line from origin to end of link 1
line([0 endOfLink1(1)],[0 endOfLink1(2)],[0 endOfLink1(3)],....
      'LineWidth',5,'Color','green');

%Draw line from end of link 1 to end of link 2
line([endOfLink1(1) endOfLink2(1)],...
     [endOfLink1(2) endOfLink2(2)],...
     [endOfLink1(3) endOfLink2(3)],...
     'LineWidth',5,'Color','magenta');

xlabel('x');
ylabel('y');
zlabel('z');
grid on; %if you want the grid to show up.
axis('equal'); %make the axis equal, to avoid scaling effect

view([0,90]); %top view (for 2-d plot). Comment out if you want a 3-d plot

%These set the x and y limits for the axis (will need adjustment)
xlim([-2 2]);
ylim([-2 2]);
zlim([-2 2]);
```



Published with MATLAB® R2013b