EENG 428 Introduction to Robotics Laboratory

Lab Session 6

Objective:

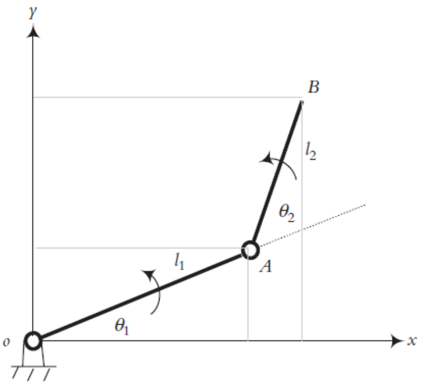
In this session, modelling fundamental matrices representing differential motion for robotic manipulators using Matlab is to be introduced.

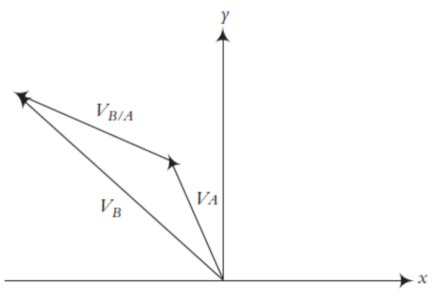
Background:

Differential motions are small movements Measured in a small period of time that can be used to derive velocity relationships between different parts of the mechanism.

Example

consider a simple 2-DOF mechanism where each link can independently rotate. The rotation of the first link θ_1 is measured relative to the reference frame, whereas the rotation of the second link θ_2 is measured relative to the first link.





- **The equations that describe the position**

$$x_B = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2)$$
$$y_B = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2)$$

- **Differentiating the equations with respect to the two variables θ1 and θ2**

$$dx_B = -l_1 \sin\theta_1 d\theta_1 - l_2 \sin(\theta_1 + \theta_2)(d\theta_1 + d\theta_2)$$
$$dy_B = l_1 \cos\theta_1 d\theta_1 + l_2 \cos(\theta_1 + \theta_2)(d\theta_1 + d\theta_2)$$

- **In matrix form**

$$\underbrace{\begin{bmatrix} dx_B \\ dy_B \end{bmatrix}}_{\text{Differential motion of B}} = \underbrace{\begin{bmatrix} -l_1 \sin\theta_1 - l_2 \sin(\theta_1+\theta_2) & -l_2 \sin(\theta_1+\theta_2) \\ l_1 \cos\theta_1 + l_2 \cos(\theta_1+\theta_2) & l_2 \cos(\theta_1+\theta_2) \end{bmatrix}}_{\text{Jacobian}} \underbrace{\begin{bmatrix} d\theta_1 \\ d\theta_2 \end{bmatrix}}_{\text{Differential motion of joints}}$$

- **If both sides of Equation are divided by dt**

$$\begin{bmatrix} dx_B \\ dy_B \end{bmatrix} \Big/ dt = \begin{bmatrix} -l_1 \sin\theta_1 - l_2 \sin(\theta_1+\theta_2) & -l_2 \sin(\theta_1+\theta_2) \\ l_1 \cos\theta_1 + l_2 \cos(\theta_1+\theta_2) & l_2 \cos(\theta_1+\theta_2) \end{bmatrix} \begin{bmatrix} d\theta_1 \\ d\theta_2 \end{bmatrix} \Big/ dt$$

- **We can verify the above equation by finding the velocity of point B using the velocity diagram**

$$\begin{bmatrix} v_{B_x} \\ v_{B_y} \end{bmatrix} = \begin{bmatrix} -l_1 \sin\theta_1 - l_2 \sin(\theta_1+\theta_2) & -l_2 \sin(\theta_1+\theta_2) \\ l_1 \cos\theta_1 + l_2 \cos(\theta_1+\theta_2) & l_2 \cos(\theta_1+\theta_2) \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}$$

We can conclude that, ***the joint differential motions, or velocities***, can be related to the ***differential motion, or velocity, of the hand*** in a robot with many degrees of freedom by using the **Jacobian matrix**.

Differential motions of a frame

Differential motion of a frame can be expressed as a Differential translation and rotation.

A *differential translation* is the translation of a frame at differential values. This expression $Trans(dx, dy, dz)$ means a differential move of the end-pose along the reference coordinate frame axes x, y, z. The *differential rotation* of x amount about the x-axis is denoted by Rot(x, δx). Similarly, rotations about y- and z-axes are denoted by Rot(y, δy) and Rot(z, δz).

All transformation matrices involve the calculation of sinusoids of an angle. In the case of differential motion, the following notation is usually accepted.

$$cos(dt) \sim 1 \qquad sin(dt) \sim dt$$

```
>> cos(0.001)        >> sin(0.001)
ans =                ans =
   1.0000               1.0000e-03
```

Consider a transformation matrix represents a rotation about the x-axis:

```
>> syms t
>> trotx(t)

ans =

[ 1,      0,       0, 0]
[ 0, cos(t), -sin(t), 0]
[ 0, sin(t),  cos(t), 0]
[ 0,      0,       0, 1]

>> trotx(0.001)

ans =

    1.0000         0         0         0
         0    1.0000   -0.0010         0
         0    0.0010    1.0000         0
         0         0         0    1.0000
```

Subsequently, the rotation matrices representing differential rotations about the x-, y-, and z-axes will be

$$Rot(x,\delta x) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & -\delta x & 0 \\ 0 & \delta x & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},\ Rot(y,\delta y) = \begin{bmatrix} 1 & 0 & \delta y & 0 \\ 0 & 1 & 0 & 0 \\ -\delta y & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},\ Rot(z,\delta z) = \begin{bmatrix} 1 & -\delta z & 0 & 0 \\ \delta z & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Sequential rotations are not commutative in general. However, the differential rotations are commutative if we neglect the higher differential terms

$>> sin(0.001) * sin(0.001)$

$ans\ =$

1.0000e-06 1000 times less than the original differential motion

neglecting the higher differential terms, results in

$$\mathrm{Rot}(x, \delta x)\ \mathrm{Rot}(y, \delta y) = \mathrm{Rot}(y, \delta y)\ \mathrm{Rot}(x, \delta x)$$

```
>> trotx(0.001)*troty(0.001)

ans =

    1.0000         0    0.0010         0
    0.0000    1.0000   -0.0010         0
   -0.0010    0.0010    1.0000         0
         0         0         0    1.0000

>> troty(0.001)*trotx(0.001)

ans =

    1.0000    0.0000    0.0010         0
         0    1.0000   -0.0010         0
   -0.0010    0.0010    1.0000         0
         0         0         0    1.0000
```

```
>> syms tx ty
>> trotx(tx)*troty(ty)

ans =

[           cos(ty),       0,            sin(ty), 0
[  sin(tx)*sin(ty), cos(tx), -cos(ty)*sin(tx), 0
[ -cos(tx)*sin(ty), sin(tx),  cos(tx)*cos(ty), 0
[                0,       0,                0, 1

>> troty(ty)*trotx(tx)

ans =

[  cos(ty), sin(tx)*sin(ty), cos(tx)*sin(ty), 0]
[        0,         cos(tx),        -sin(tx), 0]
[ -sin(ty), cos(ty)*sin(tx), cos(tx)*cos(ty), 0]
[        0,               0,               0, 1]
```

since the order of multiplication for differential rotations is not important, we can multiply differential rotations in any order. As a result, we can assume that a differential rotation about a general axis q is composed of three differential rotations about the three axes, in an arbitrary order

```
>> syms tx ty tz
>> trotx(tx)*troty(ty)*trotz(tz)

[                    cos(ty)*cos(tz),                    -cos(ty)*sin(tz),          sin(ty),  0]
[ cos(tx)*sin(tz) + cos(tz)*sin(tx)*sin(ty), cos(tx)*cos(tz) - sin(tx)*sin(ty)*sin(tz), -cos(ty)*sin(tx), 0]
[ sin(tx)*sin(tz) - cos(tx)*cos(tz)*sin(ty), cos(tz)*sin(tx) + cos(tx)*sin(ty)*sin(tz),  cos(tx)*cos(ty), 0]
[                                  0,                                   0,                0, 1]
```

Rot(q, δθ) = Rot(x, δx) Rot(y, δy) Rot(z, δz) = $\begin{bmatrix} 1 & -\delta z & \delta y & 0 \\ \delta z & 1 & -\delta x & 0 \\ -\delta y & \delta x & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

```
>> trotx(0.001)*troty(0.002)*trotz(0.003)

ans =

    1.0000   -0.0030    0.0020         0
    0.0030    1.0000   -0.0010         0
   -0.0020    0.0010    1.0000         0
         0         0         0    1.0000
```

The ***differential transformation of a frame*** is a combination of differential translations and rotations in any order. If we denote the original frame as T and assume that dT is the change in the frame T as a result of a differential transformation

$$T_{new} = T + dT = Rot(q, \delta\theta)\, Transl(dx, dy, dz)\, T$$

The equation could be modified to become

$$dT = (\, Rot(q, \delta\theta)\, Transl(dx, dy, dz) - I\,)\ T$$

$$dT = \begin{bmatrix} 1 & -\delta_z & \delta_y & d_x \\ \delta_z & 1 & -\delta_x & d_y \\ -\delta_y & \delta_x & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} T$$

$$dT = \begin{bmatrix} 0 & -\delta_z & \delta_y & d_x \\ \delta_z & 0 & -\delta_x & d_y \\ -\delta_y & \delta_x & 0 & d_z \\ 0 & 0 & 0 & 0 \end{bmatrix} T$$

$$dT = \Delta\, T$$

The operator Δ is called *differential operator*. Multiplying a frame by the differential operator Δ will yield the change in the frame. the differential operator is not a transformation matrix, or a frame. It is only an operator, and it yields the changes in a frame.

Exercise

Write the differential operator matrix for the following differential transformations

$d_x = 0.05$, $d_y = 0.03$, $d_z = 0.01$ units and $\delta_x = 0.02$, $\delta_y = 0.04$, $\delta_z = 0.06$ radians

The differential operator Δ represents a differential operator relative to the fixed reference frame. However, it is possible to define differential operator relative to the current frame itself ${}^{T}\Delta$. Since the differential operator ${}^{T}\Delta$ relative to the current frame, to find the changes in the frame we must post multiply the frame by ${}^{T}\Delta$

$$dT = \Delta\, T \;\; = T\, {}^{T}\Delta$$

Then by multiply the both sides by T^{-1}

$${}^{T}\Delta = T^{-1} \Delta\, T$$

Exercise

- Find the effect of a differential rotation of 0.15 radian about the z-axis followed by a differential translation of [0.2, 0.1, 0] on the given frame A.

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Find the location and the orientation of frame A after the move.
- Calculate ${}^{A}\Delta$

Jacobian analysis

Robot Jacobian matrix is a powerful tool that relates the **joint differential motions** of a robot to the **differential motion of its hand frame**

$$\begin{bmatrix} dx \\ dy \\ dz \\ \delta x \\ \delta y \\ \delta z \end{bmatrix} = \left[\quad \text{Robot Jacobian} \quad \right] \begin{bmatrix} d\theta_1 \\ d\theta_2 \\ d\theta_3 \\ d\theta_4 \\ d\theta_5 \\ d\theta_6 \end{bmatrix}$$

where dx, dy, and dz represent the differential motions of the hand along the x-, y-, and z-axes, δx; δy; and δz represent the differential rotations of the hand around the x-, y-, and z-axes, and $d\theta_i$ represents the differential motions of the joint i.

Calculation of the Jacobian matrix

Finding the Jacobian of a manipulator is not easy without a systematic approach. There are several ways to derive the Jacobian of serial manipulators. However, two methods are covered in the scope of this course. Paul's Method (deriving the Jacobian w.r.t the hand coordinate frame) and Vector cross product method (deriving the Jacobian w.r.t the reference coordinate frame)

➢ ***Paul's Method***

the Jacobian has 6 rows, and n columns; where n is the DOF of the manipulator. The calculation of the **i^{th} column of the Jacobian** is derived from ${}^{i-1}T_n$

$For\ i = 1:n$

Determine $${}^{i-1}T_n = \begin{bmatrix} nx & ox & ax & px \\ ny & oy & ay & py \\ nz & oz & az & pz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

if i is revolute

$$J_i = \begin{bmatrix} -nx\,py + ny\,px \\ -ox\,py + oy\,px \\ -ax\,py + ay\,px \\ nz \\ oz \\ az \end{bmatrix}$$

else if i is prismatic

$$J_i = \begin{bmatrix} nz \\ oz \\ az \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

end

end

Example

Given the following DH parameters for a 4DOF, RRPR manipulator.

	θ	d	a	α
Link 1	θ_1	d_1	a_1	0
Link 2	θ_2	0	a_2	pi
Link 3	0	d_3	0	0
Link 4	θ_4	d_4	0	0

```
%% Jacobian wrt hand frame (Paul's method)
syms t d al a
syms a1 d1 t1 a2 t2 d3 d4 t4
A=trotz(t)*transl(a,0,d)*trotx(al);
A1=subs(A,[al a d t],[0 a1 d1 t1]);
A2=subs(A,[al a d t],[pi a2 0 t2]);
A3=subs(A,[al a d t],[0 0 d3 0]);
A4=subs(A,[al a d t],[0 0 d4 t4]);
A34=A3*A4;
A234=A2*A34;
A1234=A1*A234;
J1=[A1234(1,4)*A1234(2,1)-A1234(2,4)*A1234(1,1)
A1234(1,4)*A1234(2,2)-A1234(2,4)*A1234(1,2)
A1234(1,4)*A1234(2,3)-A1234(2,4)*A1234(1,3)
A1234(3,1)
A1234(3,2)
A1234(3,3)];
J1=simplify(J1);
J2=[A234(1,4)*A234(2,1)-A234(2,4)*A234(1,1)
A234(1,4)*A234(2,2)-A234(2,4)*A234(1,2)
A234(1,4)*A234(2,3)-A234(2,4)*A234(1,3)
A234(3,1)
A234(3,2)
A234(3,3)];
J2=simplify(J2);
J3=[A34(3,1)
    A34(3,2)
    A34(3,3)
      0
      0
      0];
J3=simplify(J3);
J4=   [A4(1,4)*A4(2,1)-A4(2,4)*A4(1,1)
      A4(1,4)*A4(2,2)-A4(2,4)*A4(1,2)
      A4(1,4)*A4(2,3)-A4(2,4)*A4(1,3)
      A4(3,1)
      A4(3,2)
      A4(3,3)];
J4=simplify(J4);
Jh=[J1 J2 J3 J4];
```

➢ Vector Cross Product Method

This method is used to derive the Jacobian of the serial manipulator w.r.t the base coordinate frame. The calculation of the <u>***i^{th}* column of the Jacobian**</u> is derived from ${}^{0}T_{i-1}$

Determine ${}^{0}T_{n} = \begin{bmatrix} nx & ox & ax & px \\ ny & oy & ay & py \\ nz & oz & az & pz \\ 0 & 0 & 0 & 1 \end{bmatrix}$ and find $pn = \begin{bmatrix} px \\ py \\ pz \end{bmatrix}$

$For\ i = 1:n$

Determine ${}^{0}T_{i-1} = \begin{bmatrix} R & P \\ 0 & 1 \end{bmatrix}$ and find $R = [\,n \;\; o \;\; a\,]$ $\quad p = \begin{bmatrix} p1 \\ p2 \\ p3 \end{bmatrix}$

if i is revolute

$$J_i = \begin{bmatrix} a \times (pn - p) \\ a \end{bmatrix}$$

else if i is prismatic

$$J_i = \begin{bmatrix} a \\ 0 \end{bmatrix}$$

end

end

Exercise

Given the following DH parameters for a 4DOF, RRPR manipulator.

	θ	d	a	α
Link 1	θ_1	d_1	a_1	0
Link 2	θ_2	0	a_2	pi
Link 3	0	d_3	0	0
Link 4	θ_4	d_4	0	0

Write a matlab code can find the Jacobian matrix with respect to the base reference frame

➢ Converting the Jacobian from Hand frame to Tool Frame

The conversion is given by the equation

$${}^{0}J = M\, {}^{H}J \qquad\qquad {}^{H}J = M^{-1}\, {}^{0}J$$

Where M is defined as $M = \begin{bmatrix} R & 0 \\ 0 & R \end{bmatrix}$ R is the orientation matrix of the matrix ${}^{0}T_{n}$

Jacobian matrix and Differential Operator relation

Suppose a robot's joints are moved a differential amount and knowing the Jacobian matrix, then we can calculate dx, dy, dz, δx, δy and δz using the following equation

$$\begin{bmatrix} dx \\ dy \\ dz \\ \delta x \\ \delta y \\ \delta z \end{bmatrix} = \begin{bmatrix} & Jacobian\ Matrix & \end{bmatrix} \begin{bmatrix} d\theta_1 \\ d\theta_2 \\ d\theta_3 \\ d\theta_4 \\ d\theta_5 \\ d\theta_6 \end{bmatrix}$$

These values dx, dy, dz, δx, δy and δz can be substituted easily to form the differential operator

$$\Delta = \begin{bmatrix} 0 & -\delta_z & \delta_y & d_x \\ \delta_z & 0 & -\delta_x & d_y \\ -\delta_y & \delta_x & 0 & d_z \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

By using the following equation, we can calculate dT which is the change in the frame T as a result of a differential transformation

$$dT = \Delta\, T$$

Subsequently, dT can be used to locate the new position and orientation of the robot's hand

$$T_{new} = T + dT$$

Exercise

Given the hand frame of a 5-DOF robot as follow

$$^{0}T_5 = \begin{bmatrix} 1 & 0 & 0 & 5 \\ 0 & 0 & -1 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The numerical Jacobian for this instance is

$$^{H}J = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ -2 & 0 & 1 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The robot has a 2RP2R configuration. Find the new location of the hand after the differential motion.

Inverse Jacobian

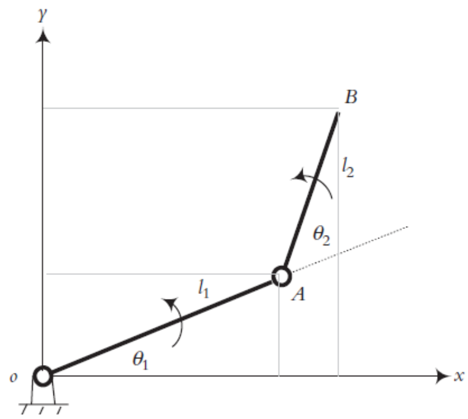
In order to calculate the differential motions needed at the joints of the robot for a desired hand differential motion we need to calculate the inverse of the Jacobian

$$\begin{bmatrix} d\theta_1 \\ d\theta_2 \\ d\theta_3 \\ d\theta_4 \\ d\theta_5 \\ d\theta_6 \end{bmatrix} = [\ J\ ]^{-1} \begin{bmatrix} dx \\ dy \\ dz \\ \delta x \\ \delta y \\ \delta z \end{bmatrix}$$

This means that knowing the inverse of the Jacobian, we can calculate how fast each joint must move, such that the robot's hand will yield a desired differential motion or velocity.

Example

consider a simple 2-DOF mechanism where each link can independently rotate. The rotation of the first link θ_1 is measured relative to the reference frame, whereas the rotation of the second link θ_2 is measured relative to the first link.



- The equations that describe the position of point B

$$x_B = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2)$$
$$y_B = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2)$$

- Let the manipulator link lengths as following

$l1 = 50$	$l2 = 40\ cm$

Let the point B move from initial point (0, 10) to the final point (50,10) in 12 steps. Assuming the joint-position for initial point is (pi/2, -pi) radians, calculate the joint displacements along the path using the inverse Jacobian and implement the solution in Matlab.

Let

$$p1 = (0,10) \quad \text{and} \quad pn = (50,10)$$

$$q1 = \ (pi/2, -pi)$$

$$n = 12 \quad \textit{and} \quad i = 1$$

For $k = \ 1\!: n$

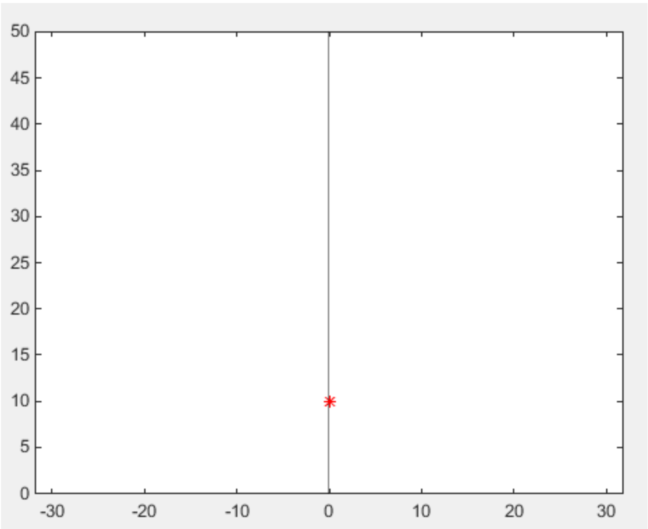
$$i = i + 1$$

$$pi = p1 + \left(\frac{k}{n}\right) * (\, pn - p0 \,)$$

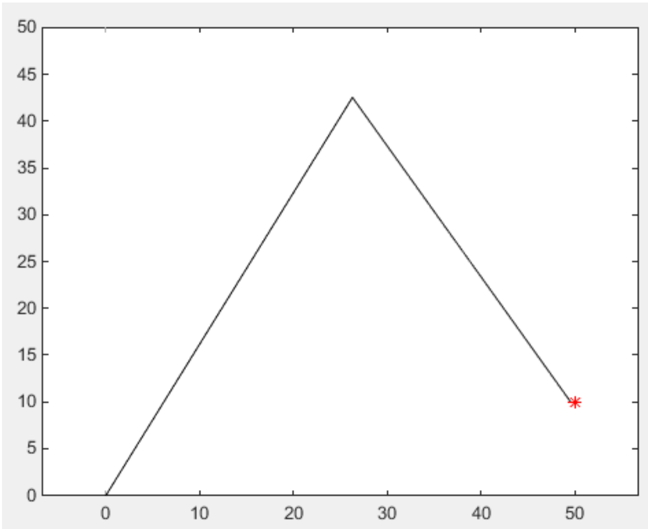
$$q_i = q_{i-1} + (\, J^{-1} * (p_i - p_{i-1})\,)$$

end

Implementation the following matlab code will yield the following plot



Initial position



Final position

```
% function for finding the inverse jacobian for the given 2 links manipulator
function Jinv = IJ(r);
a1 = 50 ; a2 = 40 ;
J = [-a1*r.S1-a2*r.S12 -a2*r.S12
a1*r.C1+a2*r.C12 a2*r.C12 ];
D = a1*a2*r.S2 ;
Jinv = [ J(2,2)/D -J(1,2)/D ;
-J(2,1)/D J(1,1)/D ];
end
```

```
clear all
clc; clf; close all;
r.a1 = 50 ; r.a2 = 40 ;
p0=[0 10] ;
q0= [pi/2 -pi];
pn=[50 10];
n=10;
i=1;
pp(i,:)=p0;
qq(i,:)=q0
for k=1:n
i=i+1;
pp(i,:)= p0 + (k/n)*(pn-p0)
q=qq(i-1,:);
r.C1=cos(q(1)); r.S1=sin(q(1));
r.C2=cos(q(2)); r.S2=sin(q(2));
r.C12=cos(q(1)+q(2)); r.S12=sin(q(1)+q(2));
qq(i,:)=q + transpose(IJ(r)* transpose( pp(i,:)-
pp(i-1,:) ));
end
x=[0 0 0]; y=[0 0 0];
for i=1:length(qq);
xo=x; yo=y;
x(1)=0; y(1)=0; % origin
x(2)= r.a1*cos(qq(i,1)); y(2)=r.a1*sin(qq(i,1));
x(3)= x(2)+r.a2*cos(qq(i,1)+qq(i,2));
y(3)= y(2)+r.a2*sin(qq(i,1)+qq(i,2));
plot(xo,yo,'w-'); hold on;
plot(x,y,'k-'); axis equal;
 hold on
 plot(50,10,'r*')
pause(0.2);
end
```

References:

[1] Craig, J. J. (2005). Introduction to robotics: mechanics and control (Vol. 3, pp. 48-70). Upper Saddle River: Pearson Prentice Hall.

[2] Tsai, L. W. (1999). Robot analysis: the mechanics of serial and parallel manipulators. John Wiley & Sons.

[3] Niku, S. (2010). Introduction to robotics. John Wiley & Sons.