

**Mechanics and Control of Robotic Manipulators**  
**Professor. Santhakumar Mohan**  
**Department of Mechanical Engineering**  
**Indian Institute of Technology, Palakkad**  
**Lecture No. 27**  
**Dynamic model derivation using Lagrange-Euler method in Matlab**

Welcome back to mechanics and control of robotic manipulator. In the last lecture what we have seen; how to derive the equation of motion using simple Newton Euler method. In this particular lecture, we are going to see the same thing with the help of Lagrangian Euler again MATLAB environment.

(Refer Slide Time: 0:33)



**Note:**

The presentation for this lecture have been prepared from a wide range of sources including books, websites/ pages, research articles, etc. These slides and this presentation are intended for purely educational purposes only.



Lagrange-Euler method 000 Matlab code 0000000

DYNAMIC MODEL DERIVATION USING LE METHOD THROUGH MATLAB

1 Lagrange-Euler method

2 Matlab code



SANTHAKUMAR MOHAN, IIT PALAKKAD  
MECHANICS AND CONTROL OF ROBOTIC MANIPULATORS



So, in the sense what we are trying to see here is, so, we are trying to recall what is Lagrangian Euler method and how we can put it in MATLAB.

(Refer Slide Time: 0:43)

Lagrange-Euler method 000 Matlab code 0000000

Euler-Lagrange or Lagrangian formulation method:

$$\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{q}_i} \right) - \frac{\partial L}{\partial q_i} = \tau_i \quad (1)$$

$$L = K.E. - P.E. \quad (2)$$

where, *K.E.* is the kinetic energy and *P.E.* is the potential energy of the system.



SANTHAKUMAR MOHAN, IIT PALAKKAD  
MECHANICS AND CONTROL OF ROBOTIC MANIPULATORS



So, the same example we are going to take, but before that let me recall what is the equation is coming here. So, here you can see that the *L* is a Lagrangian which would be kinetic energy minus potential energy. However, when we tried to derive the  $\tau_i$ , so, it is having two different things one is the partial derivative, the other one is time derivative.

In the sense so, you have to write the MATLAB code which is mixed of everything. So, that is what one of the complexity, however if you go with Mathematica or probably Maple it may be

straightforward however, so, since we started with the MATLAB we will see how to do it in MATLAB itself.


(Refer Slide Time: 1:31)


Lagrange-Euler method Matlab code  
○○○○○○○

General expression for the total kinetic energy ( $K.E.$ )

$$K.E. = \frac{1}{2} \left( \sum m_i {}^0\mathbf{v}_{ci}^T {}^0\mathbf{v}_{ci} + \sum {}^i\boldsymbol{\omega}^T \mathbf{I}_{ci} {}^i\boldsymbol{\omega} \right) \quad (3)$$

General expression for the total potential energy ( $P.E.$ )

$$P.E. = \sum \left( m_i {}^0\mathbf{g}^T {}^0\mathbf{P}_{ci} \right) = \sum \left( m_i {}^0\mathbf{a}^T {}^0\mathbf{P}_{ci} \right) \quad (4)$$


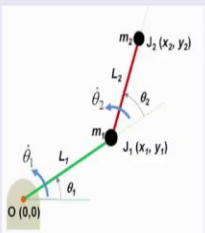

 SANTHAKUMAR MOHAN, IIT PALAKKAD  
MECHANICS AND CONTROL OF ROBOTIC MANIPULATORS


So, in that sense, we are trying to recall what is Lagrangian and we will see like how we can derive the kinetic energy and how to derive the potential energy and all.

(Refer Slide Time: 1:38)

Lagrange-Euler method Matlab code  
○○○○○○○

Example: A planar 2R serial manipulator

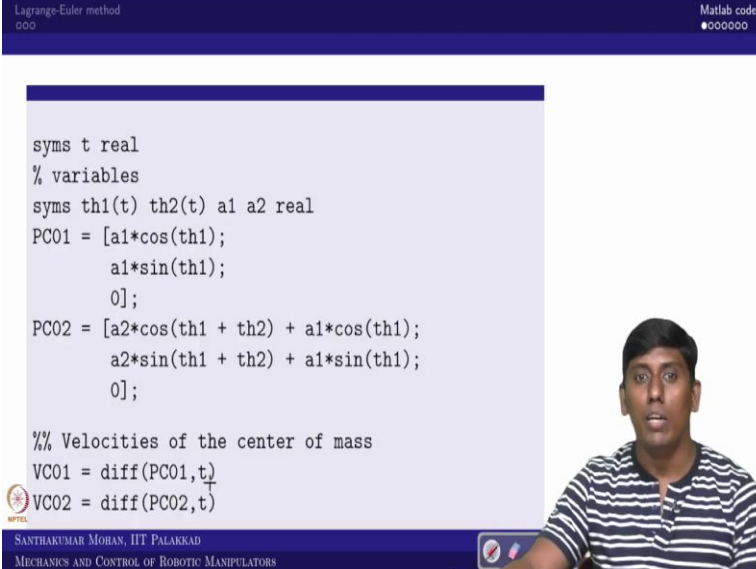
 SANTHAKUMAR MOHAN, IIT PALAKKAD  
MECHANICS AND CONTROL OF ROBOTIC MANIPULATORS

So, now, based on that the same example we will take and right now, what are the things will come. So, you need to find out what is the center of mass or location of center of mass  $m_1$  and

$m_2$  we need to find which is we have written as  $x_1$  and  $y_1$  and  $x_2$  and  $y_2$ . So, now, we need to take the derivative of that, so, which is  $\dot{x}_1$  and  $\dot{y}_1$  and  $\dot{x}_2$  and  $\dot{y}_2$ .

Once we derive this, so, based on that we will go with what you call kinetic energy and potential energy. Once we obtained that, then we will take the time derivative or before that partial derivative then we will do the time derivative, and we will map the equation.

(Refer Slide Time: 2:18)



```
Lagrange-Euler method
000
Matlab code
••••••••

syms t real
% variables
syms th1(t) th2(t) a1 a2 real
PC01 = [a1*cos(th1);
        a1*sin(th1);
        0];
PC02 = [a2*cos(th1 + th2) + a1*cos(th1);
        a2*sin(th1 + th2) + a1*sin(th1);
        0];

%% Velocities of the center of mass
VC01 = diff(PC01,t)
VC02 = diff(PC02,t)
```

SANTHAKUMAR MOHAN, IIT PALAKKAD  
MECHANICS AND CONTROL OF ROBOTIC MANIPULATORS

So, in that sense what we are trying to do first we are trying to find out what is the location of center of mass. So, for finding that we can use again the same equation what we have used in the DH table and dh representation your forward kinematic model will give location of center of masses, which I have taken already PC01 is a location of center of mass of link 1 and PC02 is for 2, link 2.

So, this is we all recall, this is straight forward this is equivalent  $x_1$ , this is equivalent to  $y_1$ , and this is  $x_2$ , this is  $y_2$  and  $z_1$  and  $z_2$  are 0, because it is in a plane. Right now, what we are calling the theta 1 is function of time. And we say that the  $t$  is time, which is real variable. So,  $a_1$  and  $a_2$  are constant, but theta 1 and theta 2 are time dependent variables. So, in that sense what one can do so we can take the velocity of center of masses simply take the time differentiation.

The time differentiation I am doing as a difference as the command or the function in MATLAB. So, PC1 comma time in the sense I am taking time as the variable. So, I am differentiating the

PC01 with respect to time. So, now what this will do VC01 and VC02 will you the velocity values.

(Refer Slide Time: 3:44)

```
Lagrange-Euler method
000
Matlab code
0●00000

VC01 = [ -a1*sin(th1)*th1dot;
         a1*cos(th1)*th1dot;
         0];
VC02 = [ - a2*sin(th1 + th2)*(th1dot + th2dot) - a1*sin(th1)*th1dot;
         a2*cos(th1 + th2)*(th1dot + th2dot) + a1*cos(th1)*th1dot;
         0];

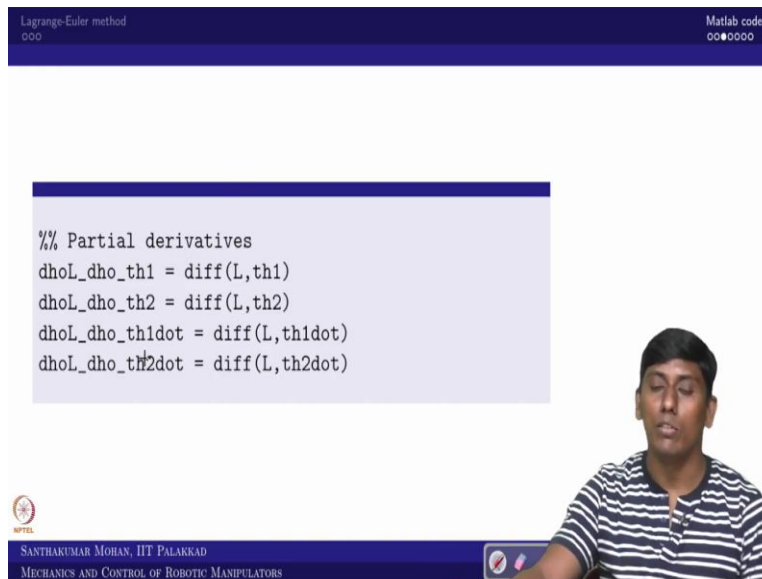
KE = 1/2 * (m1*VC01'*VC01+m2*VC02'*VC02);
PE = m1*g*PC01(2) + m2*g*PC02(2);
L = KE - PE;
```

SANTHAKUMAR MOHAN, IIT PALAKKAD  
MECHANICS AND CONTROL OF ROBOTIC MANIPULATORS

So, that velocity value I am rewriting in theta 1 dot and theta 2 dot manner. So, now what I got VC01 and VC02 and now what I wanted? I wanted the kinetic energy. So, the kinetic energy is half into  $m \text{VC1}^T \text{VC1}$ , VC01. So, this is what we have seen how to write in as you can see syntax and the similar way the second link kinetic energy also. So, here fortunate, there is no rotational kinetic energy, so we can leave it, only the translational kinetic energy we have taken.

Similarly, the potential energy what that would be the  $y_1$ , because it is vertically down, the  $y_1$  we need to write. So, the  $y_1$  is PC01 of 2 is  $y_1$ ,  $y_2$  is PC02 of 2. So, in the sense, so I hope there is a typo, this is the  $m_2$ . So, I will correct it in the MATLAB code and the handout also I will correct this is so e  $m_2$ . So, please correct that this is  $m_2$ . So, in that sense, the potential energy is giving this point. So, the Lagrange is a kinetic energy minus potential energy, so we can write.

(Refer Slide Time: 4:56)



The slide displays MATLAB code for calculating partial derivatives. The code is as follows:

```
Lagrange-Euler method
ooo
Matlab code
oo●oooo

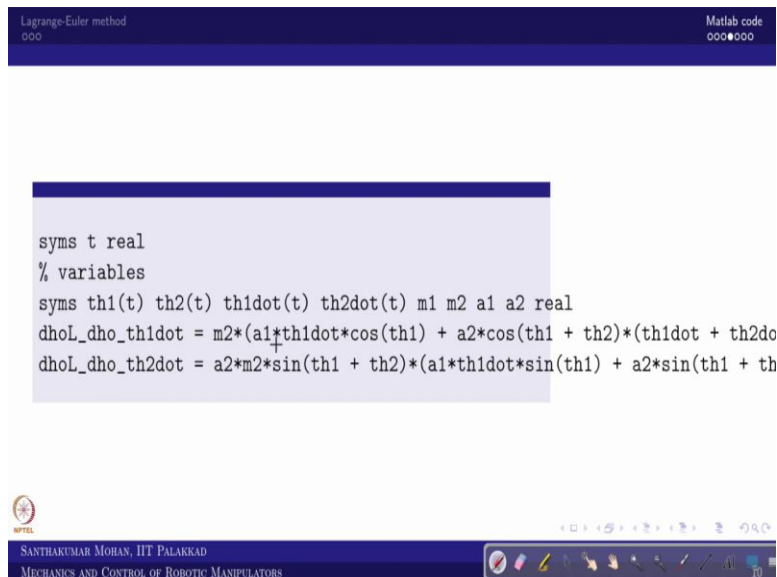
%% Partial derivatives
dhoL_dho_th1 = diff(L,th1)
dhoL_dho_th2 = diff(L,th2)
dhoL_dho_th1dot = diff(L,th1dot)
dhoL_dho_th2dot = diff(L,th2dot)

SANTHAKUMAR MOHAN, IIT PALAKKAD
MECHANICS AND CONTROL OF ROBOTIC MANIPULATORS
```

A small video inset shows a man in a striped shirt speaking.

So, now what we need to do, we need to do the partial derivative, for doing the partial derivative once you know the; what you call the kinetic energy and potential energy you calculate Lagrangian, then you take the partial derivative with respect to theta 1 and theta 2 and theta 1 dot and theta 2 dot. So, these all need to be done in a separate file, because earlier what we defined theta 1 is a function of time. So, now, here it is without it is independent variable, that way only we are considering. So, now, in that sense we can derive this.

(Refer Slide Time: 5:31)



The slide displays MATLAB code for defining symbolic variables and calculating partial derivatives. The code is as follows:

```
Lagrange-Euler method
ooo
Matlab code
ooo●ooo

syms t real
% variables
syms th1(t) th2(t) th1dot(t) th2dot(t) m1 m2 a1 a2 real
dhoL_dho_th1dot = m2*(a1*th1dot*cos(th1) + a2*cos(th1 + th2)*(th1dot + th2do
dhoL_dho_th2dot = a2*m2*sin(th1 + th2)*(a1*th1dot*sin(th1) + a2*sin(th1 + th

SANTHAKUMAR MOHAN, IIT PALAKKAD
MECHANICS AND CONTROL OF ROBOTIC MANIPULATORS
```

```

d_dhoL_dho_th1dot_dt = diff(dhoL_dho_th1dot,t)
d_dhoL_dho_th2dot_dt = diff(dhoL_dho_th2dot,t)

rr_file = fopen('simple_d_dho','w');
fprintf(rr_file,'n1 = %s;\n',char(simplify(d_dhoL_dho_th1dot_dt)))
fprintf(rr_file,'n2 = %s;\n',char(simplify(d_dhoL_dho_th2dot_dt)))
fclose(rr_file)

```



So, once we derive the theta 1 or dhoL by dho theta 1 dot, and dhoL by dho theta 2 dot, would be coming into this way. And this we can rewrite, and then now, we are actually taking the time derivative. So, once you all do this, so, what you can see, you have already taken dhoL by dho theta 1 and dhoL by dho theta 2, so then you can simply substitute it, and then do it. Since these equations are lengthy, I have used one file transfer. So, in the sense this simple MATLAB code itself, I have tried to explain all the background work.

(Refer Slide Time: 6:11)

```

clear all; close all; clc;
syms th1 th2 th1dot th2dot th1ddot th2ddot a1 a2 m1 m2 g real
d_dhoL_dho_th1dot_dt = a1^2*m1*th1ddot + a1^2*m2*th1ddot + a2^2*m2*th1ddot +
d_dhoL_dho_th2dot_dt = a2^2*m2*th1ddot + a2^2*m2*th2ddot + a1*a2*m2*cos(th2)
dhoL_dho_th1 = -g*m1*(a2*cos(th1 + th2) + a1*cos(th1)) - a1*g*m1*cos(th1);
dhoL_dho_th2 = a2*m2*cos(th1 + th2)*(a1*th1dot*sin(th1) + a2*sin(th1 + th2))*

```



So, in that sense, we can write this equation in this form. So, you know, dhoL by dho theta 1 dot, so, then I could take the time derivative also. So, in the sense dhoL by dho theta 1 dot I have

taken time derivative in the sense d of dhoL by dt of dho theta 1 that I have taken. So, in that sense, this is what the case and this is dhoL by dho theta 1.

(Refer Slide Time: 6:40)

```

Lagrange-Euler method
000
Matlab code
0000000

%% Vector of inputs
tau1 = d_dhoL_dho_th1dot_dt - dhoL_dho_th1
tau2 = d_dhoL_dho_th2dot_dt - dhoL_dho_th2

tau1 = a1^2*m1*th1ddot + a1^2*m2*th1ddot + a2^2*m2*th1ddot
+ a2^2*m2*th2ddot + a2*g*m1*cos(th1 + th2) + 2*a1*g*m1*cos(th1)
- a1*a2*m2*th2dot^2*sin(th2) + 2*a1*a2*m2*th1dot*cos(th2)
+ a1*a2*m2*th2ddot*cos(th2) - 2*a1*a2*m2*th1dot*th2dot*sin(th2)
tau2 = a2*(a1*m2*sin(th2)*th1dot^2 + a2*m2*th1ddot + a2*m2*th2ddot
+ g*m1*cos(th1 + th2) + a1*m2*th1dot*cos(th2))

```

SANJAY KUMAR MOHAN, IIT PALAKKAD  
MECHANICS AND CONTROL OF ROBOTIC MANIPULATORS

So, this minus of this would give tau. So, this is what we have taken. So, now, tau 1 and tau 2 we have obtained. So, now, these are the bigger equations, which we have obtained. So, this is the equation which we obtained even in the Newton Euler method you can recall, this is what we obtained in the Newton Euler. So, now, in that case, if we go to MATLAB environment, what we need to do, so, we need to do it several things.

(Refer Slide Time: 7:07)

```

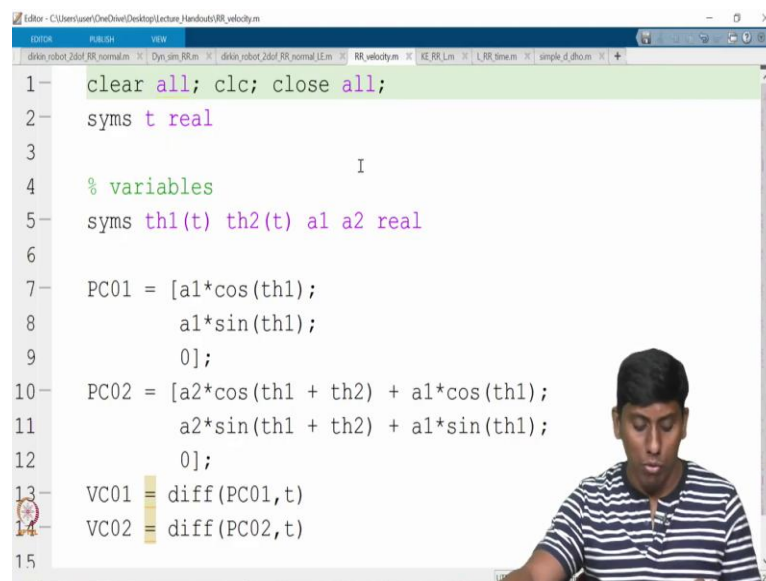
Editor - C:\Users\user\OneDrive\Desktop\RR_velocity.m
EDITOR  PUBLISH  VIEW
dikin_robot_2dof_RR_normal.m  Dyn_sim_RR.m  dikin_robot_2dof_RR_normal(E.m  RR_velocity.m  KE_RR_Lin  L_RR_Sim.m  simple_d_dho.m
1  syms t real
2
3  % variables
4  syms th1(t) th2(t) a1 a2 real
5
6  PC01 = [a1*cos(th1);
7         a1*sin(th1);
8         0];
9  PC02 = [a2*cos(th1 + th2) + a1*cos(th1);
10         a2*sin(th1 + th2) + a1*sin(th1);
11         0];
12  VC01 = diff(PC01,t)
13  VC02 = diff(PC02,t)
14
15

```

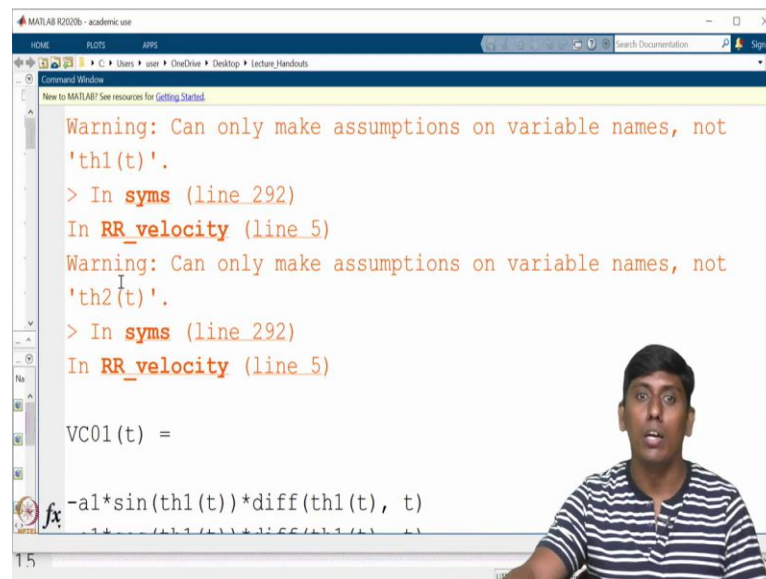


So, what first, so, first we need to calculate what you call the velocity, so, this is what we did in the first two MATLAB code. So, I have assumed that the theta 1 and theta 2 are time dependent, so, I have actually given PC1 and PC2. So, now, I differentiate with respect to time if I run this, you can see why I am doing it here, I will make it one more command just for your benefit.

(Refer Slide Time: 7:38)



```
1 clear all; clc; close all;
2 syms t real
3
4 % variables
5 syms th1(t) th2(t) a1 a2 real
6
7 PC01 = [a1*cos(th1);
8         a1*sin(th1);
9         0];
10 PC02 = [a2*cos(th1 + th2) + a1*cos(th1);
11         a2*sin(th1 + th2) + a1*sin(th1);
12         0];
13 VC01 = diff(PC01,t)
14 VC02 = diff(PC02,t)
15
```



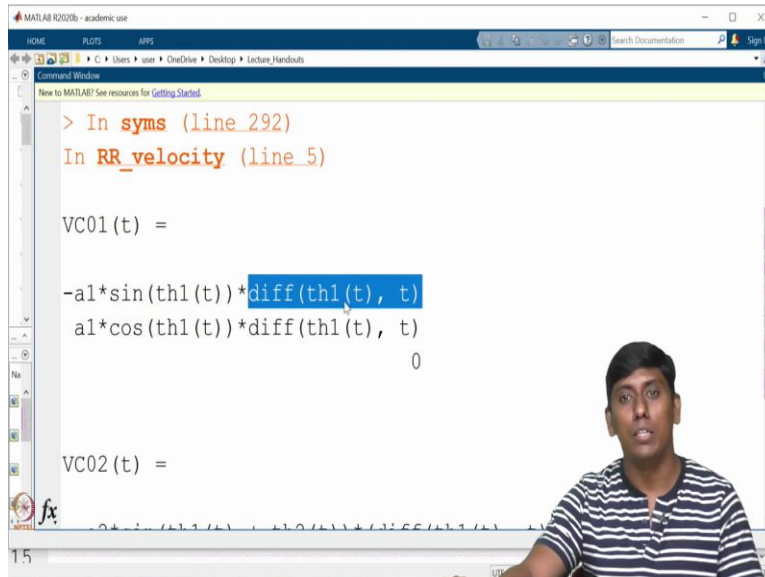
```
Warning: Can only make assumptions on variable names, not
'th1(t)'.
> In syms (line 292)
In RR_velocity (line 5)
Warning: Can only make assumptions on variable names, not
'th2(t)'.
> In syms (line 292)
In RR_velocity (line 5)

VC01(t) =
-a1*sin(th1(t))*diff(th1(t), t)
```

So, clear all and close all and your close all is not required here, because there is no figure window here. So, now, I just ran this code, so there was no error command. So, you can see like in this case, so VC1 and VC2 is coming. So, this is what I was saying that assumption is constraint, because we assume that theta 1 and theta 2 are a function of time that is what it is

giving a warning. It can be actually like, you can write theta 1 equal to symbol of theta 1 of time also you can write then this warning will not come, but for betterment, you can write it as per the earlier definition itself.

(Refer Slide Time: 8:19)



The image shows a MATLAB Command Window with the following text:

```

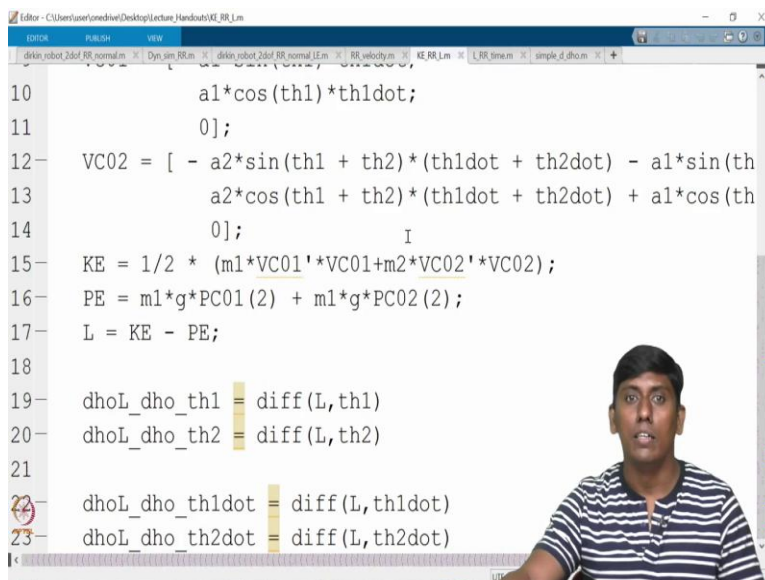
> In syms (line 292)
In RR_velocity (line 5)

VC01(t) =

-a1*sin(th1(t))*diff(th1(t), t)
 a1*cos(th1(t))*diff(th1(t), t)
                                0

VC02(t) =

```



The image shows a MATLAB Editor window with the following code:

```

10         a1*cos(th1)*th1dot;
11         0];
12     VC02 = [ - a2*sin(th1 + th2)*(th1dot + th2dot) - a1*sin(th
13             a2*cos(th1 + th2)*(th1dot + th2dot) + a1*cos(th
14             0];
15     KE = 1/2 * (m1*VC01'*VC01+m2*VC02'*VC02);
16     PE = m1*g*PC01(2) + m1*g*PC02(2);
17     L = KE - PE;
18
19     dhoL_dho_th1 = diff(L,th1)
20     dhoL_dho_th2 = diff(L,th2)
21
22     dhoL_dho_th1dot = diff(L,th1dot)
23     dhoL_dho_th2dot = diff(L,th2dot)

```

```

4      a1*sin(th1);
5      0];
6      PC02 = [a2*cos(th1 + th2) + a1*cos(th1);
7             a2*sin(th1 + th2) + a1*sin(th1);
8             0];
9      VC01 = [ -a1*sin(th1)*th1dot;
10             a1*cos(th1)*th1dot;
11             0];
12     VC02 = [ - a2*sin(th1 + th2)*(th1dot + th2dot) - a1*sin(th1)*th1dot;
13             a2*cos(th1 + th2)*(th1dot + th2dot) - a1*cos(th1)*th1dot;
14             0];
15     KE = 1/2 * (m1*VC01'*VC01+m2*VC02'*VC02);
16     PE = m1*g*PC01(2) + m1*g*PC02(2);
17     L = KE - PE;

```

So now, you can see here, this is giving us a difference. So now, this difference of theta 1 of t comma t is supposed to be theta 1 dot. So, for that, I copy this, and I rewrite that, I will write that here. So, I copied this, and I rewrite that as a theta 1 dot and here theta 2 dot and the potential energy I said that typo is there. So, there is the correction. So, now, you can see that the kinetic energy we calculated potential energy we have calculated, and the Lagrangian we have calculated.

So, then you can see here, so, the theta 1 theta 2 theta 1 theta 2 dot all independent variable, in the sense there is no time as a t as defined. So, now, if I actually take this dhoL by dho theta 1 and dhoL by theta 1 dot will be found.

(Refer Slide Time: 9:20)

```

MATLAB R2020b - academic use
HOME PLOTS APPS
C:\Users\user\Desktop\Lecture_Handouts
Command Window
New to MATLAB? See resources for Getting Started.
dhoL_dho_th1dot =

a2*m2*cos(th1 + th2)*(a1*th1dot*sin(th1) + a2*sin(th1 +
th2)*(th1dot + th2dot))*(th1dot + th2dot) - a2*m2*sin(th1 +
th2)*(a1*th1dot*cos(th1) + a2*cos(th1 + th2)*(th1dot +
th2dot))*(th1dot + th2dot) - a2*g*m2*cos(th1 + th2)

dhoL_dho_th1dot =

m2*(a1*th1dot*cos(th1) + a2*cos(th1 + th2)*(th1dot
th2dot))*(a2*cos(th1 + th2) + a1*cos(th1)) +
fx m2*(a1*th1dot*sin(th1) + a2*sin(th1 + th2)

```

```

MATLAB R2020b - academic use
HOME PLOTS APPS
C:\Users\user\Desktop\Lecture_Handouts
Command Window
New to MATLAB? See resources for Getting Started.
th2)*(th1dot + th2dot))*(th1dot + th2dot) - a2*m2*sin(th1 +
th2)*(a1*th1dot*cos(th1) + a2*cos(th1 + th2)*(th1dot +
th2dot))*(th1dot + th2dot) - a2*g*m2*cos(th1 + th2)

dhoL_dho_th1dot =

m2*(a1*th1dot*cos(th1) + a2*cos(th1 + th2)*(th1
th2dot))*(a2*cos(th1 + th2) + a1*cos(th1)) +
m2*(a1*th1dot*sin(th1) + a2*sin(th1 + th2)*(th1
th2dot))*(a2*sin(th1 + th2) + a1*sin(th1)) +
fx a1^2*m1*th1dot*cos(th1)^2 + a1^2*m1*th1dot*

```

```

MATLAB R2020b - academic use
HOME PLOTS APPS
C:\Users\user\Desktop\Lecture_Handouts
Command Window
New to MATLAB? See resources for Getting Started.

dhoL_dho_th2dot =

a2*m2*sin(th1 + th2)*(a1*th1dot*sin(th1) + a2*sin(th1 +
th2)*(th1dot + th2dot)) + a2*m2*cos(th1 +
th2)*(a1*th1dot*cos(th1) + a2*cos(th1 + th2)*(th1dot +
th2dot))

>>
>>
>>
>>
fx >>

```

```

MATLAB R2020b - academic use
HOME PLOTS APPS
C:\Users\user\Desktop\Lecture_Handouts
Command Window
New to MATLAB? See resources for Getting Started.

dhoL_dho_th1dot =

m2*(a1*th1dot*cos(th1) + a2*cos(th1 + th2)*(th1dot +
th2dot))*(a2*cos(th1 + th2) + a1*cos(th1)) +
m2*(a1*th1dot*sin(th1) + a2*sin(th1 + th2)*(th1dot +
th2dot))*(a2*sin(th1 + th2) + a1*sin(th1)) +
a1^2*m1*th1dot*cos(th1)^2 + a1^2*m1*th1dot*sin(th1)^2

dhoL_dho_th2dot =

a2*m2*sin(th1 + th2)*(a1*th1dot*sin(th1) +
th2)*(th1dot + th2dot)) + a2*m2*cos(th1 +

```

So, now, if I do this, so, what I will get I will get a bigger equations, you can see like this is dhoL by dho theta 1 and this is dhoL by dho theta 2. This is dhoL by dho theta 1 dot and this is is dhoL by dho theta 2 dot. So, I just wanted to show you can see right. So, now, these two are supposed to be taken as a time derivative what one can do, you can copy these two terms, and then that theta 1 and theta 2 are the time dependent. So, that is what we are doing it in the third thing.

(Refer Slide Time: 9:54)

```

Editor - C:\Users\user\OneDrive\Desktop\Lecture_Handouts\1_RR_time.m
2 syms t real
3 % variables
4 syms th1(t) th2(t) th1dot(t) th2dot(t) m1 m2 a1 a2 real
5 dhoL_dho_th1dot = m2*(a1*th1dot*cos(th1) + a2*cos(th1 + th
6 dhoL_dho_th2dot = a2*m2*sin(th1 + th2)*(a1*th1dot*sin(th1)
7
8 d_dhoL_dho_th1dot_dt = diff(dhoL_dho_th1dot,t)
9 d_dhoL_dho_th2dot_dt = diff(dhoL_dho_th2dot,t)
10
11 rr_file = fopen('simple_d_dho','w');
12
13 fprintf(rr_file,'n1 = %s;\n',char(simplify(d_dhoL_dho_th1dot_dt)));
14 fprintf(rr_file,'n2 = %s;\n',char(simplify(d_dhoL_dho_th2dot_dt)));
15 fclose(rr_file)

```

```

MATLAB R2019b - academic use
HOME PLOTS APPS
Command Window
How to MATLAB? See resources for Getting Started
d_dhoL_dho_th1dot_dt(t) =
m2*(a2*cos(th1(t) + th2(t))*(diff(th1(t), t) + diff(th2(t),
t)) + a1*cos(th1(t))*diff(th1(t),
t))*(a1*sin(th1(t))*th1dot(t) + a2*sin(th1(t) +
th2(t))*(th1dot(t) + th2dot(t))) - m2*(a2*sin(th1(t) +
th2(t))*(diff(th1(t), t) + diff(th2(t), t)) +
a1*sin(th1(t))*diff(th1(t), t))*(a2*cos(th1(t) +
th2(t))*(th1dot(t) + th2dot(t)) + a1*cos(th1(t))*diff(th1(t),
t)) + m2*(a1*cos(th1(t) + a2*cos(th1(t) +
th2(t)))*(a2*cos(th1(t) + th2(t))*(diff(th1dot(t), t) +
diff(th2dot(t), t)) + a1*cos(th1(t))*diff(th1(t), t)) +
a2*sin(th1(t) + th2(t))*(th1dot(t) +

```

So, you can see, so here, so I am writing as theta 1 dot and theta 2 dot and theta 1 and theta 2 all are time dependent. So, I will even make it because I run it as independent. So, I will write it clear all and close. So, so now you can see like this is what we have obtained, and I am taking a time derivative, the time derivative I am putting into one simple file. So, that file I am writing theta 1 dho 1, so that I can show the other one. So, now, if I run this what it will do, so, it will give the time derivative of these two terms. So, now I will run this.

So, now although the display in the command window, the command window will come. So, but what I have made it I made it for what you call in the file also. So, now, in that sense, I open this a simple underscore d and dho theta 1 for example, I just open that.



(Refer Slide Time: 11:10)

```
1 n1 = a1^2*m1*diff(th1dot(t), t) + a1^2*m2*diff(th1dot(t), t) + a
2 n2 = a2^2*m2*diff(th1dot(t), t) + a2^2*m2*diff(th2dot(t), t) + a
3
...
12
13 fprintf(rr_file,'n1 = %s;\n',char(simplify(d...th1d
14 fprintf(rr_file,'n2 = %s;\n',char(simplif...d
15 fclose(rr_file)
```

I hope simple, so, I can see that is all file, you can see. So, now, this is having a difference command because you see like we have taken time differentiation, now probably would have already realized that what is the benefit of what you call a Newton Euler method, you see Newton Euler method, we did not go back and forth, we did not create these number of files.

So, now, you can see this n1 and n2, which is we have written. So, which is equivalent to the time derivative that is having time differentiation of theta 1 dot theta 2 dot and it is having everything is time dependent. So, this you have to copy and replace this theta 1 differentiation as

theta 1 double dot and differentiation of theta 1 of t comma t as theta 1 dot. Like that, you have to modify. So, you can see like here these are all variations and modification you have to do.

(Refer Slide Time: 12:26)

The image consists of two screenshots from a video lecture. The top screenshot shows a text editor window with the following code:

```
1 n1 = a1^2*m1*diff(th1dot(t), t) + a1^2*m2*diff(th1dot(t), t) + a
2 n2 = a2^2*m2*diff(th1dot(t), t) + a2^2*m2*diff(th2dot(t), t) + a
3
```

A 'Find & Replace' dialog box is open, showing 'Find what: diff(th1dot(t), t)' and 'Replace with: th1ddot'. The 'Replace All' button is highlighted. The bottom screenshot shows the same code after replacement:

```
1 n1 = a1^2*m1*th1ddot + a1^2*m2*th1ddot + a2^2*m2*th1ddot + a2^2*
2 n2 = a2^2*m2*th1ddot + a2^2*m2*diff(th2dot(t), t) + a1*a2*m2*cos
3
```

The 'Find & Replace' dialog box is still open, and the 'Replace All' button is highlighted. The status bar at the bottom left of the editor window shows '6 of 6 items replaced'.

So, when I copy this and modify for example, if I say this, so, I call this Ctrl F if I put So, this theta 1 dot of this differentiation I can write as theta 1 double dot. So, then what you can see that I can replace everything, so, then that would make.



(Refer Slide Time: 12:41)

1  $n1 = a1^2 * m1 * th1ddot + a1^2 * m2 * th1ddot + a2^2 * m2 * th1ddot + a2^2 * m2 * th2ddot$   
2  $n2 = a2^2 * m2 * th1ddot + a2^2 * m2 * diff(th2dot(), t) + a1 * a2 * m2 * cos(th2(t)) * th2dot$   
3

Find & Replace dialog box:  
Find what: `diff(th2dot(), t)`  
Replace with: `th2dot`  
Look in: Editor - Current File (simple\_d\_gho1)  
 Match case  Whole word  Wrap around

6 of 6 items replaced

1  $n1 = a1^2 * m1 * th1ddot + a1^2 * m2 * th1ddot + a2^2 * m2 * th1ddot + a2^2 * m2 * th2ddot$   
2  $n2 = a2^2 * m2 * th1ddot + a2^2 * m2 * th2ddot + a1 * a2 * m2 * cos(th2(t)) * th2dot$   
3

Find & Replace dialog box:  
Find what: `diff(th2dot(), t)`  
Replace with: `th2dot`  
Look in: Editor - Current File (simple\_d\_gho1)  
 Match case  Whole word  Wrap around

3 of 3 items replaced

```

Editor - C:\Users\user\onedrive\Desktop\Lecture_Handouts\simple_d_dho.m
1 clear all; close all; clc;
2 syms th1 th2 th1dot th2dot th1ddot th2ddot a1 a2 m1 m2 g re
3 d_dhoL_dho_th1dot_dt = a1^2*m1*th1ddot + a1^2*m2*th1ddot + a
4 d_dhoL_dho_th2dot_dt = a2^2*m2*th1ddot + a2^2*m2*th2ddot + a
5 dhoL_dho_th1 = - g*m1*(a2*cos(th1 + th2) + a1*cos(th1)) - a1
6 dhoL_dho_th2 = a2*m2*cos(th1 + th2)*(a1*th1dot*sin(th1) + a2
7
8 tau1 = d_dhoL_dho_th1dot_dt - dhoL_dho_th1
9 tau2 = d_dhoL_dho_th2dot_dt - dhoL_dho_th2

```

So, now, the similar way I can modify this theta 2 double dot as this form So, now, you can see that these all modified. So, this modification I have already done, and you can see, now, this time differentiation I already modified and if you see here, there is no function of time anywhere. So, now, if you look at it even till the end, so, there is no time dependent variable. So, now, what you can do, you can take tau 1 and tau 2 in this form.

(Refer Slide Time: 13:16)

```

MATLAB R2020b - academic use
HOME PLOTS APPS
Command Window
New to MATLAB? See resources for Getting Started.
Z^d1^d2^m2^th1dot^th2dot^sin(th2)

tau2 =

a2^2*m2*th1ddot + a2^2*m2*th2ddot + a2*g*m1*cos(th1 + th2) +
a2*m2*sin(th1 + th2)*(a1*th1dot*cos(th1) + a2*cos(th1 +
th2)*(th1dot + th2dot))*(th1dot + th2dot) - a2*m2*cos(th1 +
th2)*(a1*th1dot*sin(th1) + a2*sin(th1 + th2)*(th1dot + th2dot))
- a1*a2*m2*th1dot*th2dot*sin(th2)

fx >> simplify(tau1

```

```

>>
>>
>>
>>
>> simplify(tau2)

ans =

a2*(a1*m2*sin(th2)*th1dot^2 + a2*m2*th1ddot + a2*m2*th2ddot
+ g*m1*cos(th1 + th2) + a1*m2*th1ddot*cos(th2))
  
```

```

ans =

a2*(a1*m2*sin(th2)*th1dot^2 + a2*m2*th1ddot + a2*m2*th2ddot
+ g*m1*cos(th1 + th2) + a1*m2*th1ddot*cos(th2))

>>
>>
>>
>>
>>
>>
  
```

So, now, if I actually run this, so, if I run this, so, what I can actually get, whatever I have seen in the earlier case, the tau 1 and tau 2 is obtained the same form. So, only thing it is not simplified. So, if I say simplify of so tau 1, so, you can see what you have obtained in the previous lecture the same thing is coming. For example, you want to see what is the centripetal term? Centripetal term  $l_1 l_2 m_2 \sin \theta_2$  which is a minus sign the same thing is coming.

So, instead of  $l_1 l_2$  we have taken  $a_1$  and  $a_2$  similarly, the Coriolis component, so, 2 times of  $a_1 a_2 \cos \theta_2$  into  $m_2$ . So, that is also coming. So, similar way you can find, so, every other thing. So, even you can simplify tau 2. So, you can see this is also like a coming as per that. So,

here you can see like only the centripetal term would be coming that is you can see, which is  $a^2 \sin^2 \theta$  that is there.

And you can see the gravity term also coming, so, like that, you obtain. In the sense what you can see, so, the same result both formulations are giving but the formulations if you talk about computational perspective, even I am not talking about numerical computation, I am saying even you are deriving the equation of motion in symbolic form, I felt, personally I felt that Newton Euler is giving much more flexible, in addition to that, that can be even give all 3 axes forces and 3 axes access moment which can be used for structural dynamics as well.

Further even you can go out to the zeroth joint or zeroth frame then you can even transfer what is the; you can call shaking force and moment. So, based on that like your end effector is taking this much load, your base is supposed to be stronger this way for example, you are putting into a mobile base. So, your end effector is lifting something, your support reaction is not enough what happen? The vehicle will topple.

So, in that sense if you calculate this end effector to base forces the shaking forces and moment the support reaction is directly coming, the support reaction all the time is positive if the support reaction is giving a negative sign what in the sense your manipulator is losing the contact from the ground and similar way if the shaking forces and moments are known, you can decide what support you should have or what foundation you should have and whether we can mount it on the vehicle or mounted on the table or your foundation should be strong and other thing.

So, these are all ideas will give and Newton Euler even will you the axis or axial forces and moments. So, in that sense even you can do the structural aspects, you can go for simple dynamic design study and then you can see what would be the cross section, better cross section for supporting this load at this mass or not.

And that idea will always better. So, in that sense, what we have seen in this particular lecture, so we have talked about how to derive the equation of motion with the help of Lagrangian Euler using MATLAB code that is what we have seen. So, I hope in the next class we will see the numerical simulation of a dynamic model and then we will close this dynamic part and then we will move to the controller and trajectory generation. So, with that, see you then bye, take care.