

**Wheeled Mobile Robots**  
**Prof. Santhakumar Mohan**  
**Department of Mechanical Engineering**  
**Indian Institute of Technology, Palakkad**

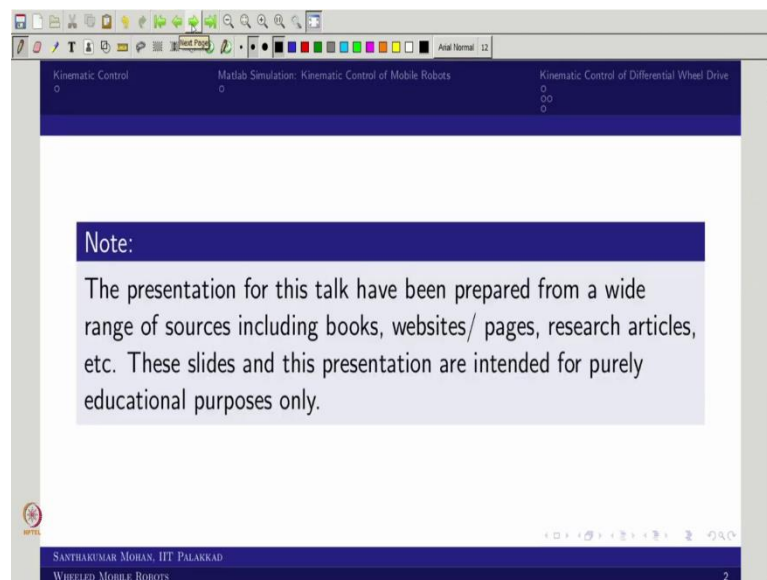
**Lecture - 36**

**Simulation of Land-based Mobile Robots along with Kinematic Control Part 1**

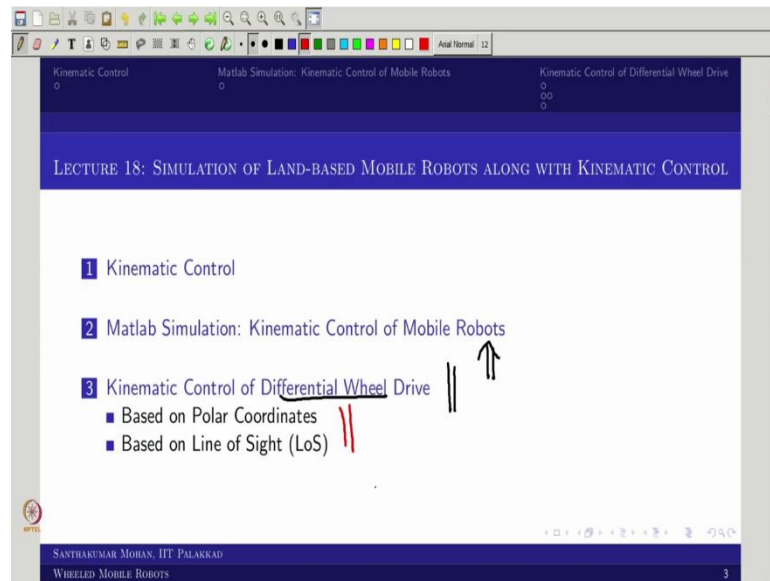
Welcome back to Wheeled Mobile Robots. So, last class or last lecture what we have seen is actually like we introduce the kinematic control. So, with the help of first order error dynamics, where we consider that the error is actually like asymptotically stable when  $t$  tends to infinity, in the sense when  $t$  tends to infinity how the error goes to 0. That what we have considered which is what we call first order error dynamics.

So, based on that we derive the; you call kinematic control law. And, then what we have seen like this may be applicable to several system, and how that would be actually like initially we do a simulation part. And, then we will actually like see how that would not applicable for certain system, which we call non-holonomic system.

(Refer Slide Time: 01:04)



(Refer Slide Time: 01:06)



So, that is what we are trying to cover in this particular lecture. This particular lecture would be talking about you can say MATLAB simulation of kinematic control of mobile robot. So, here we would be seeing a few example. So, first we will talk about general land base.

So, after that we will see like a few of the what you call holonomic robot, then will bring back to the non-holonomic in that one specific particular case, what you call you can say differential drive; in that what we can actually like do, and then what are the two method we can apply.

(Refer Slide Time: 01:32)

**Robot Kinematic (Motion) Control**

- **Desired:**
  - Desired positions,  $\eta_d(t)$  ✓
  - Desired velocities,  $\dot{\eta}_d(t)$ , for set-point control:  $\dot{\eta}_d(t) = 0$
- **Available:**
  - Actual positions,  $\eta(t)$
  - Jacobian matrix,  $J(\eta), J^{-1}(\eta)$
- **To find:**
  - Control inputs,  $\xi$
- **Objective:**
  - Asymptotically (exponentially) stable,  $t \rightarrow \infty, \eta \rightarrow \eta_d$  (or)
  - in other words,  $t \rightarrow \infty, \tilde{\eta} \rightarrow 0 \Rightarrow \tilde{\eta}(t) = e^{-\lambda t}, \lambda > 0$
  - where  $\tilde{\eta} = \eta_d - \eta$

SANTANU KUMAR MOHAN, IIT PALAKKAD  
WHEELED MOBILE ROBOTS 4

So, in that sense we will first go to the MATLAB side. So, you know this is what we have taken. So, what we have taken? We have taken that the desired is given, where you have a  $\eta_d$  desired and  $\dot{\eta}_d$  desired dot would be given the same thing, if it is a set point control this  $\dot{\eta}_d = 0$ .

So, that is what we have actually like taken consideration and we assume that the sensors are available to give all actual positions which include the orientation. And, for the kinematic relation we assume that the Jacobian matrix are available. We are trying to find out what is  $\xi$ . The  $\xi$  we found based on simple first order error dynamics.

(Refer Slide Time: 02:11)

Kinematic Control of Mobile Robots

Choosing the control input based on the computed velocity control, as follows:

$$\xi = J^{-1}(\eta) [\dot{\eta}_d(t) + \lambda \tilde{\eta}(t)] \quad (1)$$

Based on this control input vector, we can find the individual wheel velocities of the mobile robot, as follows:

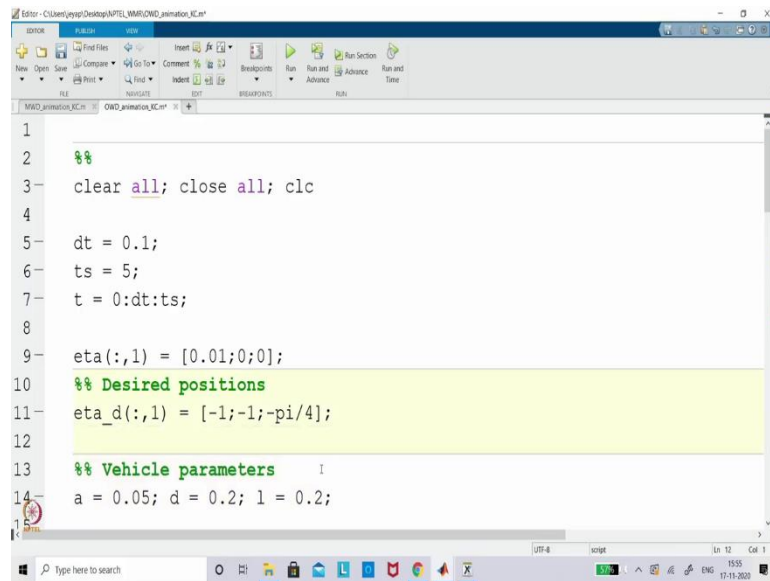
$$\omega = W^+ \xi \quad (2)$$

SANTHAKUMAR MOHAN, IIT PALAKKAD  
WHEELED MOBILE ROBOTS 5

So, this is what we have actually like used. So, for that what we are trying to use this particular equation what you call computed velocity control. So, now we will first do this in what you call in a MATLAB base. So, after that we can see like the same thing you can actually like extend.

So, this  $\xi$  can be written as  $\omega$ , in the form of  $\omega$  where this  $W^+$  what you call pseudo inverse. If it is actually like 3 by 3 then this would be a inverse. So, this also we will try to adapt and see how the situation goes. So, now, we will actually like move back to MATLAB window, where we ended in the last class. And, then we will see how we can apply this particular control scheme and see how further changes can be done, ok.

(Refer Slide Time: 02:53)

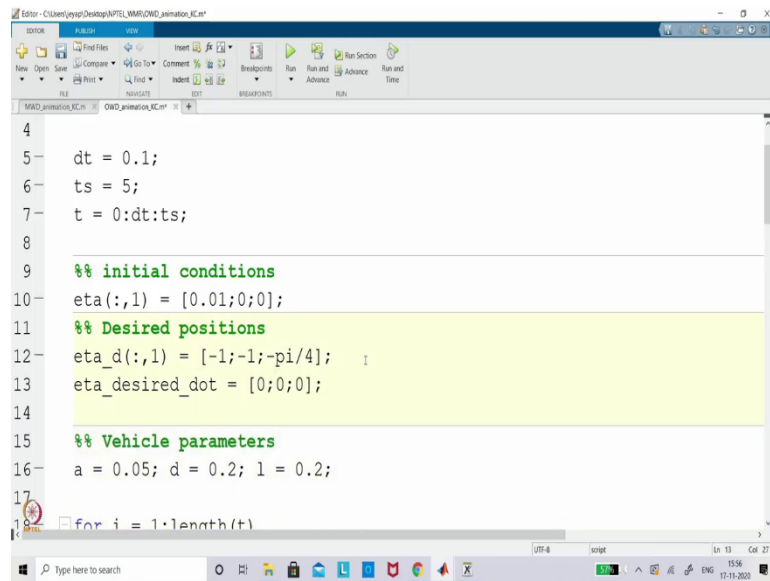


```
1
2
3 clear all; close all; clc
4
5 dt = 0.1;
6 ts = 5;
7 t = 0:dt:ts;
8
9 eta(:,1) = [0.01;0;0];
10 %% Desired positions
11 eta_d(:,1) = [-1;-1;-pi/4];
12
13 %% Vehicle parameters
14 a = 0.05; d = 0.2; l = 0.2;
```

So, now we can actually like see that the same aspect what we have done that we are actually like brought it. So, only thing I have actually like took the code which is very simple. So, you can see like these are your simulation parameter and this is what your initial condition ok.

Then, I am actually like giving the desired as a constant. So, Desired you call positions. So, this is what you call generalized coordinate. So, in addition to that you have actually like Vehicle parameters. So, in this case the vehicle parameter is length, then you call width and as well as the wheel radius.

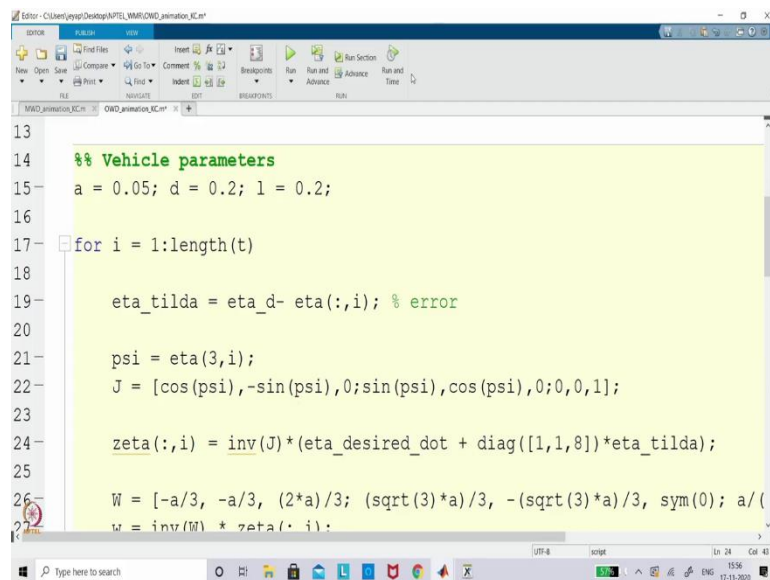
(Refer Slide Time: 03:32)



```
4
5 dt = 0.1;
6 ts = 5;
7 t = 0:dt:ts;
8
9 %% initial conditions
10 eta(:,1) = [0.01;0;0];
11 %% Desired positions
12 eta_d(:,1) = [-1;-1;-pi/4];
13 eta_desired_dot = [0;0;0];
14
15 %% Vehicle parameters
16 a = 0.05; d = 0.2; l = 0.2;
17
18 for i = 1:length(t)
```

So, these three we have considered. So, this I will actually like write it just for your initial understanding ok.

(Refer Slide Time: 03:44)



```
13
14 %% Vehicle parameters
15 a = 0.05; d = 0.2; l = 0.2;
16
17 for i = 1:length(t)
18
19     eta_tilda = eta_d - eta(:,i); % error
20
21     psi = eta(3,i);
22     J = [cos(psi), -sin(psi), 0; sin(psi), cos(psi), 0; 0, 0, 1];
23
24     zeta(:,i) = inv(J)*(eta_desired_dot + diag([1,1,8])*eta_tilda);
25
26     W = [-a/3, -a/3, (2*a)/3; (sqrt(3)*a)/3, -(sqrt(3)*a)/3, sym(0); a/(
27     w = inv(W) * zeta(:,i);
```

So, now what we are trying to do? So, we are trying to find out  $\eta$  tilde so, which is what we have written as  $\eta_d - \eta$ . So, now, we can actually like calculate this is what you call error ok. So, this error supposed to goes to 0, when  $t \rightarrow \infty$  that is what the whole idea. So, for that we are actually like taking the Jacobian matrix.

So, now you know why this Jacobian matrix is required so, you can recall here. So, you can see like this  $J^{-1}$  is coming in this case the J matrix is actually like orthogonal matrix. So, the  $J^{-1}$  is just a transpose, but still you can actually like see this J purpose is required.

So, that is why we defined  $\Psi$  and J. So, now, you can see like what we have applied. So, that particular control we have used. So, only thing I did not added the  $\eta$  you can say desired dot. So, that is actually like in this case it is 0. So, I am just putting it that. So,  $\dot{\eta}_d$  is actually like a 0 vector.

(Refer Slide Time: 04:50)

```

22-     psi = eta(3,i);
23-     J = [cos(psi), -sin(psi), 0; sin(psi), cos(psi), 0; 0, 0, 1];
24-
25-     zeta(:,i) = inv(J)*(eta_desired_dot + diag([1,1,8])*eta_tilda);
26-
27-     W = [-a/3, -a/3, (2*a)/3; (sqrt(3)*a)/3, -(sqrt(3)*a)/3, sym(0)]; a/(
28-     w = inv(W) * zeta(:,i);
29-     zeta(:,i) = W*w;
30-     eta(:,i+1) = eta(:,i) + (1-exp(-1*t(i))) * J*zeta(:,i) * dt;
31- end
32-
33-
34- veh_box = [0.4*cosd(0:360); 0.4*sind(0:360)];
35- wheel_b = 0.5*[-0.2 0.2 0.2 -0.2 -0.2;-0.05 -0.05 0.05 0.05 -0.05];

```

So, now if I apply; what I will get? So, I will get  $\xi$ . So, now that  $\xi$  I can actually like; you can actually like substitute here and then you can actually like integrate. Further you can see like last class we did not see this, but suddenly there is something is appearing here right. So, what this? This is actually like for considering the actuator dynamics.

I am assuming that the actuator is actually like a simple first order system, where the what you call time constant is actually like 1 in the sense, the delayed time of this particular actuator would be 1 second. So, that way you can actually like get this particular what you call exponential model.

So, this can be rewritten as your actuated dynamics. So, otherwise this is what we have used in the last you can say lectures and all. So, now, I am adding the actuated dynamics

included and if I assume that this is a Omni wheel drive, then this will come, right now, I am actually like ignoring that as a Omni wheel drive ok.

(Refer Slide Time: 05:42)

```

Editor - C:\Users\yyp\Desktop\MPTEL\MWR\OWD_animation_KC.m
28 % w = inv(W) * zeta(:,i);
29 % zeta(:,i) = W*w;
30 % eta(:,i+1) = eta(:,i) + (1-exp(-1*t(i))) * J*zeta(:,i)*dt;
31 % w = inv(W) * zeta(:,i);
32 % zeta(:,i) = W*w;
33 % eta(:,i+1) = eta(:,i) + (1-exp(-1*t(i))) * J*zeta(:,i)*dt;
34 end
35
36
37 % veh_box = [0.4*cosd(0:360);0.4*sind(0:360)];
38 % wheel_b = 0.5*[-0.2 0.2 0.2 -0.2 -0.2;-0.05 -0.05 0.05 0.05 -0.05;];
39 %
40 % for i = 1:length(t)
41 %     psi = eta(3,i);
42 %     x(i) = eta(1,i); y(i) = eta(2,i);

```

So, now in that case what I am trying to do? I am just trying to show as a plot. So, where  $t$  and  $\eta$  you can say desired that  $\eta_d$  would be a simple step, ok. So, now, this is what you are actually like trying to plot, where  $\eta_1$  and  $\eta_2$  where  $x$  and  $y$  so, that if we are trying to plot. So, even I am adding all 3.

(Refer Slide Time: 06:18)

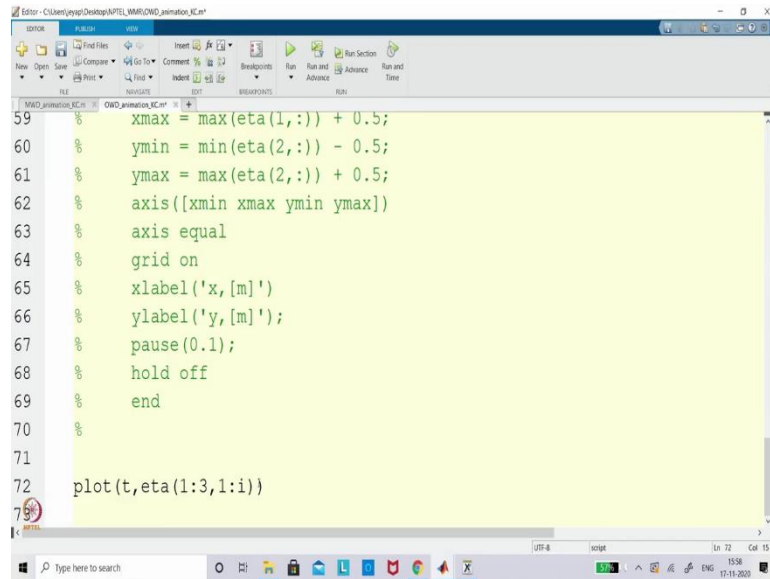
```

Editor - C:\Users\yyp\Desktop\MPTEL\MWR\OWD_animation_KC.m
43 % R2 = [cos(psi+5*pi/6), -sin(psi+5*pi/6); sin(psi+5*pi/6), cos(psi+5*pi/6)];
44 % R3 = [cos(psi+3*pi/2), -sin(psi+3*pi/2); sin(psi+3*pi/2), cos(psi+3*pi/2)];
45 % w_m3 = R3*(wheel_b+[1;0]);
46 % w_m1 = R1*(wheel_b+[1;0]);
47 % w_m2 = R2*(wheel_b+[1;0]);
48 % fill(v_m(1,:) + x(i), v_m(2,:) + y(i), 'y');
49 % hold on
50 % fill(w_m1(1,:) + x(i), w_m1(2,:) + y(i), 'k');
51 % fill(w_m2(1,:) + x(i), w_m2(2,:) + y(i), 'k');
52 % fill(w_m3(1,:) + x(i), w_m3(2,:) + y(i), 'k');
53 %
54 % plot(eta_d(1), eta_d(2), 'k')
55 % plot([eta_d(1), eta_d(1)+0.2*cos(eta_d(3))], [eta_d(2), eta_d(2)+0.2*cos(eta_d(3))], 'b-')
56 % plot(eta(1,1:i), eta(2,1:i), 'b-')
57 % plot(eta_d(1)+0.1*cosd(0:360), eta_d(2)+0.1*sind(0:360), 'c--')

```



(Refer Slide Time: 06:20)



```
59 %   xmax = max(eta(1,:)) + 0.5;
60 %   ymin = min(eta(2,:)) - 0.5;
61 %   ymax = max(eta(2,:)) + 0.5;
62 %   axis([xmin xmax ymin ymax])
63 %   axis equal
64 %   grid on
65 %   xlabel('x, [m]')
66 %   ylabel('y, [m]');
67 %   pause(0.1);
68 %   hold off
69 %   end
70
71
72 plot(t,eta(1:3,1:i))
73
```

So, all 3 up to you call the 1 is to i in the sense 0 to t equal to in this case it is 5 second. So, that I am actually like trying to run. So, if you want actually like you want animation also you can bring it that animation from the previous code and then you can do it, but here the interest is actually like if you are taking the  $\xi$  in the form of what you call computed, you can say velocity control how this case.

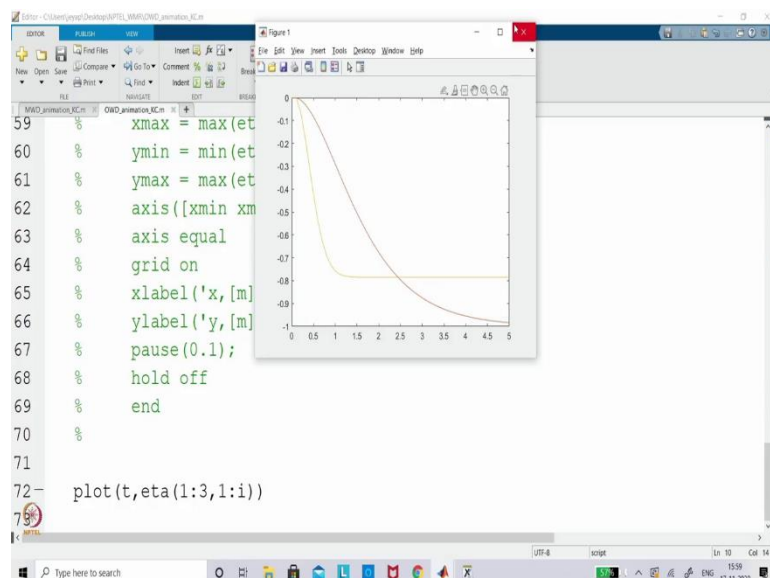
So, now you can see that this is what you have considered as in this case called  $\lambda$ . So, this  $\lambda$  is actually like diagonal matrix in our case. So, that is what we are actually taking the diagonal matrix here is actually like 1 and 1 in x and y axis and 8 only in the  $\Psi$  axis. So, it can be even considered as 1, but I will just took it because the  $\Psi$  would be required more control activity. Because of you can say non-zero initial error, but right now we have actually like considered the 0 initial error, ok.

(Refer Slide Time: 07:07)

```
Editor - C:\Users\jyng\Desktop\MPEL\MAR\OVD_animation_EC.m
5 dt = 0.1;
6 ts = 5;
7 t = 0:dt:ts;
8
9 %% initial conditions
10 eta(:,1) = [0;0;0];
11 %% Desired positions
12 eta_d(:,1) = [-1;-0.5;-pi/4];
13 eta_desired_dot = [0;0;0];
14
15 %% Vehicle parameters
16 a = 0.05; d = 0.2; l = 0.2;
17
18 for i = 1:length(t)
19
```

So, now I am actually like giving all the 3 are actually like starting from 0 ok. And I am actually like asking the manipulator or you can say not manipulator the mobile robot to move so, - 1 in both x and y and as well as -  $\pi$  by 4 in the sense  $-45^\circ$ . So, we will see with vehicle configuration; right now we will actually like try to run this. So, then this particular plot will say something, ok.

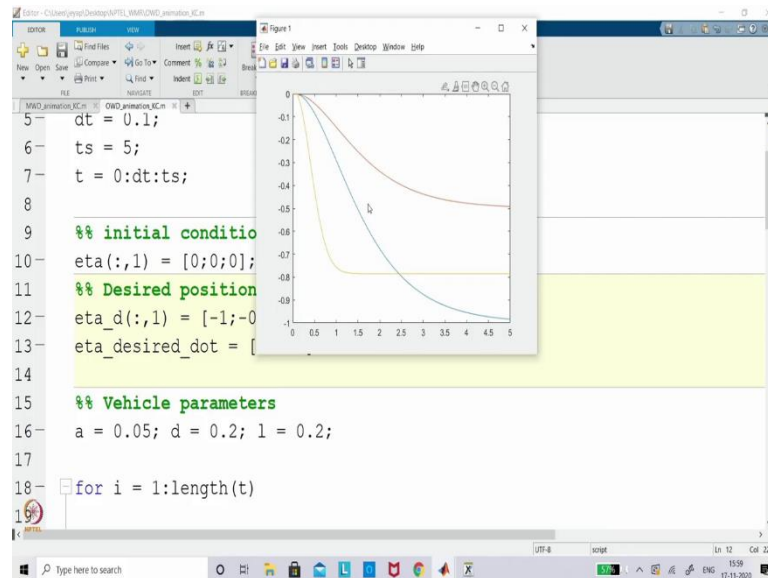
(Refer Slide Time: 07:33)



So, that is what we are seen; you can see like these two other curve x and y directly go to 1 and 1 in the sense - direction and this is going to pi by you can say 4. So, you know

like the meaningful line diagram, you can convert it into what you call animated model that we will bring it later. So, right now you can see it is overlapping. So, I am just to showing that these are actually like different.

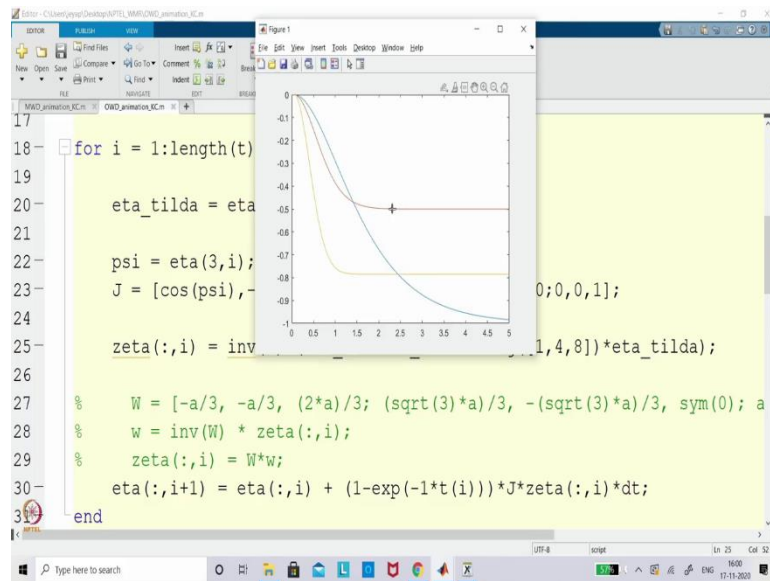
(Refer Slide Time: 07:57)



So, you can actually like see. So, one is actually like end with you can say what you call  $\frac{\pi}{4}$ . The other one is actually like going one is actually like trying to reach - 1. So, the other one is trying to reach - 0.5. So, now, you can see the difference I put into one different value in the  $\lambda$ , you see the  $\lambda$  if I keep on increasing what it is trying to do it is trying to reduce the what you call the rise time.

If you recall your control course so, you know what is rise time. So, it is actually like increasing the you can say you can say performance by reducing the overall rise time.

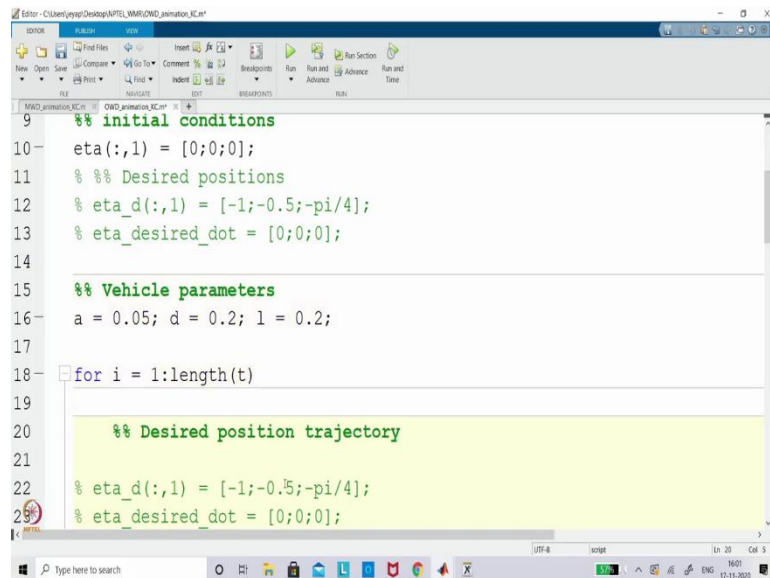
(Refer Slide Time: 08:35)



So, now in order to verify that you can see I am actually like varying this as probably 4, now you can actually like see that it is actually like reaching probably around 2.5 second as 0.5; whereas the x axis position is actually taking long time, but y axis is reached in 2.5 second. And where the  $\Psi$  is reached almost like 1.25 second, right.

So, now, you can see like what is the value of  $\lambda$  that how it play if keep on increasing the  $\lambda$  the output will reach faster to the desired. So, now what we have done here is actually like a given a step right. So, I want to actually like follow probably time dependent profile. What that mean I am actually taking a sinusoidal profile for x and y and I assume that the vehicle should not have any orientation.

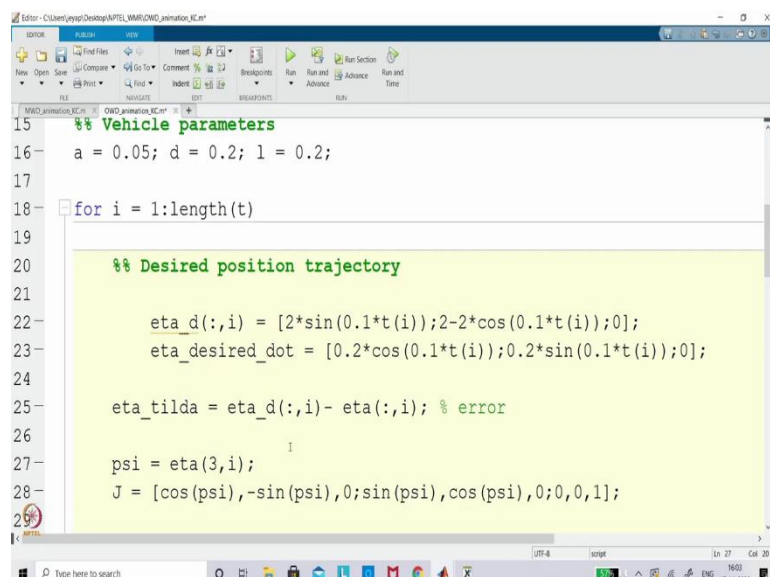
(Refer Slide Time: 09:22)



```
9 %% initial conditions
10 eta(:,1) = [0;0;0];
11 %% Desired positions
12 eta_d(:,1) = [-1;-0.5;-pi/4];
13 eta_desired_dot = [0;0;0];
14
15 %% Vehicle parameters
16 a = 0.05; d = 0.2; l = 0.2;
17
18 for i = 1:length(t)
19
20     %% Desired position trajectory
21
22     eta_d(:,1) = [-1;-0.5;-pi/4];
23     eta_desired_dot = [0;0;0];
```

So, in that sense what happen? The desired is actually like; I will actually like make it as what you call, so, new thing as a desired trajectory not it is. So, I will write Desired position trajectory. So, now, this is what I am actually trying to use. So, in the sense what I am actually like going to give this I am giving to as a profile. So, in the sense what I am trying to use it, I said already I am trying to use a sinusoidal curve.

(Refer Slide Time: 09:51)



```
15 %% Vehicle parameters
16 a = 0.05; d = 0.2; l = 0.2;
17
18 for i = 1:length(t)
19
20     %% Desired position trajectory
21
22     eta_d(:,i) = [2*sin(0.1*t(i));2-2*cos(0.1*t(i));0];
23     eta_desired_dot = [0.2*cos(0.1*t(i));0.2*sin(0.1*t(i));0];
24
25     eta_tilda = eta_d(:,i)- eta(:,i); % error
26
27     psi = eta(3,i);
28     J = [cos(psi), -sin(psi), 0; sin(psi), cos(psi), 0; 0, 0, 1];
```

So, now this I am giving as a two you can say sin, it is actually like with the  $0.1 \omega$  case. So, the other one is actually like I am saying that this is simple what you call sinusoidal,

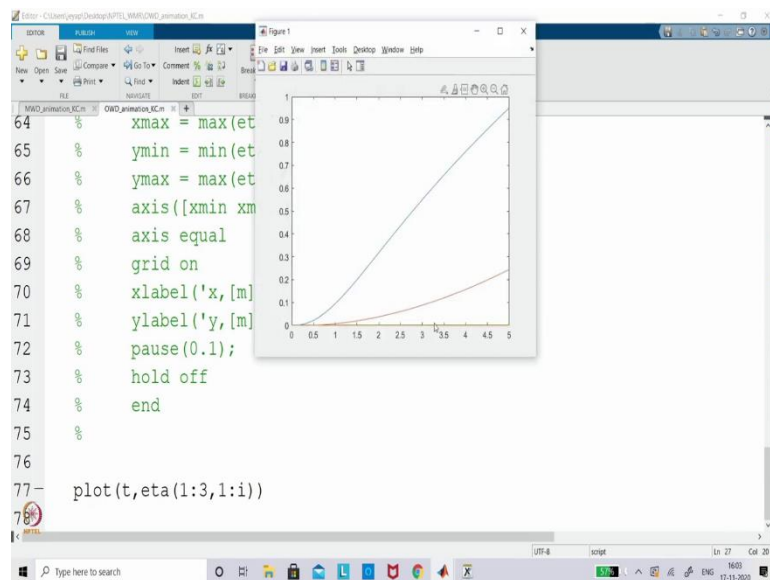
but it is something like I am trying to or draw a circle. The circle will have x and y coordinate is sin and cos or cos and sin either way you can do it.

So, I am actually like taking it this y is actually like a cos of this. So, now, I am actually like taking it this. So, later on I will show you how you can actually like even modify. So, now, i assume that the vehicle position supposed to be orientation supposed to be 0, but the vehicle position supposed to move in a x y plane as a circle.

So, now then the  $\dot{\eta}_d$  again actually like what happened, this would be what you call it would be a non-zero value. So, that we can actually like get it from differentiating this so, in the sense this would be  $0.1 \times t(i)$  and this would be again  $0.2 \times \sin(0.1) \times t$ , ok.

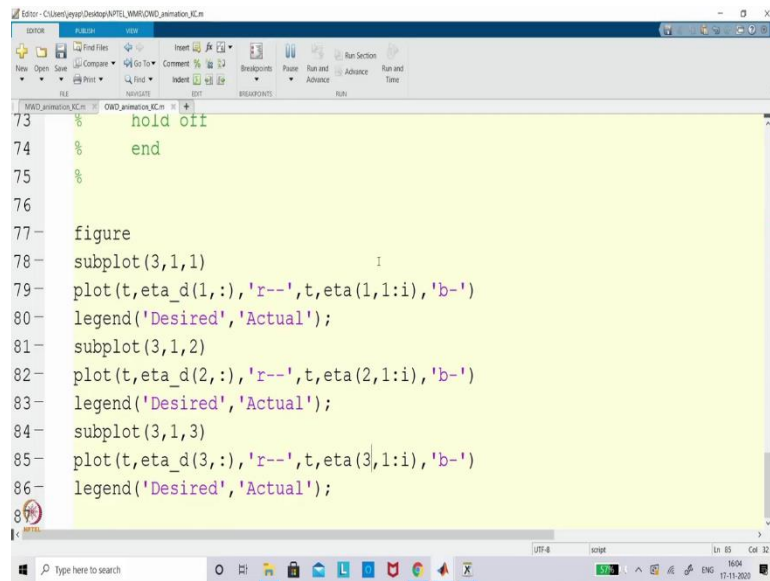
So, in that sense what you can see? You got the  $\dot{\eta}_d$  is non-zero, earlier we have taken set point where it is 0. Now, it is actually like non-zero, now  $\eta_d$  is actually like what you call it is no longer, you can say a single constant now it is a loop. So, that is why I have actually like changed.

(Refer Slide Time: 11:21)



So, now if I actually like ran what you can expect. So, this would be actually like following some profile right, we will see whether that is happening it is look like that, but, I will actually like try to plot in a more; you can say proper way.

(Refer Slide Time: 11:29)



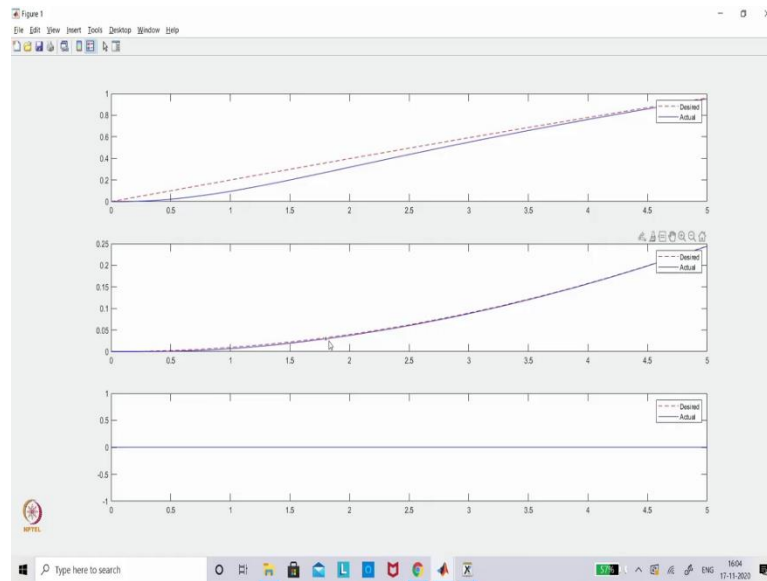
```
73 % hold off
74 % end
75 %
76
77 figure
78 subplot(3,1,1)
79 plot(t,eta_d(1,:), 'r--', t, eta(1,1:i), 'b-')
80 legend('Desired', 'Actual');
81 subplot(3,1,2)
82 plot(t,eta_d(2,:), 'r--', t, eta(2,1:i), 'b-')
83 legend('Desired', 'Actual');
84 subplot(3,1,3)
85 plot(t,eta_d(3,:), 'r--', t, eta(3,1:i), 'b-')
86 legend('Desired', 'Actual');
```

So, I am actually like putting this is figure. So, where I will just make it so, plot (t,  $\eta$ ) so,  $\eta_d$  so of 1 is 2 you can say this is actually like i. So, which I am writing in red color ok. So, there itself I am actually like plotting  $\eta$  of so, 1 so 1 is to i ok. So, this I am actually like plotting in a blue color. So, now, if I actually like plot this what happen? This would be a desired and this would be what you call the actual.

So, I am just putting a legend that way. So, I am writing that this is Desired and this is I am calling what you call Actual. So, I am just to showing x profile alone. So, the same way I can actually like make it. So, I am actually like making a subplot of so, 3 is to 1. So, there are 3 rows I am trying to make it.

So, I am making a 3 sub plots. So, 1 would be actually like this. So, second one would be y axis ok. So, on the third one would be you are what you call the angle, ok. So, now, I am actually like trying to plot this. So, we will see whether that is actually like following it or not.

(Refer Slide Time: 12:52)



So, you can see like an x axis position, it is actually like trying to maintain, because the control gain value we have given only 1. But whereas, it is 4 you can see that the steady state error or you can say the transient respond is actually like vastly going and here we have taken as 0, right. So, it is starting from 0 and end with 0.

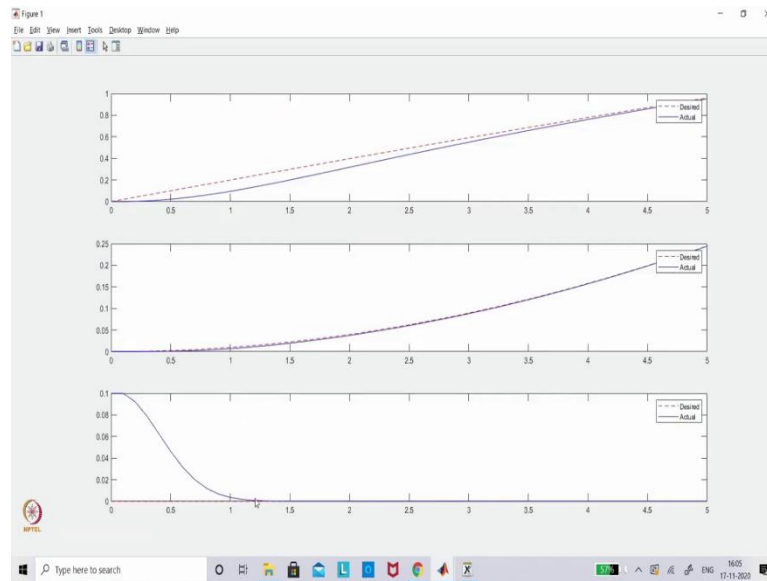
(Refer Slide Time: 13:16)

```
Editor - C:\Users\jyyp\Desktop\MTEL\MAR\OVD_animation_KC.m
7- t = 0:dt:ts;
8
9 %% initial conditions
10 eta(:,1) = [0;0;0.1];
11 %% Desired positions
12 % eta_d(:,1) = [-1;-0.5;-pi/4];
13 % eta_desired_dot = [0;0;0];
14
15 %% Vehicle parameters
16 a = 0.05; d = 0.2; l = 0.2;
17
18 for i = 1:length(t)
19
20 %% Desired position trajectory
```

So, in order to give small clarity it is starting from 0.1, ok. So, now, what happened? There is a non-zero initial orientation, but the vehicle supposed to orient according to the desired.



(Refer Slide Time: 13:25)



So, now you can see that it is actually like trying to vary. So, now, even you increase the  $x$  gain value all you can actually like do it.

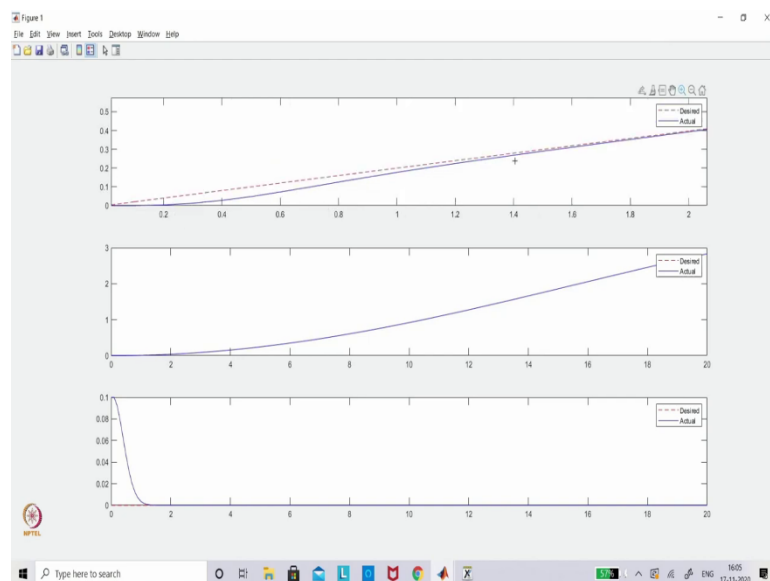
(Refer Slide Time: 13:35)

```
Editor - C:\Users\yasp\Desktop\MPTEL\MAR\OVD_animation_KC.m
EDITOR
File Edit View Insert Tools Desktop Window Help
Find Files Find in Files Find in Selected Files Find in Selected Files
New Open Save Print Compare Go To Comment Indent Unindent
File Edit View Insert Tools Desktop Window Help
MWD_animation_KC.m OVD_animation_KC.m
1
2 %%
3 clear all; close all; clc
4
5 dt = 0.1;
6 ts = 20;
7 t = 0:dt:ts;
8
9 %% initial conditions
10 eta(:,1) = [0;0;0.1];
11 %% Desired positions
12 eta_d(:,1) = [-1;-0.5;-pi/4];
13 %% eta_desired_dot = [0;0;0];
14
15 %% Vehicle parameters
16
```

(Refer Slide Time: 13:44)

```
Editor - C:\Users\jyng\Desktop\MPTEL\MAR\OVD_animation_EC.m
25     eta_tilda = eta_d(:,i) - eta(:,i); % error
26
27     psi = eta(3,i);
28     J = [cos(psi), -sin(psi), 0; sin(psi), cos(psi), 0; 0, 0, 1];
29
30     zeta(:,i) = inv(J)*(eta_desired_dot + diag([8,4,8])*eta_tilda);
31
32     % W = [-a/3, -a/3, (2*a)/3; (sqrt(3)*a)/3, -(sqrt(3)*a)/3, sym(0); a
33     % w = inv(W) * zeta(:,i);
34     % zeta(:,i) = W*w;
35     eta(:,i+1) = eta(:,i) + (1-exp(-1*t(i)))*J*zeta(:,i)*dt;
36 end
37
38
39 % veh box = [0.4*cosd(0.360); 0.4*sind(0.360)];
```

(Refer Slide Time: 13:49)



And similarly I can change the what you call the time scale also I can change. So, for example, I am actually like increasing the x gain value also probably 8, I am just showing you can actually like see that. Now, it would be giving a bigger profile, but you can see that the transient what you can see earlier. So, that was actually like changed right.

So, you can see it is actually like reached in almost like less than 2 second whereas, in the earlier case even up to 5 second it was decaying right. So, these all actually like you

can see. So, now what we can bring. So, I already said the  $\xi$  can be written in the form of  $W$  into  $\omega$ . So, where  $\omega$  I can find it with the help of  $W$ . So, for making that into a picture I am taking  $W$  in the form of you can say Omni wheel drive or I am taking 3 Omni wheel drive case and I am putting that  $W$  in the picture.

(Refer Slide Time: 14:29)

```

25     eta_tilda = eta_d(:,i) - eta(:,i); % error
26
27     psi = eta(3,i);
28     J = [cos(psi), -sin(psi), 0; sin(psi), cos(psi), 0; 0, 0, 1];
29
30     zeta(:,i) = inv(J)*(eta_desired_dot + diag([8,4,8])*eta_tilda);
31
32     W = [-a/3, -a/3, (2*a)/3; (sqrt(3)*a)/3, -(sqrt(3)*a)/3, sym(0)];
33     w = inv(W) * zeta(:,i);
34     zeta(:,i) = W*w;
35     eta(:,i+1) = eta(:,i) + (1-exp(-1*t(i)))*J*zeta(:,i)*dt;
36 end
37
38
39 % veh box = [0.4*cosd(0.360); 0.4*sind(0.360)];

```

So, in the sense what I am actually like bringing it, I am bringing the wheel configuration inside. And, now I am actually like calculating the wheel angular velocity. So, now, I am recalculating the  $\xi$  why it is so, because this wheel angular velocity may be having a restriction right now we did not give the what you call limit.

So, but you can actually like limit that we will bring that in the next scenario. Now, we have seen that the result in a line diagram basis, I already told that this is not that much informatable. Now, what we can see in the next lecture we will bring the animation part, and we will see how that would be beneficial for ease of understanding.

And along with that we will bring the wheel configuration, and we will see how the non-holonomic wheels, or you can say non-holonomic mobile robot will benefit out of this particular kinematic control. So, until then we will see.