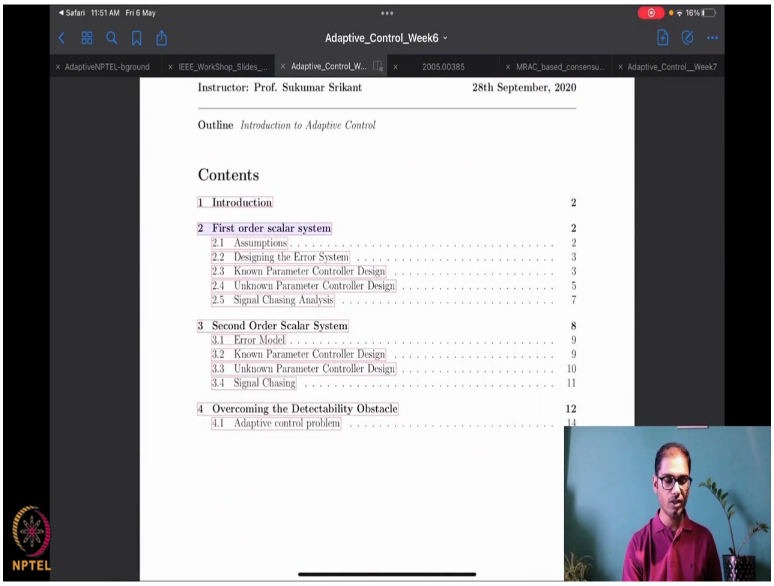
Nonlinear Adaptive Control
Professor Srikant Sukumar
Systems and Control
Indian Institute of Technology, Bombay
Week 7
Lecture No: 37
Backstepping in Adaptive Control: Introduction (Part-1)

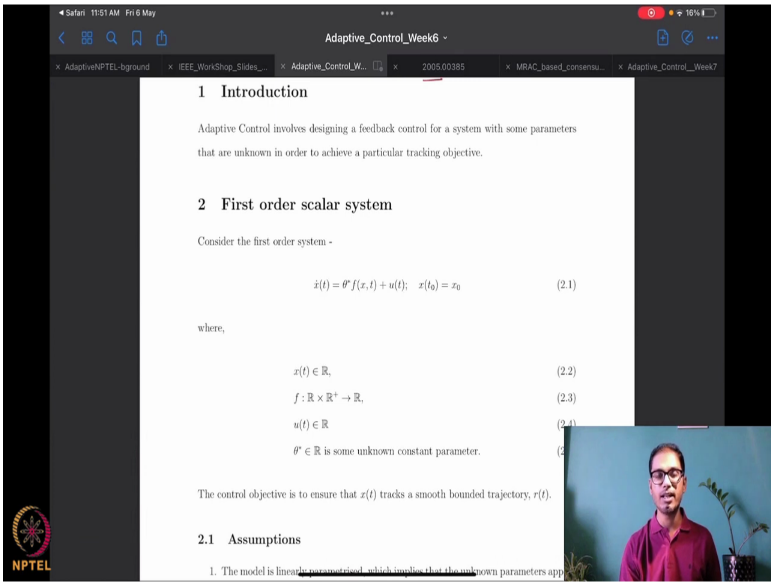
(Refer Slide Time: 00:16)

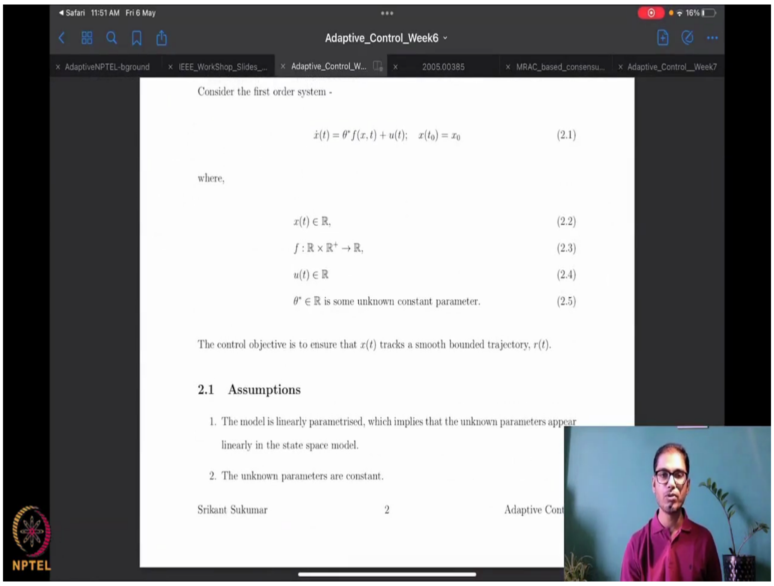
Hello, everyone. Welcome to yet another session of our NPTEL on Nonlinear and Adaptive Control. I am Srikant Sukumar from Systems and Control, IIT Bombay. So welcome to week 7 of this course on Nonlinear Adaptive Control.

And by week 6, we have already started to learn about designing algorithms that can potentially drive systems such as what we see in our background which is essentially a SpaceX satellite orbiting the Earth.

(Refer Slide Time: 01:06)







Consider the first order system -

$$\dot{x}(t) = \theta^* f(x,t) + u(t); \quad x(t_0) = x_0 \tag{2.1}$$

where,

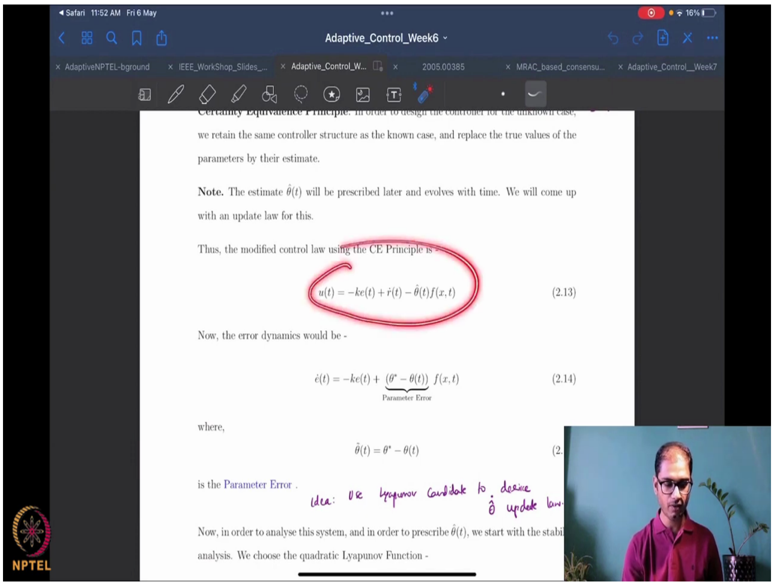
$$x(t) \in \mathbb{R}, \tag{2.2}$$
$$f : \mathbb{R} \times \mathbb{R}^+ \to \mathbb{R}, \tag{2.3}$$
$$u(t) \in \mathbb{R} \tag{2.4}$$
$$\theta^* \in \mathbb{R} \text{ is some unknown constant parameter.} \tag{2.5}$$

The control objective is to ensure that $x(t)$ tracks a smooth bounded trajectory, $r(t)$.

2.1 Assumptions

1. The model is linearly parametrised, which implies that the unknown parameters appear linearly in the state space model.
2. The unknown parameters are constant.

Srikant Sukumar 2 Adaptive Cont



we retain the same controller structure as the known case, and replace the true values of the parameters by their estimate.

Note. The estimate $\hat{\theta}(t)$ will be prescribed later and evolves with time. We will come up with an update law for this.

Thus, the modified control law using the CE Principle is -

$$u(t) = -ke(t) + \dot{r}(t) - \hat{\theta}(t)f(x,t) \tag{2.13}$$

Now, the error dynamics would be -

$$\dot{e}(t) = -ke(t) + \underbrace{(\theta^* - \theta(t))}_{\text{Parameter Error}} f(x,t) \tag{2.14}$$

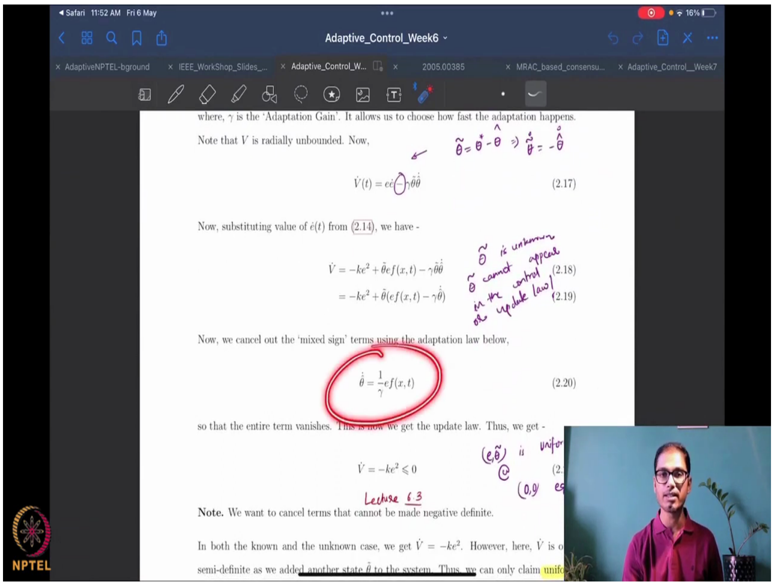
where,

$$\tilde{\theta}(t) = \theta^* - \theta(t)$$

is the Parameter Error .

Idea: Use Lyapunov Candidate to derive $\dot{\hat{\theta}}$ update law.

Now, in order to analyse this system, and in order to prescribe $\hat{\theta}(t)$, we start with the stabi analysis. We choose the quadratic Lyapunov Function -



where, γ is the 'Adaptation Gain'. It allows us to choose how fast the adaptation happens.

Note that V is radially unbounded. Now,

$\tilde{\theta} = \theta^* - \hat{\theta} \Rightarrow \dot{\tilde{\theta}} = -\dot{\hat{\theta}}$

$$\dot{V}(t) = e\dot{e} - \gamma\tilde{\theta}\dot{\hat{\theta}} \tag{2.17}$$

Now, substituting value of $\dot{e}(t)$ from (2.14), we have -

$$\dot{V} = -ke^2 + \tilde{\theta}ef(x,t) - \gamma\tilde{\theta}\dot{\hat{\theta}} \tag{2.18}$$
$$= -ke^2 + \tilde{\theta}(ef(x,t) - \gamma\dot{\hat{\theta}}) \tag{2.19}$$

$\tilde{\theta}$ is unknown
$\tilde{\theta}$ cannot appear in the control or the update law

Now, we cancel out the 'mixed sign' terms using the adaptation law below,

$$\dot{\hat{\theta}} = \frac{1}{\gamma}ef(x,t) \tag{2.20}$$

so that the entire term vanishes. This is how we get the update law. Thus, we get -

$$\dot{V} = -ke^2 \leqslant 0$$

$(e,\tilde{\theta})$ is unifo @ $(0,0)$ eq

Lecture 6.3

Note. We want to cancel terms that cannot be made negative definite.

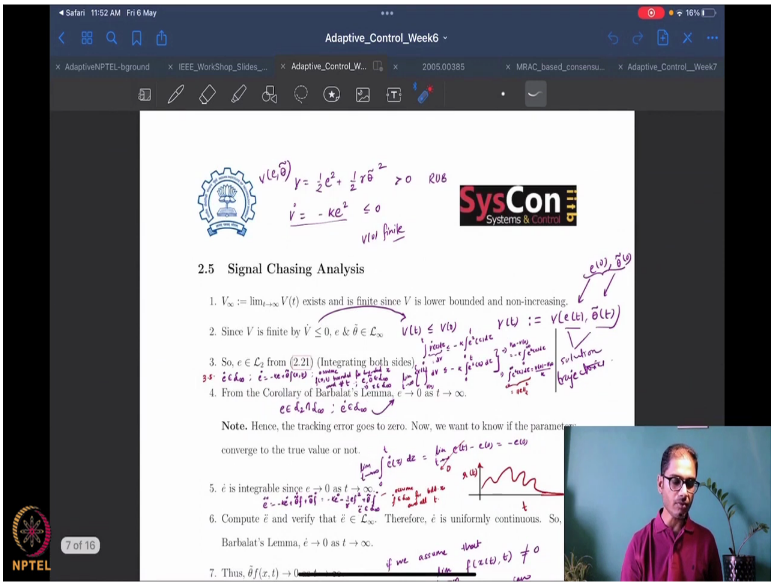
In both the known and the unknown case, we get $\dot{V} = -ke^2$. However, here, $\dot{V}$ is o semi-definite as we added another state $\tilde{\theta}$ to the system. Thus, we can only claim unifo

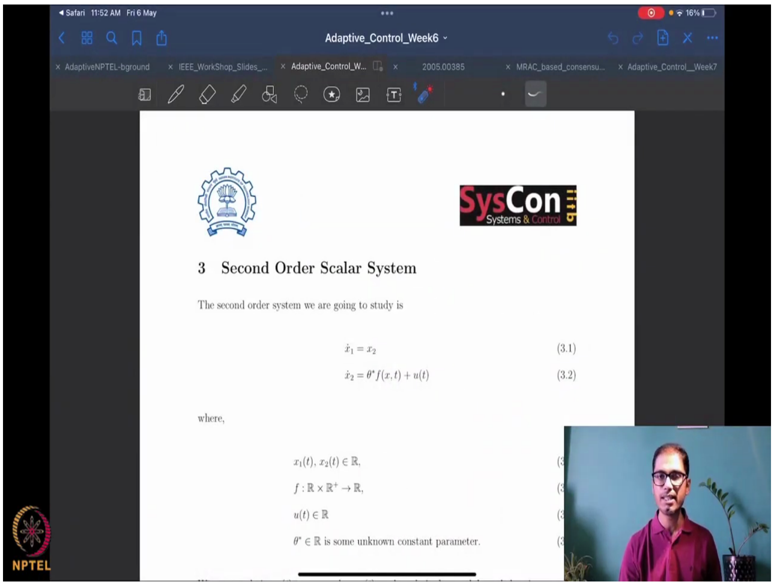
So to summarize a little bit of what we did in week 6 was that we started with a first-order scalar system. So a very, very simple sort of system so we started with the first-order scalar system and we designed an adaptive controller for the system.

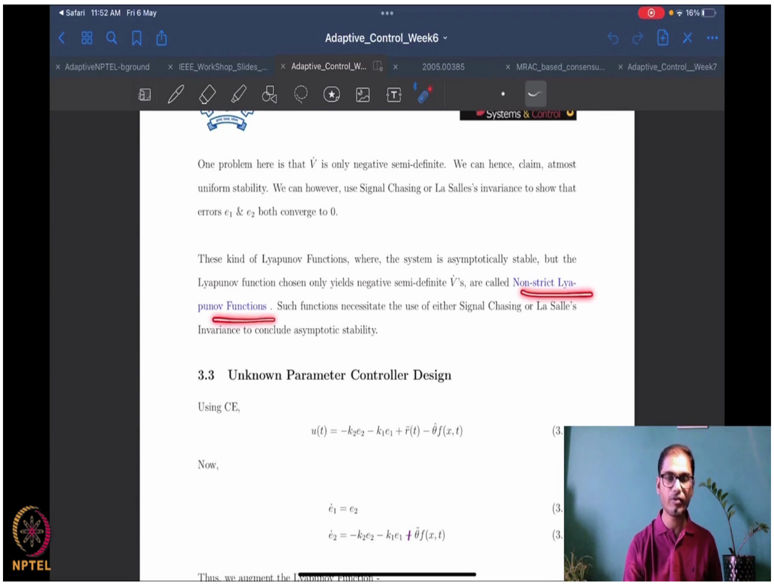
So the way we worked through it is that we started by designing a controller for the known case and then we use the certainty equivalence principle in order to design the unknown case adaptive controller.

So the adaptive controller essentially consists of a parameter estimator which feeds into the normal control signal. So this is what we did for the first order case. So we had this kind of controller and then we of course had a parameter estimator.

(Refer Slide Time: 01:58)

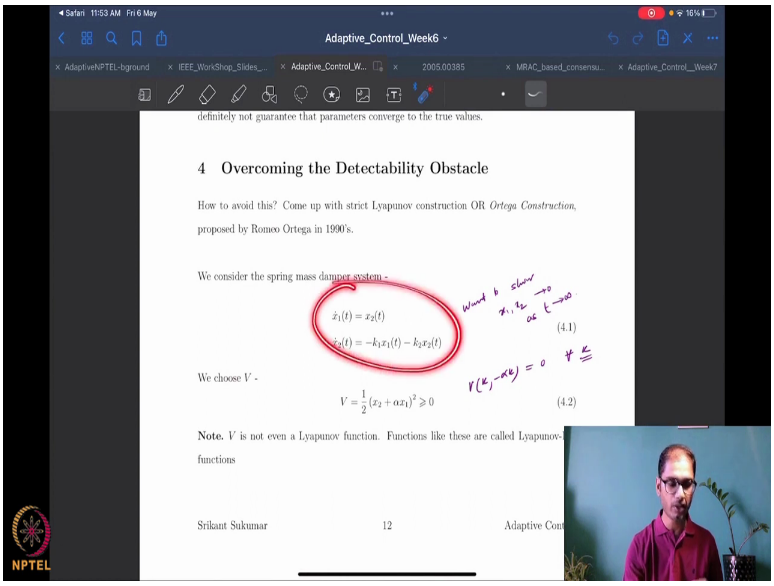


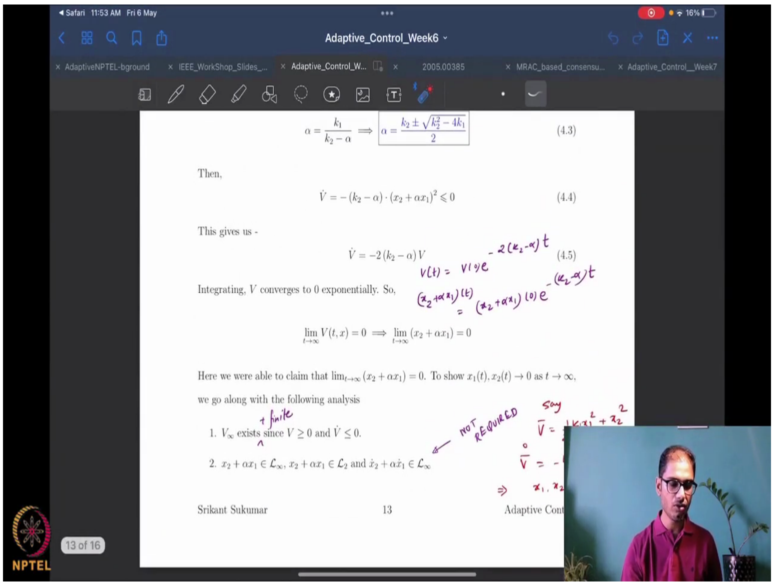


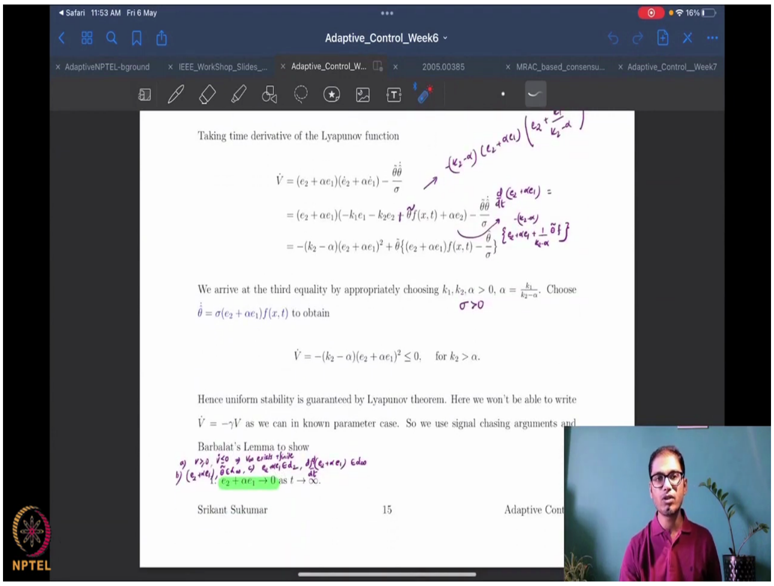


Now, further that of course we use signal chasing and so on to prove that everything works out fine. Further that we started to look at the second order system, and of course, when we used like, like a very, very standard or basic choice of Lyapunov candidate function as you would, we realize that we end up with a detectability obstacle. So this is because of the fact that we chose a non-strict Lyapunov function.

(Refer Slide Time: 02:31)



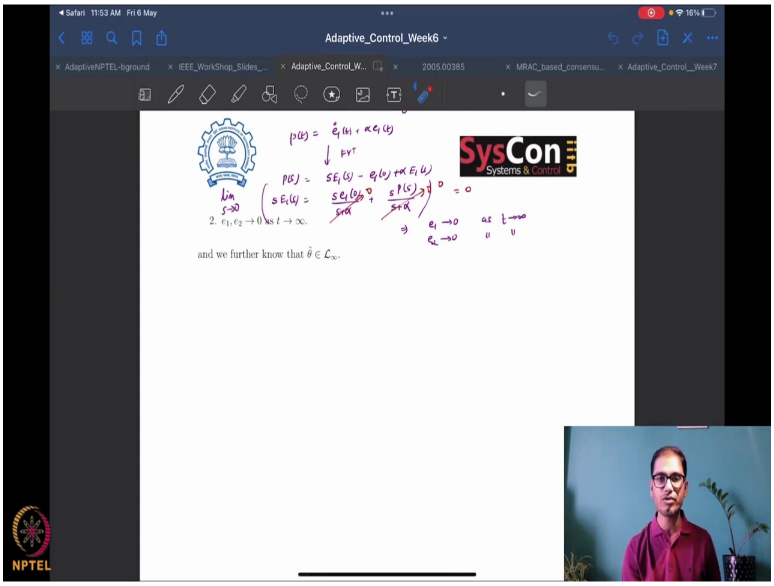




And so, what we try to do after that is essentially use something like an Ortega construction in order to overcome this detectability obstacle for second-order scalar systems. And that is what we did in this section 4.

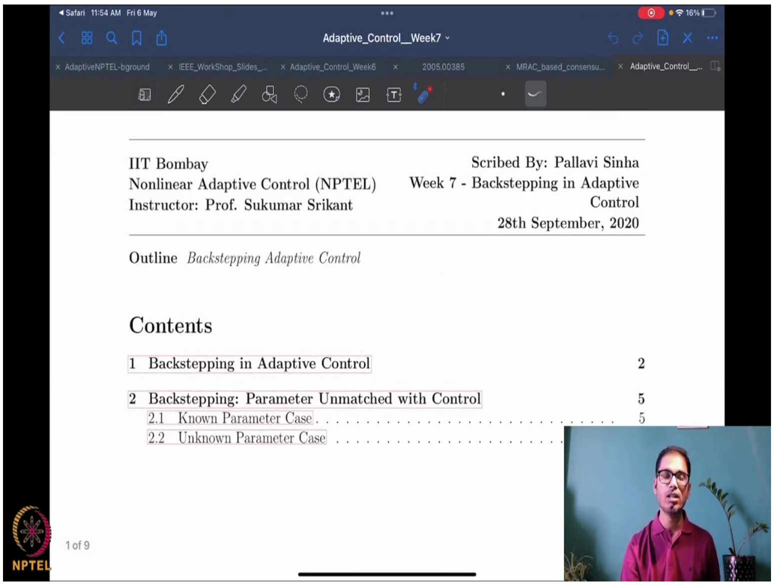
We first showed how to use this for the like, how this construction works for the general stability proof for a system like 4.1 which is like a spring mass damper system. And then we used it for adaptive control problem in this section and we showed that with this sort of construction which is not even a Lyapunov function, in fact, it is what we like to call a Lyapunov-like function, we could show that both the states that is the e1 and e2 converge to the origin as we desire.

(Refer Slide Time: 03:27)



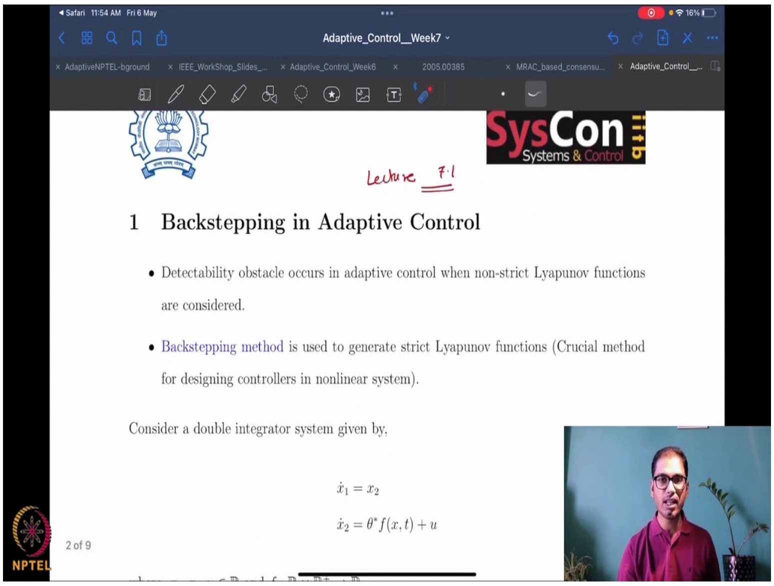
So, so, this is where we stopped.

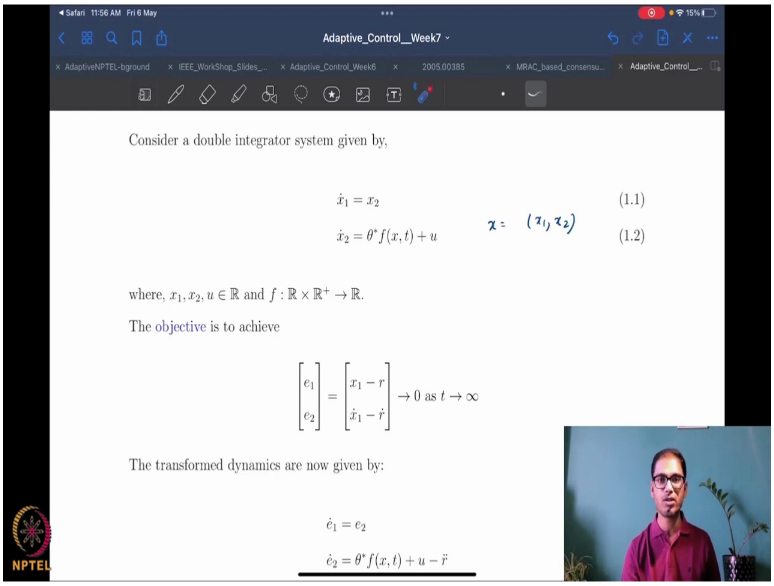
(Refer Slide Time: 03:35)

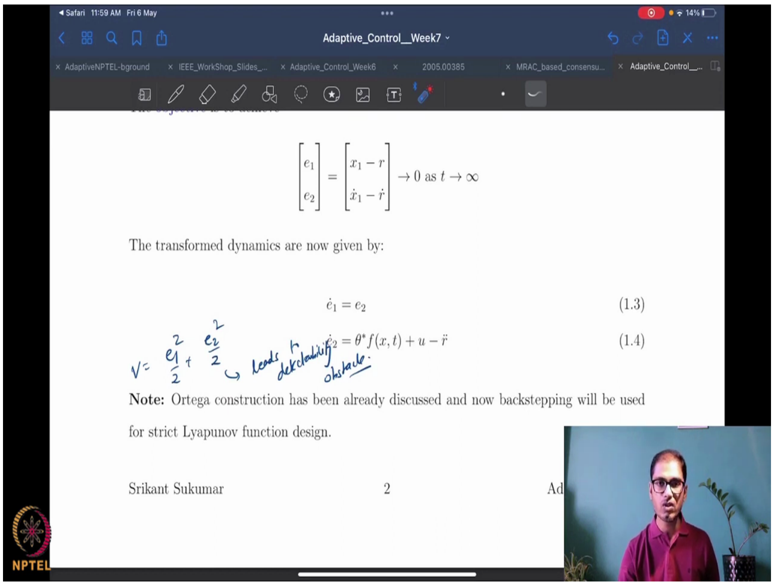


Now, what we, where we start off today is with the notion of backstepping in adaptive control. Now backstepping is a very very well-known and classical, by now classical method in nonlinear control. And in recent years it has also gained a lot of popularity in the adaptive control community also.

(Refer Slide Time: 03:59)







Now, so so, basically, this is where we start. Let me first mark the starting point of our lectures in week 7. So basically, we already see or have seen that we have these detectability obstacles that do bother us. One of the ways to get around it was something like an Ortega construction which may or may not work for more general adaptive control problems.

It was very specific if you notice to like a spring mass damper type system. So we are not quite confident if if this will work out very well if it was not in this particular kind of structure.

On the other hand, backstepping is well known to be a rather general method, and so, we want to explore this idea of backstepping for the same double integrator system that we had or nonlinear double integrator system that we were looking at.

So backstepping is of course, a well-known to be a method to generate strict Lyapunov functions for nonlinear systems. Great. So let us go back to the table and look at our usual double integrated dynamical system which is I mean, we are calling it a double integral but it is a nonlinear double integrator because of this.

And as always, we have an unknown theta star and some function f of x, t where x is of course composed of both x1 and x2. x consists of both x1 and x2 states. And we as always, our typical objective is that our states, our x1 states track some trajectory, r and x1 dot accordingly tracks some trajectory r dot.

This is obviously because of the matching condition, because our dynamics dictate that x1 is x2. Therefore, the trajectories also have to be identically designed. So we cannot have any arbitrary trajectory for x1 and x2 but they have to be related by the derivative. And this is essentially the matching condition that we have already spoken about in detail in the previous week. Great.

And so the steps are pretty standard. So as you go on, you will start getting used to, sort of doing these steps again and again. So what do we do next? We essentially derive the transformed dynamics that is now we have generated or created an error variable and our aim is to, as always drag these errors to zero, and so now, all these Lyapunov theory of considering the origin to be the equilibrium point, it all should start to make sense to all of you by now. Yeah?

So what is the error dynamics? e1 dot comes out to be e2 and e2 dot is theta star f plus u minus r double dot. So I am not even going to try to explain how we got to 1.3 and 1.4 because we have already done this in the previous week. So if you are, you still have any confusion, please go back and refer to the notes from the previous week.

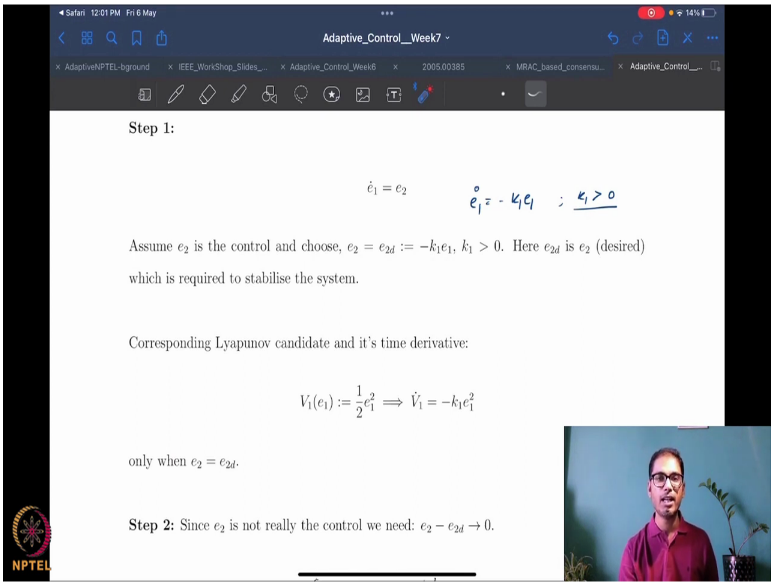
So great. So now we have this dynamics, potentially unknown quantity theta star and we want to do the standard adaptive control design. Now we already know from last week that if we use some standard Lyapunov candidate like a V equals to x1 square by 2 plus x2 squared by; sorry no x1 and x2. But if I use something basic like e1 squared by 2 plus e2 squared by 2, then I will end up in some trouble,

And this will lead to, leads to detectability obstacle, and this will lead to detectability obstacle. So we already know that. And what was the way around it? We use an Ortega construction. So now, we will try to do the same thing with backstepping.

So think of this as like a beginning step and it is a sort of apples-to-apples comparison between the two methods. Just look at it as that and nothing more for now. But of course, you will see because this backstepping-based adaptive control is something we will do for several sessions now.

So you will start to see that this has much further implications than the Ortega construction. It can be used in many many more contexts than you can use Ortega construction, All right, great. When and of course, you will also understand that the Ortega construction itself is sort of inspired by backstepping-based methods, Great.

(Refer Slide Time: 08:52)



So what do we do in backstepping? In case, for those who do not know or those who do not have not done this kind of nonlinear control design course before this, we first look at the first piece of dynamics which is e1 dot equal to e2. Forget about the second piece.

Just look at the first piece and we will assume that e2 is the control. So this variable right here, that is the second state is assumed as the control for the first state.

Remember, this is merely an assumption, an idealization for the purpose of design. You cannot actually think e2 as the control because it is in fact, a state of the system. So if I was thinking of a robot, e1 would be the position error and e2 would be the velocity error.

Again, think of an airplane e1 would be the position error, e2 would be the velocity error. It is not actually a control but the state of the system.

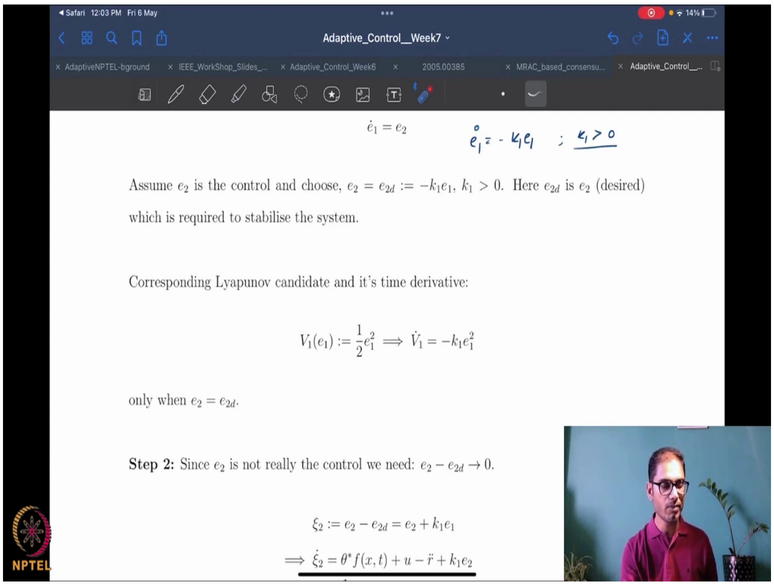
But we assume it for now, again, for the sake of the design. So what do we do? If you assume e2 to be the control, we want to design a stabilizing control. As obvious. Whenever we designed a controller, we want it to be stabilizing.

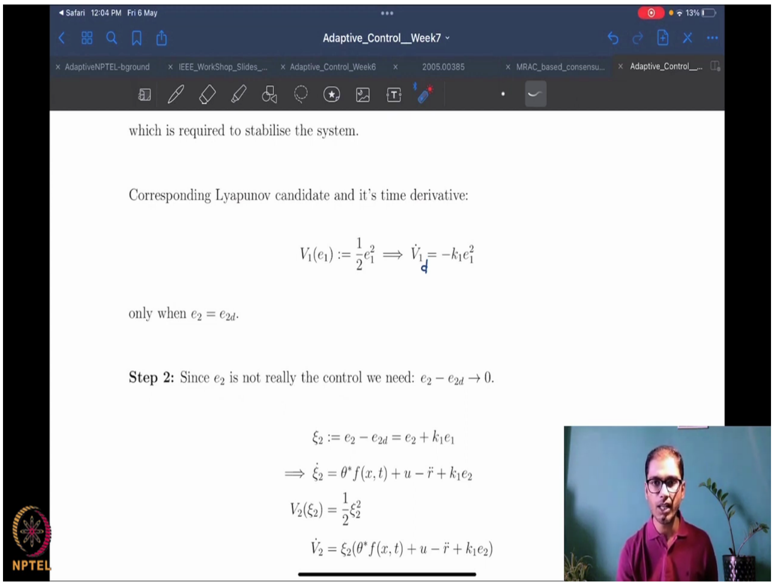
And what do we, what would be the stabilizing control? Remember, we always try to choose a model and we always try to choose a model. So what is a good model in this case? We know that if I have any e1 dot is minus k1 e1 for any k1 positive, then we know that is an exponentially decaying system.

So this is what we have been doing. We have been choosing a sort of ideal system to follow and this is ideal enough for us, So that is what we do. We choose e2 and we call it e2 desired, e2d. Because again, we know that in reality, e2 cannot be exactly this.

So we call it e2 desired and we define it as exactly this quantity: minus k1 e1 with a positive scale, k. And we know that this is under ideal circumstances: if e2 was exactly e2 desired or e2d, then it will exponentially stabilize the system. Great.

(Refer Slide Time: 11:30)





Now, we of course, also introduce a corresponding Lyapunov candidate function because honestly speaking, backstepping is a method of generating Lyapunov candidates by augmenting one piece to another piece. It is actually not a method of control design but a method of generating candidate Lyapunov functions.

And you as you might have already seen, we are by now used to design parameter update laws and control using; well, I mean, not the control yet but the parameter update laws are being designed using a Lyapunov candidate.

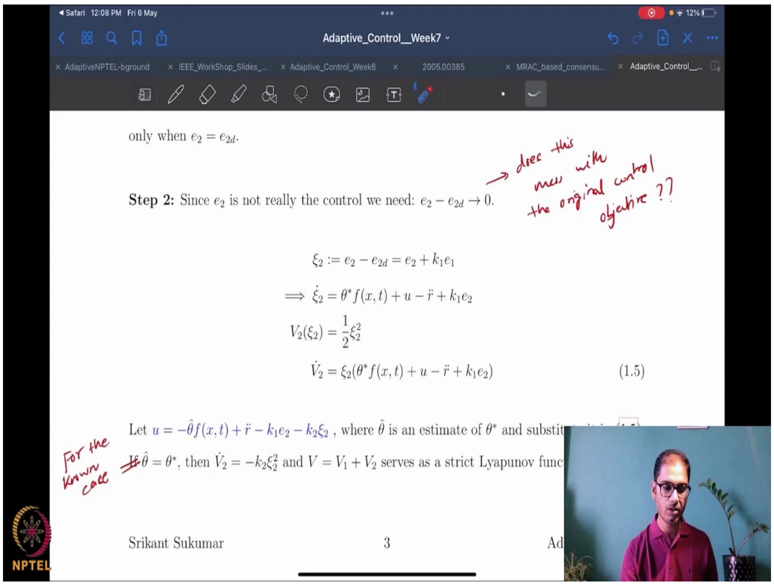
We take a derivative and we choose a parameter update law so that we ensure that V dot is negative semi-definite. So this is called Lyapunov redesign and that is what we are used to do.

So it is like we are not doing all that in this course, the notions of control Lyapunov function but essentially what we are doing is choosing a control Lyapunov function and getting a corresponding feedback, u or in this case, theta hat which is making V dot negative semi-definite, at least.

So you of course, choose a correspondingly Lyapunov function for this ideal system. Not the e1 dot equal to e2 but the ideal system which is this system. And what is the good Lyapunov candidate? It is very straightforward. Just take the first obvious quadratic that you can think of. And we have been doing this for a while now.

Of course, if this was something more complicated, you will not be able to choose a quadratic. But in this case, because it isn't, I get to choose a quadratic. And again, in the ideal case, V1 dot is minus k1 e1 squared. In fact, I can even, it might even not be completely wrong to say this is V1d because it is like V1 desired. And then this, we say clearly when e2 is exactly equal to e2 desired.

(Refer Slide Time: 13:38)



Now we go to the second step where we really understand that e2 is not really the control. So what is the next best thing we can do? So I am actually trying to help you understand also what is backstepping?

So what is the next best thing I can do? I know that e2 is not identically equal to e2 desired. But what I can do, maybe, hopefully, is that I can drive e2 to e2 desired. I will make e2 desired as the signal that e2 has to track.

Remember, my earlier objective was for e1 and e2 to go to 0. But now I am sort of shifting the goalposts. It looks like. So what I am going to say is that now I do not want e2 to go to 0 as t goes to infinity but I actually want e2 to track e2 desired.

Now one might ask, "Does this mess with the original control objective?" because we are of course, trying to, go to 0 and now we are trying to go to e2 desired. So one might ask, "Am I actually messing with the original control objective?" You will answer this. Suspense. Soon.

So what we do is we define this because we want to drive e2 to e2 desired, you will design we will, of course, define a new variable psi 2, and this psi 2 is defined as the error between e2 and e2 desired.

So this is what we have been always doing whatever we want to drive to 0, we define a new variable. So if we want e2 to go to e2 desired, we define a variable as the error between the 2, e2 minus e2 desired and that is psi 2 in this case.

And so, if I plug in for e2 desired because it was minus k1 e1, I simply get psi 2 is e2 plus k1 e or I guess I get psi 2 is e2 plus k1 e1. And now, what do I do? Construct the dynamics for this error. Same steps. The steps are similar.

Find, compute an error that you want to drive to 0. Compute the dynamics of the error and then start doing Lyapunov analysis on that. So the steps are standard. So there should be no confusion as to what direction you need to do when you start with a problem.

Always try to construct an error. Then try to design a Lyapunov candidate. Then try to define a control. Then try to find update law. Same steps. Anyway, so as of now we are not even assuming. Until this point we are not assuming theta star is unknown.

So we will find the dynamics of psi 2 which is psi 2 dot which is e2 dot plus k1 e1 dot. So k1 e1 dot is just k1 e2 and e2 dot is just the dynamics plugged in from above. Great.

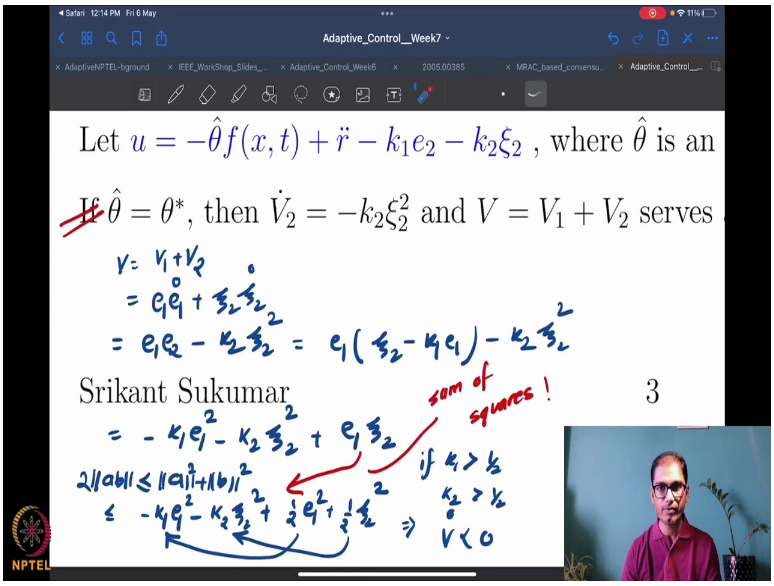
Now, what do we do? We add an additional, we add a new piece to the Lyapunov candidate. So which is again just a quadratic- half psi 2 square because we want to drag psi 2 to 0. So this is the obvious choice. And therefore, we get V2 dot as psi 2 multiplied by psi 2 dot which is just this quantity.

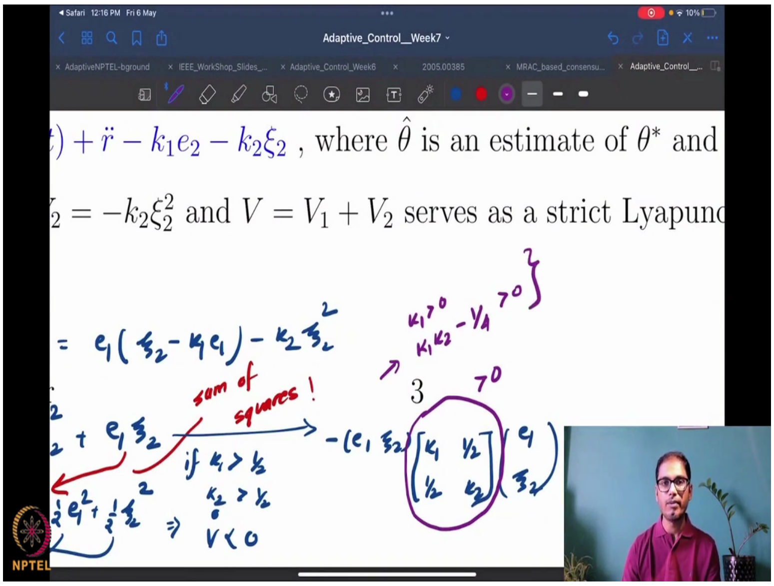
Now, if I choose my control to be of this form. What is this form? This form is essentially trying to cancel everything. I introduce a good term. Because the other terms are not definite, so we do not know how they will behave. So what we do is we try to get rid of it and we introduce a good term.

Of course, we are calling this theta hat, which is an estimate of theta star but I would say for the known case, for the known case we use theta hat equal to theta star. When we actually know the value of the parameter, theta hat is just equal to theta star.

And then what are we left with? V2 dot is minus k2 psi 2 squared. V2 dot is minus k2 psi 2 squared. And what happens? V equal to V1 plus V2 serves as a strict Lyapunov function.

(Refer Slide Time: 18:45)





So this is not complete yet because remember what we computed here was V1 desired dot not actually V1 dot. So the analysis is a little bit remaining. So if I take and I will do that here now. I will complete that here.

If I do take V equal to V1 plus V2 and now I compute the actual derivative instead of the desired derivatives, V1 was if you notice half e1 squared. So this is e1 e1 dot and V2 psi 2 psi 2 dot; psi 2 psi 2 dot.

So this is e1 dot is e2 and psi 2 psi 2 dot is what? We just did this. psi 2 infact psi 2 psi 2 dot is just wait a sec, psi psi 2 dot is exactly this guy- minus k2 psi 2 squared. As of now, there is no definiteness here; does not seem very evident.

However, notice that we have transformed e2 to psi 2. So I have to write e2 in terms of psi 2. So what is e2 in terms of psi 2? I can get that from here. So e2 is simply psi 2 minus k1 e1 and this is minus k2 psi 2 squared.

So now I start to see nice things. This is minus k1 e1 squared minus k2 psi 2 squared plus e1 times psi 2. Now we use a very, very standard trick which says that absolute value of ab is less than equal to a squared plus b squared.

So we use that to write this quantity as less than equal to minus k1 e1 square minus k2 psi 2 square plus half e1 squared plus half psi 2 squared. And now this can get clubbed with this guy and this can be clubbed with this guy.

So if k1 is larger than half and k2 is larger than half, then this is negative definite implies V dot is strictly negative definite. So I got a strict Lyapunov function. Notice. This was a strict Lyapunov function.

So if both k1 and k2 which are essentially gains of our choice. This is like a control gain that the designer can choose and as long as these k1 and k2s are greater than half, then I am guaranteed to have V dot to be negative definite.

So this method that you sort of go from here to here using this kind of an inequality is called the sum of squares method. This is called the sum of squares. And we will constantly refer to this terminology.

So whenever I say the sum of squares method, it means that I took the mixed term that look like 2ab. So notice that these things, although I have written it in terms of absolute values, although I have written it in terms of absolute values, this can actually be written in terms of norms. This is like a norm inequality also. No problem.

This is essentially like, I mean, this is just a standard a squared plus b squared plus 2ab equal to a plus b whole square type of inequality. This is just using the fact that a minus b whole square is greater than equal to zero. So that is what this is using, So this is just using the a minus b whole square is greater than or equal to zero or some such thing.

But you can, we use things like the Cauchy Schwarz inequality that we saw some time ago. Triangle inequality, Cauchy Schwarz inequality and things like that. So all these inequalities come into play when we are using the sum of squares method.

The idea behind the sum of squares method is to use any technique, any inequality, standard inequalities which will help you convert the mixed terms into squared terms which can then be combined, right, which can then be combined with this term, the other squared terms. Because that is what we did.

Because I have no idea of saying how big this is in comparison to these guys. So what I do is I write it as a sum of squares and then this can be combined now with this and this can be combined with this and so now I have a way of saying that if k1 and k2 are greater than half, then I am good to go.

Now remember that this is pretty conservative. This is fairly conservative. We can also do another thing. The only thing is it does not help us write it analytically very nicely but I can always write this whole thing as a quadratic form which is like this, which I e1 psi 2 and e1 psi 2 if I write it like that, then this is k1, k2, half and half.

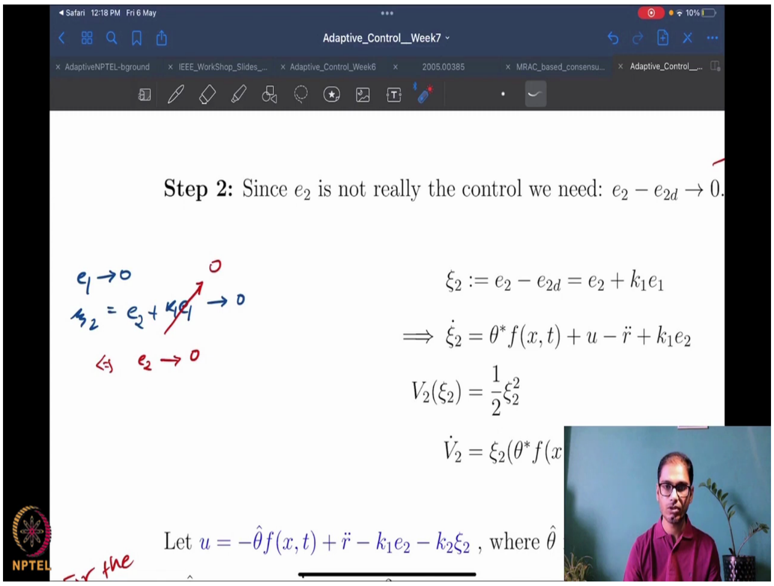
So what do you want? What you want in reality is that this matrix in between be positive definite. You have to choose the gains k1 and k2 such that this matrix is positive definite. And in fact, the conditions for that are pretty straightforward.

I hope all of you know this. This just requires that the principal minors are positive. So k1 has to be positive and k1 k2 minus one-fourth has to be positive. So these are the realistic, more realistic and because the least conservative conditions. The only thing is, it just easier to write it out in this way and more tractable. That is it.

So anyway. So the steps are not over here. Once you compute V2 dot, the point is, when you take V equal to V1 plus V2 and you compute V1 dot. Earlier, we had actually computed only V1 desired dot because we assumed V2 is exactly equal to V2 desired.

But if you do not, then you start to get an additional mixed term, e1 e2 and then you write this mixed term as in terms of psi 2. You get something like this. And then you get 2 nice negative definite terms and a mixed term which you use sum of squares to dominate. So this is really the idea behind backstepping. This is the backstepping method for the known case. It is the backstepping method for the known case.

(Refer Slide Time: 26:42)



Now, so we are sort of, sort of done in this case. See. So what have we proved? We have essentially been able to prove that e1 goes to 0 and psi 2 which is equal to e2 plus k1 e1 goes to 0. So we have proved these 2, Because why?

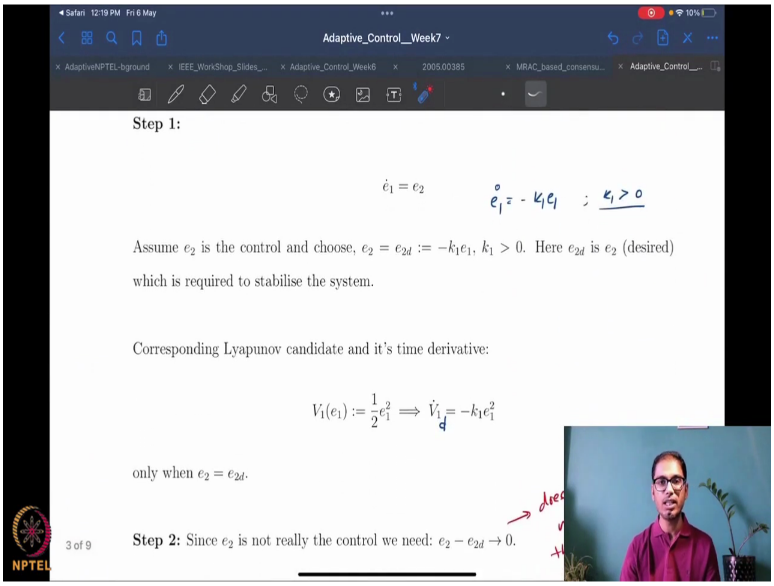
Because we took V as e1 squared plus psi 2 squared by 2 and we prove that V dot is negative semi-definite, So by standard Lyapunov theorems, both of them e1 and psi 2 have to go to 0 and of course, everything is asymptotically stable and all that nice jazz, excellent.

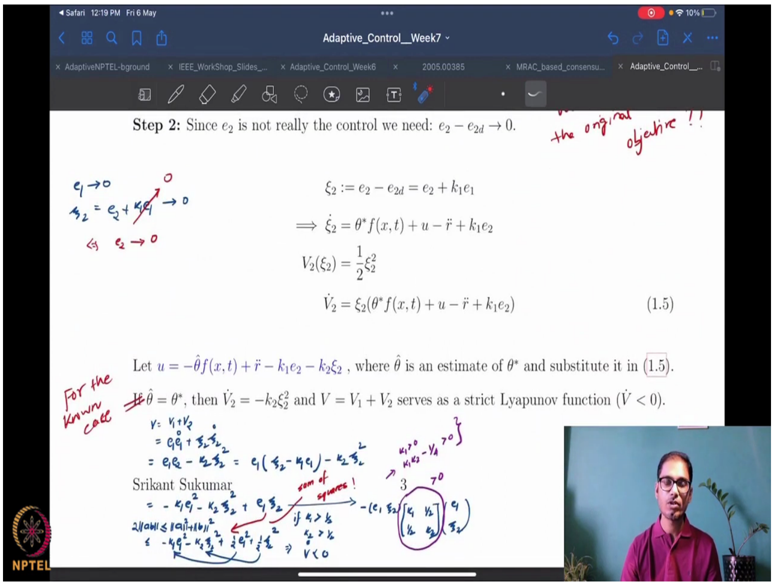
Now, we have had asked ourselves the question, “Does this mess with the original objective of driving the errors to 0?” The answer is no. Why? Because we just proved that e1 goes to 0 and e2 plus k1 e1 goes to 0 but then because in this piece e1 has already going to 0, what we have essentially proved is that e2 also has to go to 0.

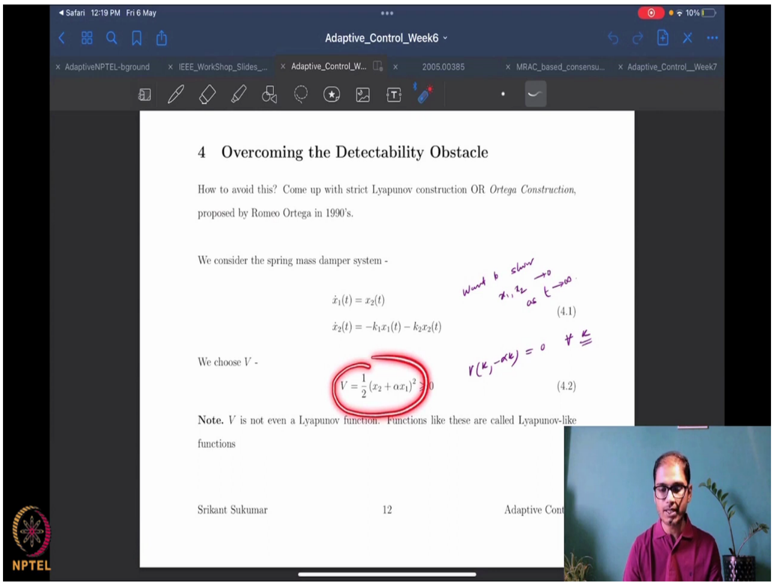
In this, if this summation is going to 0 and this piece is already going to 0 from here, then the only way the summation can go to 0 is if this guy also goes to 0, If this guy was not going to 0, then e2 would have been non-zero as t goes to infinity.

But that is not the case. e1 is in fact already going to 0 as t goes to infinity and therefore the only way for the summation psi 2 to go to 0 is if e2 also goes to 0, So this is like an equivalent thing. So we have essentially been able to recover that e1 and e2 both go to 0 as we require. Excellent.

(Refer Slide Time: 28:44)







So this is sort of how we do backstepping for a general nonlinear system. Because I say that it is for a general nonlinear system because we did not consider any unknowns. We assume theta is known, theta star is known.

So, the next step would of course be to go for the unknown theta stuff. So now notice, already you should be able to see that the Lyapunov candidate was e1 squared plus psi 2 squared where psi 2 is e2 plus k1 e1.

Now, this term is sort of very similar to what you had in the Ortega construction. I mean, again, not the adaptive problem but the non-adaptive problem. It is like x2 plus alpha x1. So e2 and in this case, you have e2 plus k1 e1 squared.

So this term essentially looks very much like the Ortega construction. But then in backstepping, there is also this additional term. So that it becomes a Lyapunov candidate and a Lyapunov function. So for this particular case, you see that the Ortega construction piece is part of the Lyapunov candidate function for the backstepping. Excellent.

(Refer Slide Time: 30:00)

Okay, so what did we look at today? We sort of started looking at backstepping in adaptive control and we started with a non-adaptive problem. Until now we have done the non-adaptive problem and we have seen how backstepping control is actually design.

Essentially, backstepping is a way of constructing additional Lyapunov functions by augmenting Lyapunov functions corresponding to each state and that is what we did. Started with a e1 state, created a Lyapunov candidate.

And then we started with the e2 state. So that is the idea of basically constructing these Lyapunov functions. So we will see further the unknown case in the upcoming session and I hope to see you there. Thank you.