Course Name: INTELLIGENT FEEDBACK AND CONTROL

Professor Name: Leena Vachhani

Department Name: Multi-disciplinary primarily for Mechanical, Electrical, Aerospace and Chemical engineering streams

Institute Name: Indian Institute of Technology Bombay

Week - 02

Lecture - 11

Hi. In this video, we will look into state-space methodology for feedforward control design. And in the last video, we had considered the feedforward design, and we have also seen when to use it and how to design using the transfer function way. Whereas in this video, we'll see how we can use the feedforward method for setpoint tracking So far, we have seen it.

What can we do for set point changes? At the same time, we will connect it with the state space representation or state space methodology method for designing the feedforward control. We'll also take one example of vibration control application, which is a very popular idea on using this feedforward methodology for controlling the vibrations instead of feedback way. All right, let's see what we are talking about set point tracking. This set point tracking, what we want here is to track the reference input R, where this reference input R is changing with time.

Means it is not a constant, just giving a step input and keeping it constant for some time. Whereas the input R is changing, varying with time. All right. So what we want here is that the output y of t should chase r of t, should approach r of t the way it is changing as t tends to infinity. So, it is a definitely a more difficult problem when R of t is an arbitrary signal instead of a known signal.

So, the change in R the input reference signal is not known to us. Let us see how we can design the control feedforward control using the state space model. We already know our state space representation for LTI system is given by x dot equals ax plus bu, and y is

equal to cx plus du. In order to design a simple feedback controller, we consider that control input u is a feedback of some gain K times the state. So what we consider in case of a simple gain feedback controller is that we consider U as, this slide is repeated again.

Let's see how we can design the feedforward controller using the state space model way. What we have here is X dot equals AX plus BU. Y is equal to CX plus DU. Instead of a transfer function way of representing representing a system, we are representing it using a state space. For a simple feedback gain-based design, we consider the control input u given by minus kx.

And as soon as we substitute this in x dot equals ax plus bu, what we get is ax minus bkx or a minus bk times X. So, what we have to consider that in the design is A minus BK should be stable enough, stable in order for the controller to work properly. So, here what we will consider is that, for example, again, we will start with constant reference input R, similar to the transfer function way we have seen it. If that is what is my constant reference input R, then my steady state is some NX times R, where NX is a vector. So control input U is now can be given by minus K X minus XS plus USS.

So in the previous case, we were considering that our equilibrium is at the origin. Instead of now, we are considering that our equilibrium is such that it is reaching to XSS. So that's the reason for this kind of correction needed. Now, this is what the X is at the steady state. My state is going to be at XSS.

So this term is going to be zero. And for that, at the steady state, we will have some bias USS sitting here. The similar way I'm calling this as a bias here. So this USS is nothing but the input given at the steady state to remain at XSS because this term is turning out to be 0. So, this finite input USS say is given by NU times R and this is required to maintain the steady state.

This is the input required to maintain the steady state at XSS. All right. So now the design part of it. So what we will do here in order to find nx and nu is we will substitute the values at steady state. So steady state has already reached.

So this is what turns out to be my steady state relationship. So XSS is anyway NX type R and USS is anyway NU times R is what we had considered or what I'm writing it here again, which is NX times R and USS is nothing but NU times R where R is my reference input. All right. So my next equation, YSS is CX plus DU at steady state is now getting modified to something like this. So my at the steady state, my output follows the reference R is whatever a demand is.

So when we try solving this using the method, the matrix manipulation itself, it turns out that we can write in the matrix form something like this. Now, with A, B, C, D known, we'll be able to find what should be NX and NU. It's very simple math here now. NX gives you minus A by B. minus B by A. All right.

Once again, NX equals minus A by B. OK. And similarly, NU can be obtained with the independent equation that we have. All right. Now control law can be, since my control law is given by U equals minus K X minus XSS plus USS, I'm simply substituting XSS as NX times R, and USS is NU times R. So my control input can be given by minus KX times this.

I am saying that this is less robust. And why? It is inevitable that we found NU and NX from the values of A, B, C and D. We just found that NU should be equal to minus A by B. It means I should know the system model very much accurately.

If I know the system matrix A and input matrix B very accurately, I will be able to design my NU correctly and I will be able to design NX correctly if C and D are known to me. And of course, R is the input reference that is given to me. This particular control input can give Y reaching to R if NU and NX are perfectly matching with the model parameters. It is completely dependent on the model parameters and that's the reason it is less robust. Alright, so far we have been chasing the constant reference input R.

What happens if my tracking is to be done when R of T is changing rapidly. All right. So what we will do here in this case is NU and NX. We already know that NU and NX rely on the information on the plant. Even for slowly varying reference input, tracking will not be accurate because it is dependent on the information on the plant itself.

All right. So what to do now? The hack here is to add integrator. Integrator will do the job for making sure that the input R is getting integrated and that's what you are trying to achieve the objective here. So this particular adding the integrator in the state space way can be done in the following way.

We will augment the system state by an extra state which is xi dot. Now, as soon as I'm adding this XI dot in this particular form, how is it giving me the reference being integrated or the output being integrated? And this is what is getting chased for. Let's see that. If we see these equations, we have XI dot given by C times XI plus R.

C times x plus r, whereas x dot is given by a times x plus bu. This is my system state space model representation. No change into it. What we augmented is this particular state xi, which is nothing but, if I say xi is nothing but integral of cx plus r. Correct? And now if I try looking at its block diagram, so what is happening is this Xi in overall in the state space way, we try looking into that the entire state which is Xi and X is approaching the origin.

It means the Xi should also approach to zero if I'm representing in this way. But if I try to understand it from the block diagram way, what we have here is this system block with its state X, input U, output Y. Now, this particular Y is, of course, since we say Y is equal to CX, this particular CX is fed back when we try writing this something similar in a transfer function way or in a block diagram way. Now, this CX plus R is integrated, right? So, what we have, we typically have is like this is my reference R and this is an integrated one by S.

Since this is an LTI system, I can clearly say, okay, what I'm giving it to the as a feedback is the integrated output. Similarly, integrated reference input. Which I can always make sure that there is some error coming out of it, and I design a controller here, which is in a feedback controller gain form is kx, u is equal to kx form. So, of course, we already know that we can have the signal flow diagram. Blocks can be moved here and there, here and there, as in a proper method way.

So I can remove these 1 by S and 1 by S from here and can add now after the summer block which is error which was similar to this thing, but now I will have an integrator

sitting here, I just moved the integrator from this path and this path to the forward path This is possible. This is what my block diagram way is. So this is now giving me Xi, some form of Xi. And this is being controlled.

So U can now be extracted from X, and it can be designed with the help of Xi and X and can provide you the necessary reference tracking. This reference tracking is being done with the help of integrator now. All right. So if I look at the feedforward way of this, feedforward representation of this. The feedforward part is coming from this integrator that we have added.

Now, this integrator is helping me in achieving the tracking as compared to just the set point regulation. And that's what the trick is. And one can do it with the help of the state space way or the transfer function way, whichever is more convenient to you. All right. Let's take another wonderful example of vibration control, which is being achieved with the help of feedforward way.

Now, we use this particular feedforward open loop control technique for input shaping. So we have been applying. So here what in the previous case, what we did was the input was modified based on the with the help of the integrator or whatnot. Now, if I have to shape this particular integrator in some way such that the output is not generating certain undesired factors is what we will look into it here. So when there are vibrations, so if, for example, there is an automobile and this is rather a very good example for the vibration control.

The seat should be designed such that driver does not experience the vibrations when a bump comes in. Bump comes in, of course, it will experience some kind of disturbance. But after this bump, once the bump is there, it should not keep vibrating on it. At the same time, the seat is sitting on a spring as well because of to avoid such bigger jerks and so on in order to give the pleasant experience to the driver for the seating experience and so on. So, here what we have is we can consider is that two rigid bodies are connected through a spring and damper arrangement.

Now, as soon as the disturbance comes, because of the flexible system, it will have certain vibrations and so on. And these residual vibrations are to be controlled, is the

control objective set for the vibration control. So we can model this particular flexible system by considering two masses connected through the spring and the damper system. This particular displacement of this particular mass is X1 and this particular mass is X2. So then one can write the mass-spring balance equations and can write the dynamical equations like this.

But it is important to to understand what we want to achieve. We want to achieve such that the displacement of two masses should move in sync. All right. So then we will consider two modes for this particular system. One is the rigid body motion, which is like both of them are moving in sync.

So the average displacement of the mass is given by X1 plus X2 by 2. Whereas, flexible motion is involved when the two masses are not moving in sync, means how much ever the first mass is moving, the second mass is moving with some other displacement and there is a mismatch between these two displacement. And we call this as E equals X1. We represent it as X1 minus X2. Of course, our objective is to make sure that this E approaches zero.

So, that the flexibility is like dampened out. Now, we can write the previous dynamical equations in this particular form very easily. We have 2m y double dot equals u and m in the form of these transformed variables y and e, it turns out to be these two equations. So our methodology for the vibration control will be to avoid the flexible mode, as we mentioned, because we don't want the two masses to move very differently, right? So this mass has moved, car has moved, car body has moved.

The seat should also move in its swing. It should not keep vibrating once the car has settled down to again in the, has moved out of the bump. So, if that is what the control objective is, then the control of the rigid body mode will be quite trivial and intuitive. All right, so now let's generalize it. Let's say there are such n-mode structures, and then each of them are having the natural frequency omega j and zeta j, and the system dynamics is now given by x dot equals ax plus bu.

So now each of this block A j is introducing a second order system given by omega j, this particular system matrix. And each of the block is giving this particular kind of an input

matrix. Whereas the state vector is now given by X1, X2, X3, X4 where each of this X3, X4, X3, X4 are repeated for individual blocks. X1 is corresponding to the rigid body and therefore for that omega naught is equal to 0. X1 and X2 are, of course, position and velocities of the rigid body mode, right?

Now, X3 and X4 of these, as I mentioned, are the states corresponding to the flexible mode, position and the velocity. So, E and E dot taken from the previous slide. Okay. In order to understand the design of the input shaper or the feedforward design here, we'll consider one mode case. All right.

So for the first mode, for only one mode means only two mass structures and only one spring and damper system in between. For that, my system matrix turns out to be A1 and turns out to be given by A1 and B1 here. All right, now what we will consider here as the input is two impulses, one at time t equal to zero and other at time t equal to T. So we will be giving two impulses, one at time T, so this is what the shaping methodology that we are evolving. And these inputs are now at a distance of two pi by omega.

Let's see why we have considered this. Let's say now delta t because of this particular impulse it generates a sinusoid. So then t minus t it will generate anti-sinusoid. So what we have is for example because of delta t what we get here is a sinusoid of this side in the flexible mode. So what we want to do here is after T seconds

For example, here we want to consider, so this is what is my T. So what we want to consider here is this should generate an anti-sinusoid. When we combine these, what will happen is this gets negated with this. And that's why it will, after one time period, the vibration control is being handled with the help of extra input that you have added, which is generating the anti-sinusoid for you. All right.

Because my system response was sinusoidal, we could understand this better and so on and so forth. But at the same time, if I have some other random input given, some arbitrary input R of T is given, Then my U of T turns out to be nothing but, since my input shaper is having this kind of a system model, system response, then this is nothing but the convolution with R of T and H of T, which gives me R of t minus R of t minus T.

So what is it? For example, my arbitrary signal is given by this, which is having the signal, it's a pulse of t f seconds.

And now my input shaper is having this particular impulse response, which is h of t is given by delta t minus delta t minus T. When we convolve with this, we get this signal, which is, of course, it was till t f. At time t equal to t, we also have the opposite signal given. And this is what we get as because of the second impulse, the superposition of first and second impulse, we get this kind of input to be given. So what the idea is, instead of giving this input, which was representing a bump, the one has to modify the input something like this.

Now, this modification happens because of this kind of an impulse shaper that we designed for. All right. So this turns out to be in the block diagram form. This is my R of T, input shaper is H of T, and this is now applied to your system. This can be combined with some feedback form and whatnot is a part of the other thing.

But in order to avoid this flexible mode, we have added this H of T block or the input shaper at the input of the signal. And that's where this is a way of feedforward design. We had seen earlier the feedforward design in terms of a transfer function way or a state-space way. Here it is a time domain where the interpretation turns out to be shaping the input based on the flexible mode time period and so on and so forth. Okay.

So, here we can see that there will be no vibration after t seconds after the command R of t has stopped. So if a bump comes of TF, we'll see that after this particular because of this input shaper, what will happen is that because we have applied a negative impulse from here onwards, the vibrations are going to be damped up. All right. So this can be since we have understood this for the single mode, one can apply this particular input shaping for multiple modes also. One can consider this particular the between car and the seat.

There are multiple modes. The flexibility within that can be a piecewise block extension. And so my input U of T can be given as the summation of these all these T minus TIs where these TIs are nothing but individual modes that we can use. That is possible because of the two masses in between. What is it between two masses kind of?

So one can model this particular engagement between the seat and the car as multiple masses piecewise way, one can consider those interfaces and so on. One has to design the input shaper for all these DI's. All right. For each of these modes, one has to make sure that the input is coming, is being applied at these DIs. Of course, one has to look forward for identifying these modes.

And this is this is a system model identification problem. These contents of this slide has been especially the vibration control method has been taken up from this particular site. And the recent one is from the June 2000 article. The idea here is to give you an insight into what you can do by simply applying it in an open loop way. Because my feedforward part is coming in a forward channel.

It is more like modifying the input in an appropriate way such that feedback methods can be used more efficiently. Thank you.