Chemical Process Control
Prof. Sujit S. Jogwar
Department of Chemical Engineering
Indian Institute of Technology – Bombay

Lecture – 41
Open loop Control and Internal Model Control

Welcome back, so we are looking at advanced control, and before the break or in the first few videos of this week we have looked at traditional advanced controllers. So we looked at simple modifications of one controlled variable, one manipulated variable, PID controllers and we saw how those could improve the performance of these PID controllers in real applications where objectives might be slightly different than having a single control variable and single manipulated input.

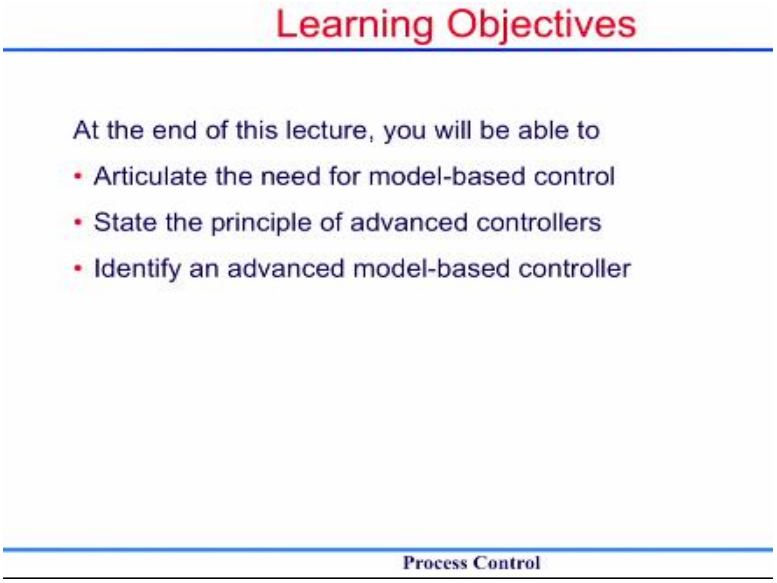
Earlier these used to be called as advance controllers because they were advancements over a single variable PID controller. Nowadays the controllers used are really advanced in terms of the logic which is used. In terms of taking the controlled variable to how it manipulates the manipulated variable and those are known as advanced controllers.

So to distinguish between the traditional advanced controllers, these, will be called as model-based advanced control because these will use some sort of a process model to improve the performance. If you go back to PID control all the information needed is in terms of what is the direction or what is the gain between the controlled variable and the manipulated variable.

As long as you know the gain is positive or negative the PID controller can work. It does not need any other information about the process. Having said that if we have information about the process can we make use of that additional information and improve the performance of the PID control. That is the focus of this week, this portion of the week where we will be using this model based advanced control.

So in terms of what are your objectives, we will see why there is a need for using process model to improve the performance of a process, and we will look at some of the advanced control.

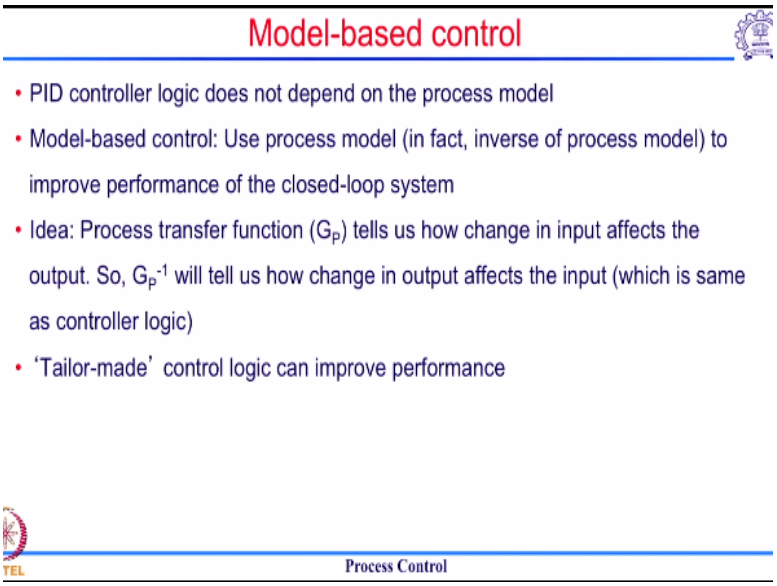
(Refer to Slide Time: 02:37)



You should be able to tell what is the main working principle behind such model-based advanced controllers. We will not be able to go through the details of implementation or consulting all the cases in such kind of a system. Each of these controllers, the advanced controllers which we are going to be looking, they are all areas of active research, a lot of books are written on them.

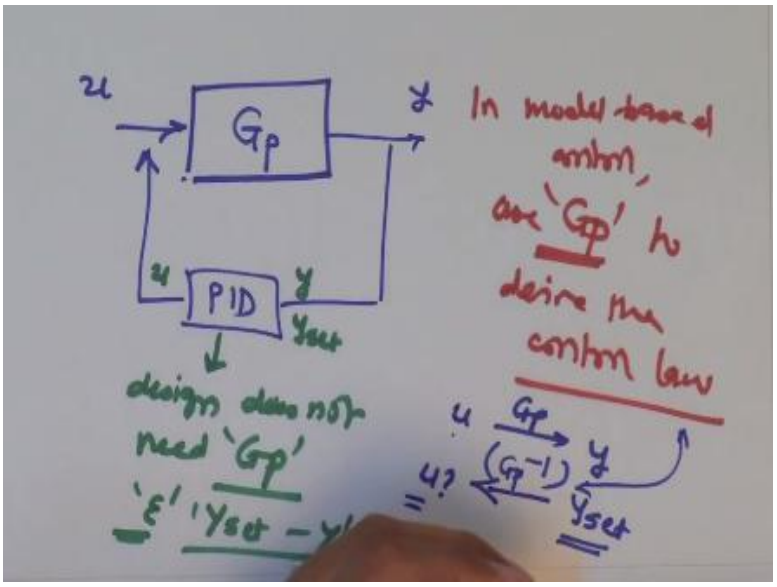
So I am just going to give you a flavor or introduction to these advanced controllers so that if you see such kind of controllers in any industry you are visiting, or you are working, you should be able to identify that and you would know what are the motivation or main components of that particular system. So all these controllers which we are going to see now are all model-based.

(Refer to Slide Time: 03:26)



So what I mean by that is when you have a PID control logic, it does not depend on the actual process model.

(Refer to Slide Time: 03:38



When you have a process, where your input is u and your output is y, you want to control your y using u and when you use a traditional PID sort of a control and let us say this is your process model Gp. So this PID control design does not need Gp per se. You can always do trial and error or logic of this control depends only on the error.

So it depends on what is my current value and what is the desired value, and the PID control law will take either a proportional action or a proportional integral action or proportional integral and derivative action. So it does not care whatever is the actual relationship between u and y. As long as you know this relationship has a positive effect or a negative effect in terms of gain, the controller would work fine.

What we are now looking at in terms of model-based control is, can we use this Gp? So in model-based control, we use this Gp to derive the control law. So when you do that, it will be a model based control because your control law is dependent on the model between the output and the input. So what is the notion of this? So for that let us look at how a control system looks like.

So typically you have this process model which is going from input to the output. So Gp tells me if I change my input how does my y change. In terms of control, it is exactly the opposite. In PID controller or model-based controller, the input for the controller is the output of the

process. So if you look at the control system your input is y and output of a control system is the manipulated input.

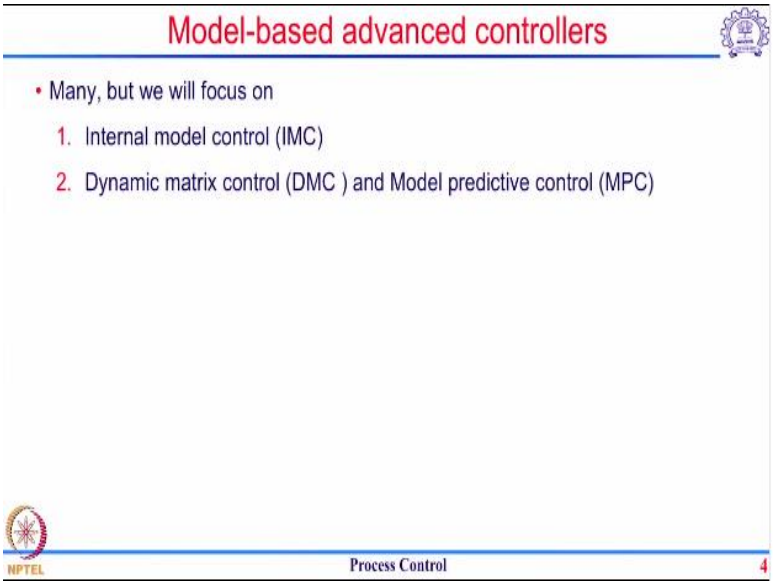
So it is exactly the reverse map of your process. In terms of the best control law what I want from a controller is, I want to make sure my y is equal to the set value. So what I want is I want my y to take a desired value. So can I do this by simply saying if I go from u to y and I use Gp? Now I want to solve the reverse problem. I want y to be = y_{set} and I want to know what u should I use.

So that is the same as inverting this model. So if I take Gp -1, it is going to tell me what form my u should take to have a value of y_{set}. So that is why when you say a model based control it is the control law which is going to use Gp -1. So even though I said it uses Gp, this model it actually uses the -1 of the model because the ideal controller model is the -1 of a process model.

You want to move from y to u and the process model is from u to y. So when you use such kind of a control law which is dependent on let us say in -1 of a process that will be a model based control. So you can see that these control law will not be universal like a PID control law, which takes error multiplies it by some gain Kc or takes its integral and multiplies it by Kc/τ_I.

So that is a universal control law. Here the control law will be dependent on the form of Gp, that is why it is a tailor made control law and as you are using process information you will see that the performance will be much better in most of the cases than a PID controller.

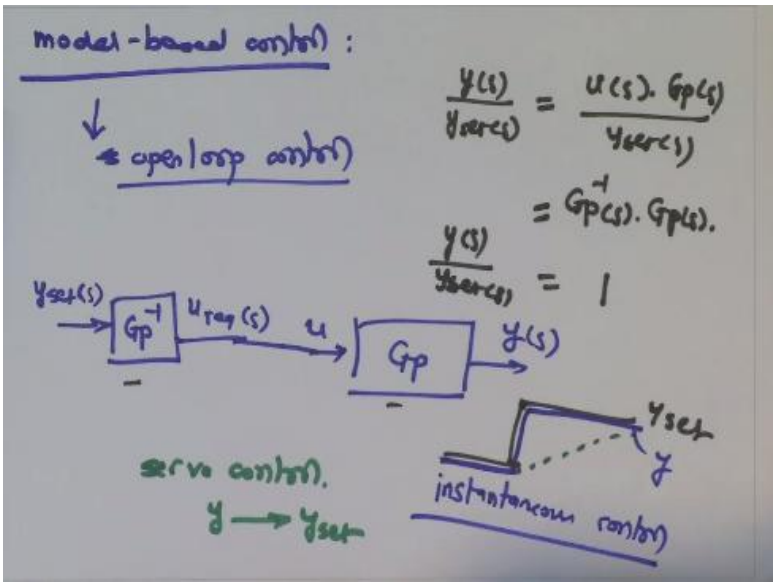
(Refer Slide Time: 08:13)



So there are many such model-based advanced control strategies, we will be focusing only on 2 of those strategies. The reason I am selecting those is because they are very popular in terms of model-based advanced control, and both of them have been commercialized and there are a lot of industrial application or industrial implementations of both these advanced controllers.

So we will be focusing specifically on these two control strategies. One is an internal model controller known as an IMC, and the other is dynamic matrix control or DMC which eventually gets in the current form it has been improved to what is known as a model predictive control or MPC. So let us start with IMC or internal model control, and its root lies with what is known as open loop control.

(Refer to Slide Time: 09:22)



So when we talk about model-based control, the first or the simplest way it can be implemented is open loop control. If you want to contrast it with PID control, PID control is a closed loop control. You had a very nice closed loop between the control variable and manipulated input, here you will see that that is no longer required when you want to do a simple model-based controller.

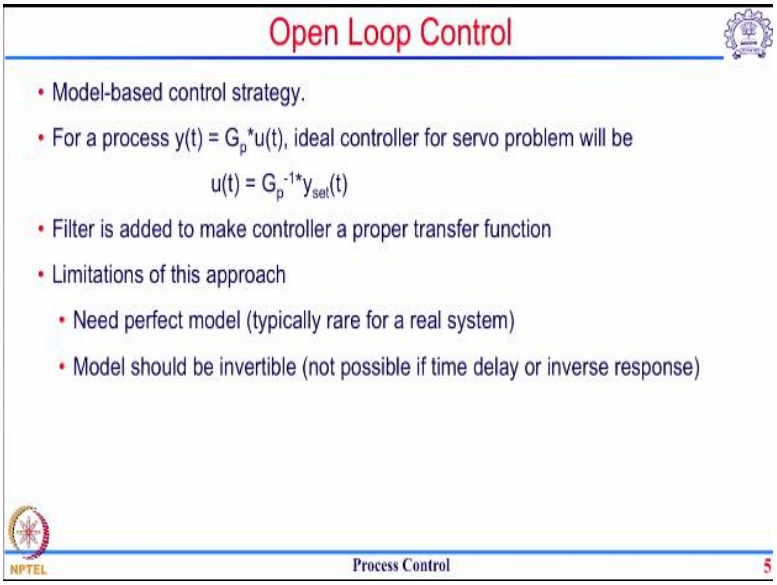
So let us say this is your process which goes from u to y and I want to look at the servo performance of this or servo control. So I want my y to go to a value of y_{set}. So what I can do is, I have this y_{set}, I can pass it through Gp^{-1}. So whatever is the u required, I will give it to the process. Let us see what this system does. Again let us say we are working in the Laplace domain.

So if you want to write your transfer function between y(s) and $y_{set}(s)$, it is the multiplication of this and this. So you can see that y(s) = u * Gp. So u(s) * Gp(s) / $y_{set}(s)$ and u(s) itself is = Gp^{-1} * $y_{set}(s)$. So $Gp^{-1}(s)$* Gp(s), $y_{set}(s)$ and $y_{set}(s)$ get cancelled. So this is = 1. So you can see that if the transfer function between y and y_{set} is 1, what it means is, if I give a step change in y_{set} then my y is going to follow y_{set} exactly.

Even though I am drawing it slightly differently they exactly follow each other, so this is y. So you have achieved your servo control that you have moved your y from current value to new y_{set} and you can see that it is an instantaneous control. This was not possible in the PID control. In PID control even though you gave a step change in y_{set} your actual output would take some time to go from the current value to the final desired value by depending on what are your controller parameters.

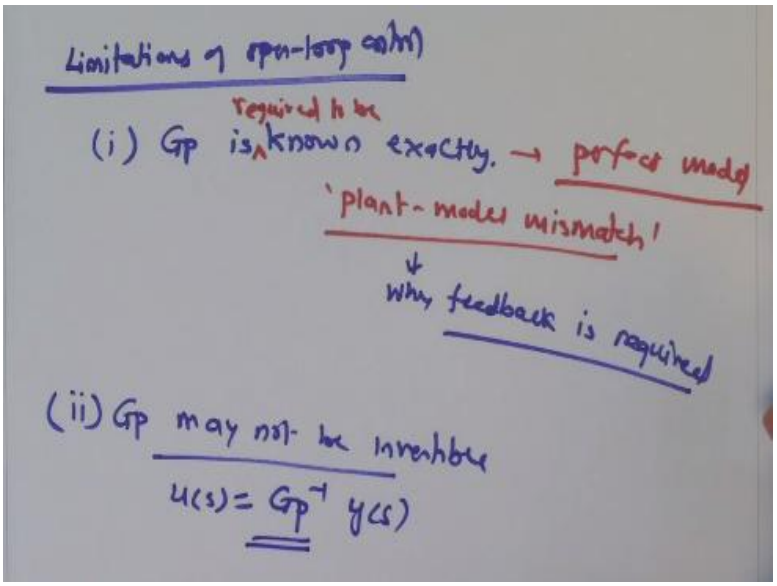
So you can see that this is much better than your traditional PID controller simply because you are using the process information. You exactly know how your system is going to respond or what is the exact input which is going to give me the desired output. So it is such a cool strategy why do not we use it, because there are certain limitations in terms of practical implementation.

(Refer to Slide Time: 13:00)



So first and foremost is here we are using the knowledge that Gp is known. So what are the limitations of this?

(Refer to Slide Time: 13:17)



So first and foremost is Gp is known exactly. If Gp is known exactly, you can have Gp^{-1} to be known exactly, and then you can take a perfect control and get an instantaneous control. Now when you talk about a real process, you will hardly have a model for the process, or there is no such model which can capture the reality. So in that case whatever Gp or the process model you have is an approximation of the actual behavior.

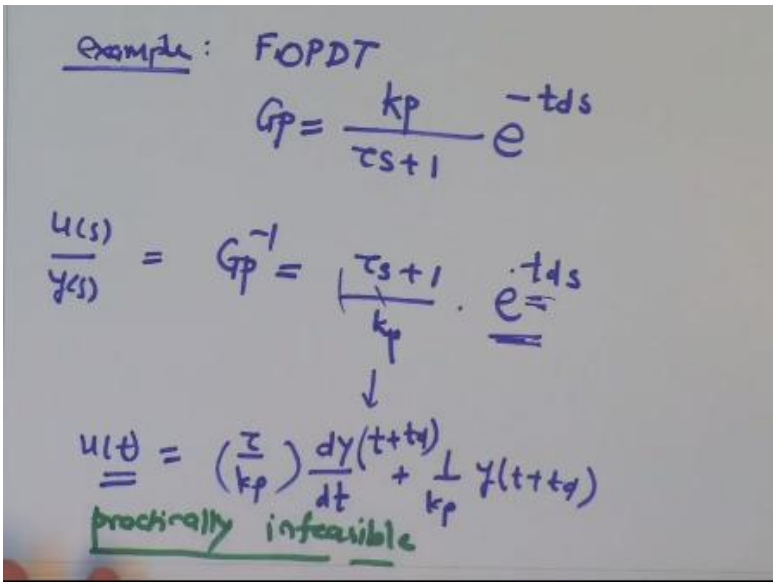
Because of that, you would never have a perfect model. So the perfect model is not available. So here Gp is required to be known exactly. So perfect model is required which is typically not possible and what you end up with having is known as the plant-model mismatch. So if

that is the case, your open loop control will not work. Because it will assume that the process is going going to follow a certain relationship, but in reality, the actual system does not follow that relationship.

So because of this plant model mismatch, you will never reach the set point, and that is one of the major reasons why a feedback control is needed or why feedback is required. So when we talked about feedback control and its robustness, the main reason comes from the fact that unless you have a feedback from a system you do not know whether you are taking a correct action or not.

So open loop control because of lack of feedback would never be able to counter plant model mismatch. The second limitation is Gp may not be invertible. I said that your control law is $Gp^{-1} * y$. So that assumes that a Gp^{-1} exists. Now let us take an example that you have first order plus dead time process.

(Refer to Slide Time: 16:03)



First order plus dead time process. So that your Gp is$[Kp /(\tau s + 1)] * e^{-tds}$. When I say Gp^{-1} what would I get, $[(\tau s + 1)/ Kp]* e^{tds}$. So you can see that now you have a positive delay or positive dead time, what does that mean? That means I want to take an action t_d times before the current value.

So this is u(s)/ y(s), so what you are really going to get is. So this is τ/Kp, as is generally the derivative, so you will have dy/dt + (1/Kp) * y. So this is the inverse of this particular piece, and then as you have e^{tds}, the values of y and the derivative I want is at $t + t_d$. So to compute

the output, the manipulated input at the current instant what I want is the derivative of the output at a future time and also the value of output at the future time.

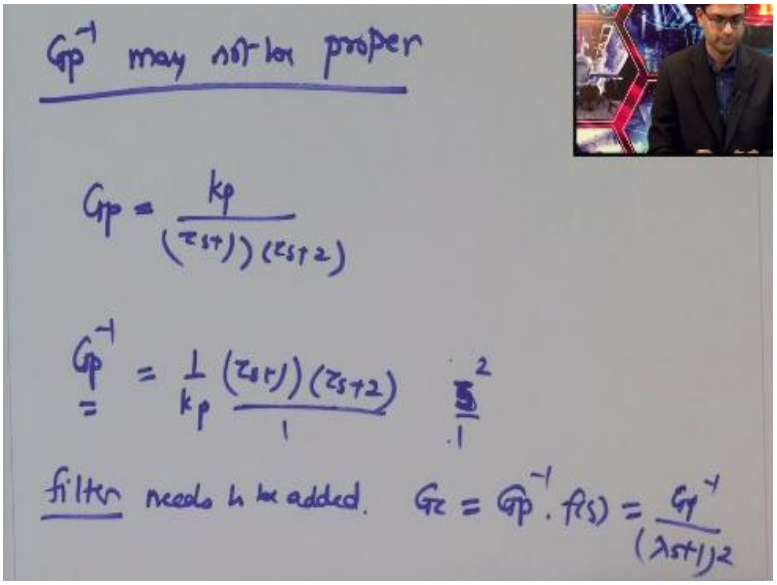
So as the process has not reached that particular future time, there is no way I can use this information and calculate the manipulated variable. So this is practically infeasible. Which is true because what is FOPDT, first order plus dead time model, when I make a certain input move unless a time of t_d is spend the output is not going to show any effect. So if that is true, then I cannot have my output follow the set point exactly or instantaneously.

You have to wait for a certain time before your system can respond to the input, so there is no way you can make your system instantaneous or let this dead time go away by having a model-based control. So physically that is the interpretation that you cannot have an instantaneous control when you have a dead time in your system.

What I mean to say is whatever mathematical form you get, it cannot be practically implemented because you require future values of the process. So this is one example that if you have a dead time, you cannot invert the model. Similarly, you can analyze that if you have an inverse response or zero on the right half plain again, you cannot invert that particular model.

So that is the limitation of open loop control that you may not have a process model which is invertible. So, in that case, your system may not give you the instantaneous control. Another limitation is when you take an inverse of the transfer function, the resulting transfer function may not be proper.

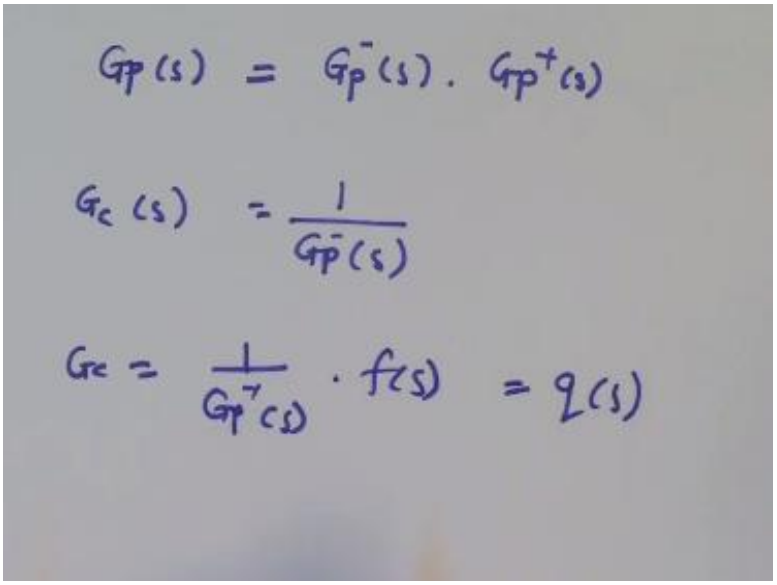
(Refer to Slide Time: 20:14)



So Gp^{-1} may not be proper. What I mean by proper transfer function is, a transfer function where you have the degree of the numerator less than the degree of denominator which can be physically implemented. If you take Gp as Kp /[(τ s + 1)(τ s +2)], a simple second order system. Then Gp^{-1} becomes (1/kp)* [(τ s + 1)(τ s +2)]. So you can see that the numerator has τ^2, but the denominator has 0.

Numerator has s^2 as the degree of 2, but denominator has 0. So no physical controller can give you a transfer function like this. So that is why you cannot use directly a Gp^{-1}. What you have to use is a filter. So filter needs to be added so that you make the order of denominator at least equal to the order of numerator. In this case what you would use is, Gc or the controller transfer function will be Gp^{-1} times what we know as a filter so that it will be Gp^{-1} /(λ s + 1)^2, so that you will have s^2 in the numerator and s^2 in the denominator. So this 1/(λ s + 1) is a filter, a simple first-order process and by having a filter of higher order, you can make sure your controller transfer function is proper.

(Refer to Slide Time: 22:15)

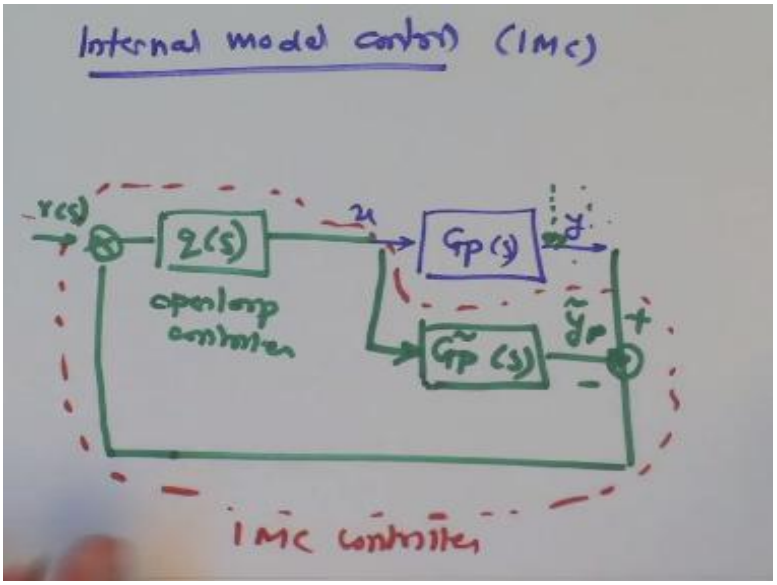


So you take your process transfer function, let us say Gp(s) is your process transfer function, you break it up into an $Gp^-(s)$ * $Gp^+(s)$ where $Gp^-(s)$ is the part which is invertible, and $Gp^+(s)$ is the part which is not invertible and then in terms of your control you invert the invertible part. So you take this, and you simply add the filter.

That becomes your controller which here will be referring to as q(s). So you take only the part which is invertible, invert it and then add a filter to it. Now we have seen there are some limitations and mostly they come from the fact that there is no feedback in this open loop control, in terms of disturbances if an unknown disturbance affects the system, there is no way this open loop control will be able to handle that.

So to satisfy this, get rid of these limitations, the practical implementation of this is known as an internal model control or IMC.

(Refer to Slide Time: 23:58)



So what we do in IMC, the motivation is that because of lack of feedback your open loop control is not able to satisfy plant model mismatch. So you introduce feedback into the system. So what you do is let us say this is your process Gp, which goes from u to y and Gp is the actual process. You may have the transfer function for it or you may not have transfer function for it.

But you can have a representation of the process. So let us say you have a model of the process we call it as $\widetilde{G_p}(s)$. So whenever you give some input to the process, you compute what would have been the output if this was the model of the process. So this takes into account whether there is a difference in terms of the plant and the model which you are going to use for the control.

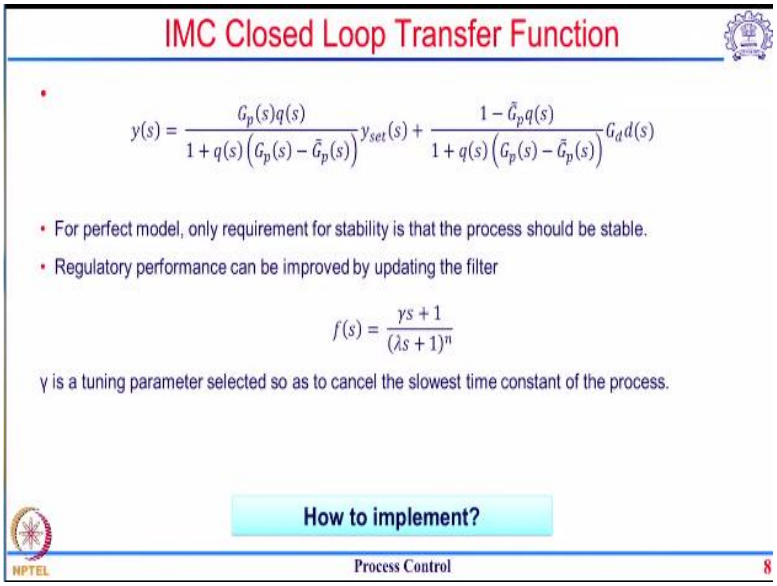
It also takes care of any disturbance which was affecting, which was not part of the model and you take the difference between the two. So you take this as positive and this as negative. So you take the difference between the actual output and the predicted output, and then you use the feedback in terms of whatever is your set point and then this is your model based controller q(s) and it will give you the u.

So this was your open loop controller, the way we derived q(s). The only difference is now you are also using the difference between actual value and the predicted value. As you are using this difference between y and y_p to drive this open loop controller, it will ensure that if there is any disturbance which is affecting the process it will trigger the difference between y and y_p and then that will be rectified to the open loop control.

So you will see that by incorporating this sort of feedback between the actual value and the predicted value you are still using the model of the process to compute the controller and this open loop controller also uses that, so it is a model-based control strategy. So your overall controller if you want to see is this. So all these actions are done inside a controller. So this would be your IMC controller. It has two parts; one is a prediction of the output and other is this control law.

So by having a desired value as well as the error between prediction and model it is going to prediction and actual plant, it is going to compute the manipulated variable. In terms of the transfer function if you compute it you would realize that the corresponding transfer function can be given as y(s) will be this $y_{set}(s)$.

(Refer to Slide Time: 27:23)



And this is the disturbance transfer function, so it looks very much similar to your PID control law. So if you try to recollect, this was your servo transfer function, and this was your regulatory transfer function, the only difference now is it is in terms of q(s), which is your model based controller and it also incorporates this factors Gp – $\widetilde{G_p}(s)$ which captures the plant model mismatch.

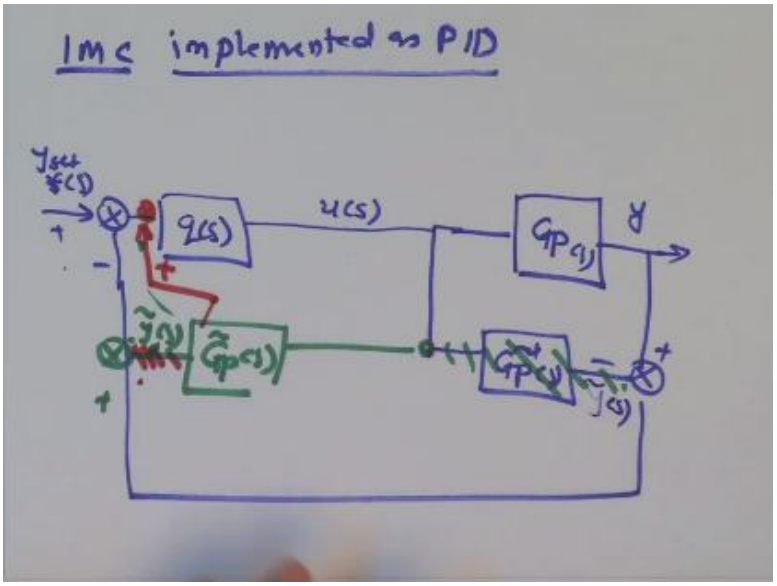
So if your model captures the plant accurately which is not practically the case, but if that is possible then this term would go away, and your entire control law will be simply Gp(s)*q(s) over 1, and so the denominator will simply be 1 in that case. So another advantage of this

control strategy is that for this system to be stable all you need is your process should be stable.

As long as your process is stable, your controller remains stable that was again not the case in terms of a PID control which we saw that in the cases of dead times or third order or higher order systems there was a limitation in terms of controller parameters to ensure stability. So these also an additional advantage in terms of going for IMC. How do we implement this IMC control?

There are multiple ways in which this IMC control can be implemented. I will show you a simple way in which IMC controller can be implemented, that is by converting it the structure into an equivalent PID control. So we can take the same structure which we had earlier.

(Refer to Slide Time: 29:14)

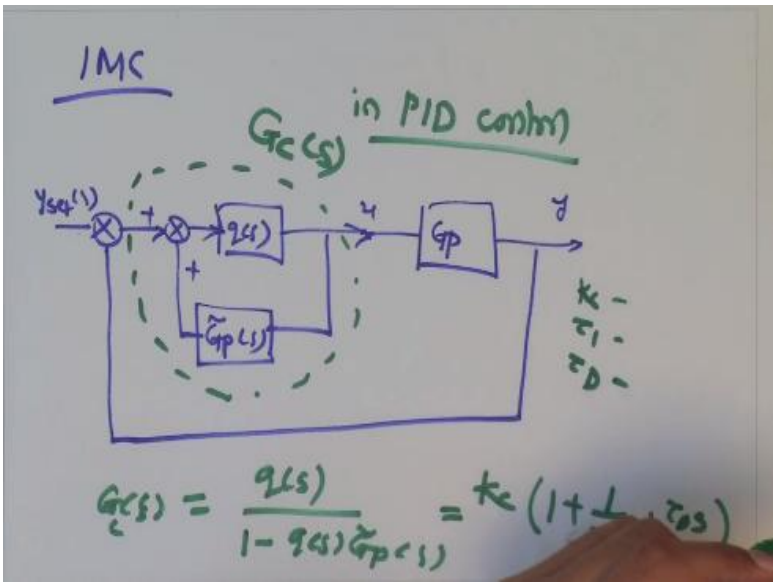


So IMC implemented as PID. So this is your process Gp, this gives you y, you also have this $\widetilde{G_p}(s)$ which is the prediction. You take the difference between these two, then this is your set point, let me call it y_{set} and then you have this controller q(s), which is going to give you u(s).

So this is y, this is $\tilde{y}(s)$, so we can simply rearrange this to be on the other side instead of showing it here, I can show it in this direction, does not make any difference, right? And now what you have is, this $\tilde{y}(s)$ is subtraction here. So this is + and this is -, so there is one subtraction here and the second substraction here. So if I want to move this $\tilde{y}(s)$ directly here, I can simply add it to this particular portion.

So I can disconnect it from here, and if I put it here, then this is - * - = +. So I can simply have a + here. So the same structure if I simply rearrange it what I get is a structure like this.

(Refer to Slide Time: 31:21)

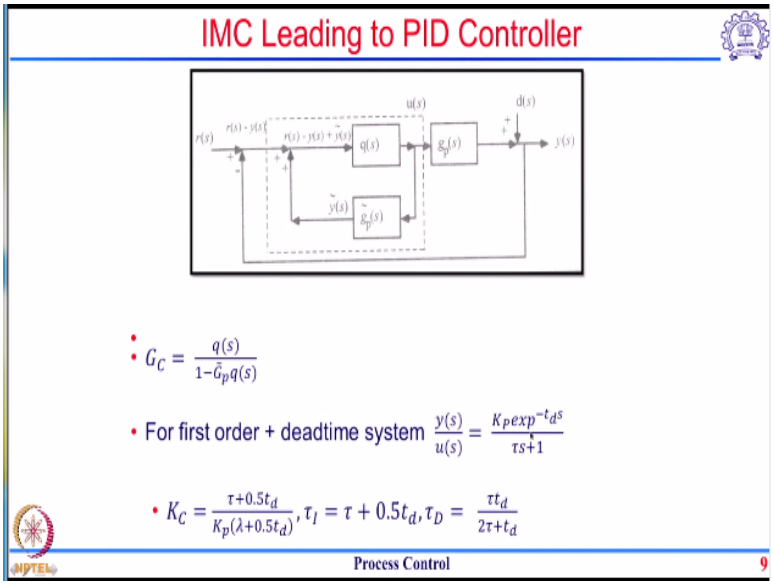


So can you notice something here? What we have done is, this is equivalent to IMC. So as I showed earlier, this is still IMC; I have rearranged the block diagram without changing the output and what you are getting is the controller transfer function which we were using earlier. So you have a process transfer function, you have y, you compare it with the set point, you get an error.

And then your controller works on the error. So this is the same as a Gc in the PID control and in this case this Gc(s) = q(s)/(1- q(s)* $\widetilde{G_p}(s)$). So for IMC controller this Gc is this and if you want to see that this IMC controller should have a form of a PID controller, I can simply equate it to Kc*(1 + 1/(τ_I*s) + τ_D*/s) and if I compare these two in order to get realisation of IMC controller in terms of PID, what you will get is you can simply compare these two and it will give you the values of Kc, τ_I and τ_D for a particular process.

So here I am showing you that if you have a first order + dead time process.

(Refer to Slide Time: 33:32)



Then what you will get is Kc as this, τ_I as this and τ_D as this. So what you are essentially doing is if you use a PID controller with these values of the tuning parameters then that PID controller will work as an IMC controller and why is this still a model based controller? Because ,Kc is dependent on τ, t_d, and Kp which are model parameters.

Your τ_I is a function of τ, which is also a model parameter. So it is still a model-based control, the only thing is that you are implementing this IMC as a PID controller. So that is one way of implementing IMC, I am not saying that is the only way IMC can be implemented, when you are doing this you are making certain assumptions about the process.

The effectiveness of IMC might be limited a little bit, but the advantage you gain is that by using the same architecture of PID control you can implement an advanced model-based control using the same architecture. So we will take a break here, and after the break, we will move on to the second advanced control strategy which is known as a dynamic matrix control. Thank you.