Process Control- Design, Analysis and Assessment
Professor Dr Raghunathan Rengaswamy
Indian Institutes of Technology Madras
Department of Chemical Engineering
Lecture 20
Traditional Advanced Control - Part 2

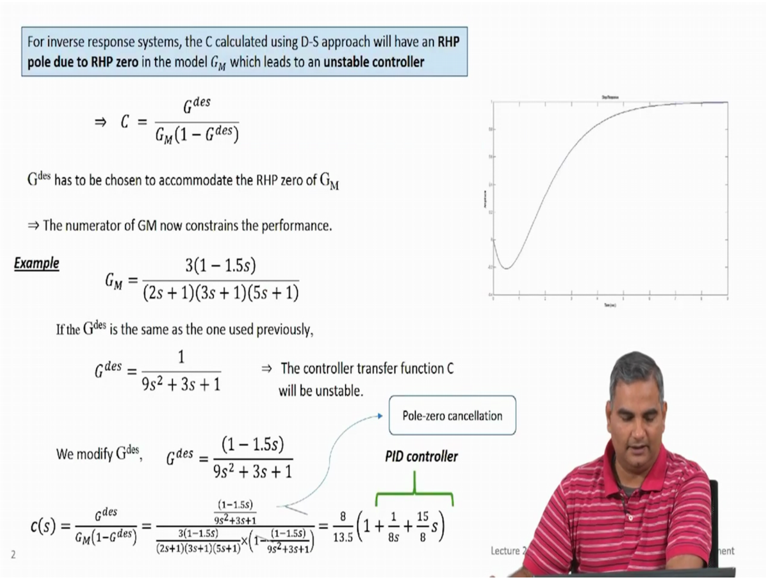
We will continue with the lecture on inverse response systems and controlling inverse response systems. In the last class I explained what inverse response means and how you identify inverse response system by looking at the transfer function model. Just as a quick recap, inverse response systems are systems where the initial duration of change is different from the final change so assuming that we start deviation variables at a value 0, if the final value that the variable takes for example, in response to the step in the input if it is positive however, if the initial slope is negative that means it is going to be certain negative value for at least initial part before it is going to turn back and then go and take a positive value, so this is what we call as inverse response system.

And I describe an example an engineering sample of an inverse response system which is a boiler, and I also mention that there are several other system which show this inverse response. And we also saw that we can quite easily identify if there is inverse response by looking at the transfer function, so for the 1st time we start focusing more carefully on the numerator transfer function and if the numerator has what we call as right half plane zero then we can say that the system have inverse response. And the generalisation of this is where I said if there are odd number of right half plane zeros in the numerator of the transfer function then we will have inverse response.

And then I will also try to explain to you why inverse response systems are difficult to control, I talked about a bike where you want to turn right but it initially turns left which makes you really confuse and then the control becomes hard because you do not know after it turns left when is it going to come back to write and so on. So that is the physical reason why the control of inverse response system becomes complicated and that is also very nicely seen in a direct synthesis approach and there I said if you remember the controller formula in direct synthesis approach is 1 over transfer function of the model GM times G desire divided by 1 minus G desire. And because we are inverting the process model, any right half plane 0 in the process will become a right half plane pole in the controller.

So if you want to get an ideal response whatever that you desire then that will mean that you will have to have a controller which has to act without stability and that is never good. Basically what this essentially means is that somehow you have to take care of this and that somehow is actually modifying our expectations for what G desire can be, so till now we have never put any constraints on G desire, we said you choose any G desire you want but now we will have to start enforcing constraints on G desire and we will have to see how we enforce constraints on G desire so that we do not design a controller that becomes unstable so that is the key idea.

(Refer Slide Time: 3:32)



Now let us understand how this is done mathematically, but before we do that let us think about the example that I thought about in the last lecture the bike example. And then say look if you already knew that if you turn the steering write the bike is going to turn left for a short bit of time and this is a problem that you have to somehow handle, you still have to turn right you also know though it turns left for a short bit of time it will ultimately turn right. And if this is the case then what is the best solution that we can think of in terms of the control problem?

The best solution would be in modifying whatever notion of the desire response of the system is, so if I know that the bike is going to turn left for a short period and I still have a G desire where I say okay whenever I want to turn right I should immediately turn right so that is a G desire which is unconstrained, I can ask for anything and I am going to ask for the fact that whenever I turn right I should turn right though the system itself is prone to go left for a little bit before coming right, I say I do not want it right.

Now that is not possible both mathematically and physically we can see this, so another approach might be understanding that the system does have inverse response, what you could simply say is okay I know that when I turned right I have to be completely turning right at all times as G desire transfer function is not possible. So maybe what I should do is I should say okay let me make a compromise, I definitely do know that once I turned right after a while the bike is going to turn right so maybe what I am going to do is I am going to live with the inconvenience of turning left for short before the bike does exactly what I want it to do, so that is the basic idea of modifying the G desire so that we still have a stable controller ok.

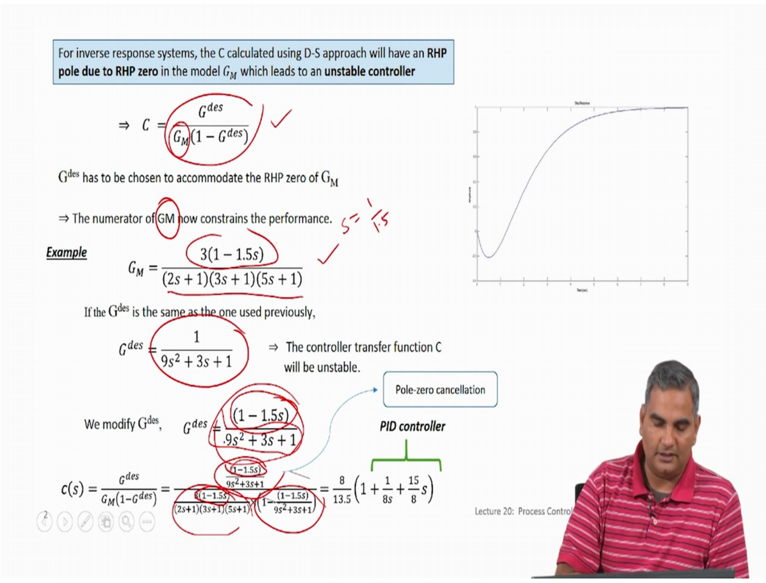
So what we are going to do is we are going to tolerate this inverse response a little bit in our G desire that we ask the closed loop system to follow, so that is the basic fundamental idea in solving this problem. So this basically says okay if I have a performance limitation in terms of RHP 0 then the ultimate closed loop performance will suffer a little but I know that I am going to tolerate this and I really do not worry about getting the best performance that I specify in an unconstrained fashion ok, so let us see how all of this works out in terms of Math that underlies these ideas. So I already talked about the controller form which is G desire minus G model times 1 minus G desire so the numerator of this G model is basically the one that constraints the performance ok, so let us take an example and see what happens here.

If you have a gym model like this and looking at this you can say that there is going to be inverse response behaviour and you are going to say that there is an inverse response behaviour because 0 of this transfer function is S equal to 1 over 1.5 which is in the RHP. You also notice that all the poles are in the left half plane, S is – half S is -1 over 3 and S is minus 1 over 5, this is the reason why I am going to get an inverse response ok. Now if I use this G desire which I have chosen here for example, and then say let me go ahead and compute the C, you will see that this C will become unstable because this will go into the denominator here. So I have in the denominator polynomial right half plane pole which will make the controller unstable, so this is something that we need to avoid so this is what we talked about till now.

So how do I avoid this? So I avoid this by saying that I will have to give up something in terms of G desire, my ideal G desire was this but I am going to give up something by actually including a right half plane 0 in the G desire. Basically what I am saying is I am going to modify my expectations and I am going to assume that I will have inverse response in the G desire also. Why would I have inverse response here, because just like how we analysed here if we analyse this transfer function here, this G desire transfer function has a right half plane 0 that means the G desire itself will show you inverse response. So basically what we are saying is we are completely translating whatever I told you physically into mathematics.

I said if you want to turn right if I had just only this then I say I want to always turn right I do not ever want to turn left, right if my final intention is turning right. However, we knew that it was not possible so the sensible thing to do is to turn your steering right and wait and do nothing, the system will turn left for a bit and then start turning right so that means in the performance it could not stop the bike from turning left a little bit and that will actually deliberately introduce ourselves in this G desire and then say okay this is fine okay because this is something that I am going to give up. And I am going to show you what happens when I give this up in terms of performance drop, I will now get a controller which is stable.

(Refer Slide Time: 9:23)



So now with this G desire I go back to the same equation here and then put this here, so this is my G desire, this is my GM and this is my 1 minus C desire. Now if I do the algebra the key thing that you want to notice here is that because this 1 over GM gets it is right half plane 0 in the transfer functions to become a pole that pole is going to be cancelled by the 0 that I have introduced in the G desire because G desire is in the numerator. So this and this will get cancelled that is how we get rid of the problem of instability of the controller then we introduced this inverse response behaviour in G desire. Now once we do this and there is a reason why I have chosen this 9 S square + 3 S + 1 because by choosing that I can get a control form which is of the PID controller form ok.

So this is again just to remind you again this is K C divided by 1 by 1 plus 1 by Tao I S + Tao D S right, now K C is 13.5, Tao I is 8 and Tao D is 15 by 8. So basically what this allows us to do is it allows us to design a PID controller for inverse response system, so it is still in the form that we are comfortable with. However, there is an understanding that we are losing certain performance and that performance loss is actually deliberately introduced by us in this G desire transfer function so as to keep this controller stable.

Notice, in doing this we have a stable controller get reasonable performance and we get a PID controller, so traditionally people have been using PID controller for inverse response system and you kind of ask why does it work reasonably well and here is the map that shows why it works reasonably well, if you are able to or if you are ready to give up a little bit on the performance that you require out of these systems.

Now mathematically another interesting thing that you should notice here is that I am cancelling, if you look at the controller transfer such I am cancelling 0 of the controller transfer function because this is in the numerator of the transfer function with a pole of this same transfer function. So if I look at the controller transfers much and I am doing what is called the pole 0 cancellation. Now whenever people do this pole 0 cancellation they are always worried about this because like I said before if you get some stability using a pole 0 cancellation and if the model is not good representation of the process and actually the 0 of the actual process is not the same as the model than would it still work in real-life is always a worry that people have.

So in this case you got to remember something and then when we moved away from the traditional undergrad control topics to more advanced topics such as one that I am teaching here, I said for the first time we are going to think about a model and the process as a distinct transfer function. We want the G model to be actually equal to GP in which case there is no process model mismatch however, in reality GP will be slightly different from G model. So we have to be very careful when we think about this calculation to understand where we are using G model and where we are using GP because traditionally till we came to this direct synthesis and then I started talking about model and inverting a model and so on.

We typically say okay here is a G process design a controller based on the G process and then put the same G process transfer function in your closed loop block diagram and then see how will it does ok. So in that case we are you know intuitively or implicitly assuming that the process and the model transfer functions are the same however, I started making this distinction about the model here when we started doing the direct synthesis approach because irrespective of what G process is what we are saying is I have an idea for model for the process and I am going to make all my judgements based on the model itself. So here I am saying okay this is the transfer function model for the process and I am going to make all my controller decision based on this model, so this model is actually a conceptualisation.

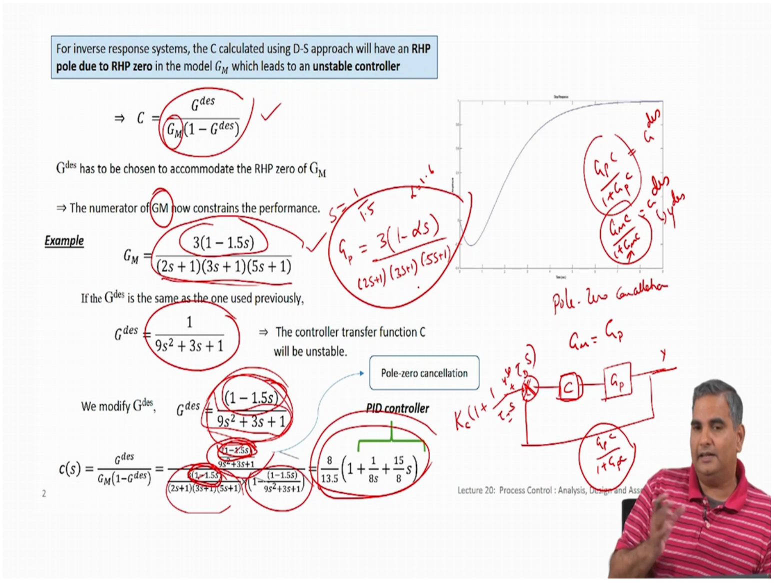
The true process could actually be something okay just to make a point ok so I could have something like this 2 S plus 1, 3 S plus 1, 5 S plus 1, so the true process could be same structure the denominator being the same but the numerator is slightly different from this 1 – 1.5 S, this is 1 minus Alpha S, may be Alpha is 1.6 and so on right. So there could be a mismatch between G process and G model but this does not come into this computation because the controller computation is always done using the model right, so whatever this Alpha is I have already conceptualised the model as being this and I have already conceptualised that the RHP 0 is at S equal to 1 over 1.5 ok so that is already decided.

So even if the 2 process is not exactly the same as the model, the controller does not change because the controller works only with the model, while it looks like a simple concept here many times we can miss this because if I simply ask you this question oh you did all this controller and did this pole 0 cancellation but what if the process is slightly different from this, will the controller become stable and will this pole 0 cancellation not work. So the answer is pole 0 cancellation will always work in this case because I am anyway going to work with the model, and irrespective of what value Alpha takes I am going to use this model transfer function. So as long as I use this model transfer function I use this G desire, this and this will get cancel and I will get this PID controller.

Now then you might ask okay if that is the case then what is the impact of process model mismatch, so the impact of process model mismatch you study now when you put this controller in your closed loop block diagram you use this G process right, so the controller has been designed assuming the model is this but actually you check the effect or the performance of this controller on the process which could be different from the model you use here in designing the controller. So that will be realistic where okay you go to plant, you have a concept of a model and then you use that model to design a controller but when you finally test the controller that controller is being tested in a real plant where the process could be different from the model.

In fact, the process could be grossly different from the model then you will have more difficulties, but if it maintains the same structure but let us say if this is the only thing that changes then we could ask what happens to the performance. Now what will happen to the performance is interesting so for example, if I have let us say a closed loop now let us say I have this controller that I have designed okay I have let us say this G process so the standard control diagram we will do, I have this set point I have this negative I have this coming here going here and I have Y here.

(Refer Slide Time: 18:27)



Now, it will turn out that you design this controller based on whatever model you have assumed okay and basically what you are saying is I am going to really look at the closed look transfer function which is GPC by 1 + GPC which is G desire right. Now when I am actually designing the controller, I am replacing this GP by GM because I have to do it only with the model I cannot do it with the true process right, so I do GMC by 1 + GMC equal to G desire ok. So basically this G desire will give you some Y desire, now if this process and model are the same, the Y you get here will be exactly Y desire or this G desire and this will be the same ok.

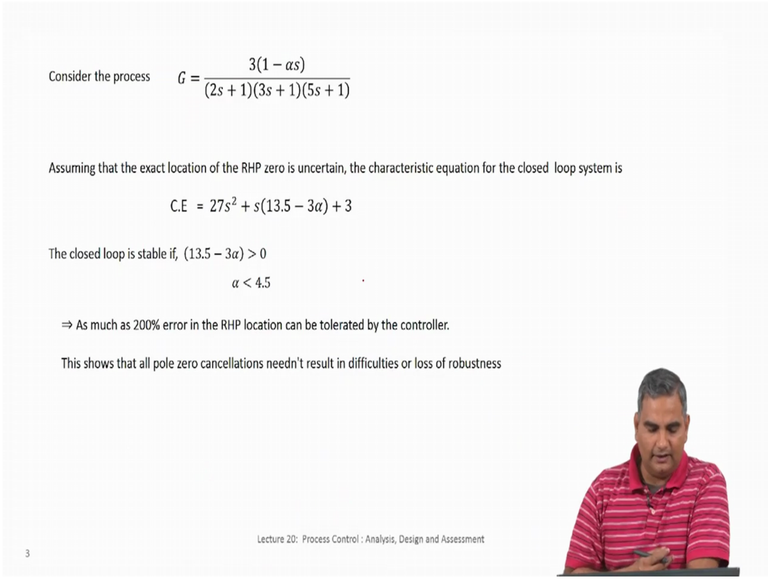
However, you design a controller based on some model and then you actually put it into a process and the process is slightly different from your model then this controller was designed using this equation but the true closed loop behaviour is basically GPC by 1 + GPC. Now G desire GMC by 1 + GMC so that need not be the same as GPC by 1 + GPC, so the true performance that you get while you implement this controller in a process is going to be given by this. This is a closed loop transfer function and this need not be exactly equal to G desire and it will in fact be equal to G desire only if GP is exactly GM, so that way this framework brings in the idea of a model a controller based on a model and then the specifications or the performance of the controller when it is actually implemented on a true process.

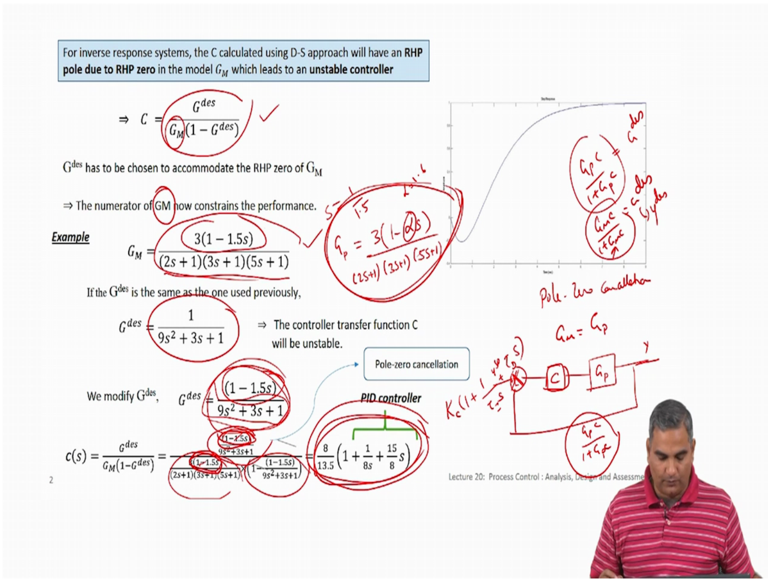
Now the true process is actually represented by a transfer functions form, so the way in theory you will use this is you could say okay this is the model but let us say that transfer function form for the process is drastically different what will happen, so those kinds of studies you can do using simulations. And ultimately the proof of all of this is when you take this controller and implement it in a true process and how well it does. And if the variety of G process models that you have tested to check the robustness of the controller times the whole garnet of you know the process operations then you will have a priory a very good idea of how your controller will work when you actually put it into real process ok.

So this is very important idea that you should understand okay, so the controller is always based on a model and the closed loop performance is based on the true process. And when we want to do all of these studies theoretically, the process model we keep it as something which can be different from the model that we used to design the controller and the G process by varying several parameters in the G process which is different from the G model we can see the robustness of the controller. So in the next slide I will show you an example where assumed that you have this is the model and then you design your controller but let us assume the process itself is slightly different, its RHP 0 is at some other location.

Now we might ask the question what will happen right, how will the performance change, I am doing a pole 0 cancellation, will I still have a stable closed loop and so on, how much uncertainty can I have in this value while I still maintain my closed loop to be stable, so these are the things that we can see, I am going to show you this to you but it is important to kind of take a step back and then think about this and understand this well so that we think about this pole 0 cancellation and we think about when they will have an effect, when they will not have an effect so here clearly I will show you it is not a big problem here but there is a case where you will see that this pole 0 cancellations could have big effect on the stability and the robustness of the closed loop ok.

(Refer Slide Time: 20:31)





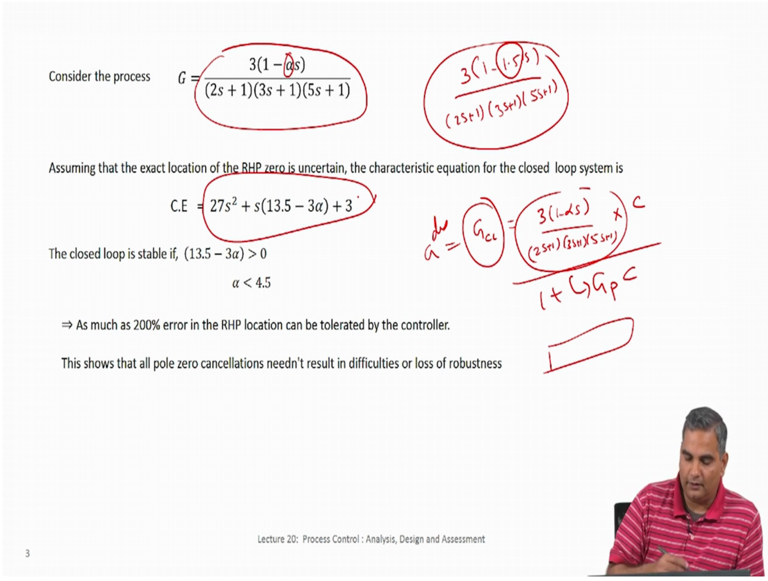
So we will get to that later but let us look at this and then look at this right here and then process which is this but let us assume we have used a model to design the controller which is 1 minus 1.5 S divided by 2 S plus 1, 3 as plus 1 and 5 S plus 1, let us say this is a model which is used to design the controller. So if I use this model to design the controller I already know the controller form which is basically this, 8 by 13.5 times 1 + 1 over 8 S + **15 by 5 uh** 15 by 8 times S ok. So now when you look at the closed loop system what you will do is the G closed loop okay is going to be equal to GPC okay, so GP has to be now 3 times 1 minus Alpha S divided by 2 S + 1 and 3 S + 1, 5 S + 1 GP times the same controller that we derived from the last time.

Now notice that this controller does not change because this is based on this model ok, so I use the same controller here and then I do this 1 + GPC, now GP is again this thing right here. Now if this GP is GM that is Alpha is 1.5 in this then you will get this to be the G desire that we have designed the controller for. But if the Alpha is different from 1.5 then you will get slightly different closed loop which will not be exactly G desire so that is where I 1st made an accommodation in the G desire for the inverse response okay so that is already incorporated and even after this accommodation I will not exactly get what I desire because of the plant model uncertainty.

So the first is actually making allowance for the difficult to control components in the model and the 2nd thing is when the actual process is slightly different from the model then you lose little more performance, so that GCL will be different from the G desire that we specified right because that difference comes because we calculate the controller using G model here but the true closed loop is G process time C divided by 1 + GPC. Now if we do all of this algebra and then come up with the closed loop transfer function and now you ask this question, so I did a pole 0 cancellation, I assumed that the 0th location is 1.5 but the true 0 location has changed to Alpha which is as said underspecified here.

So I might want to ask the question if this Alpha is slightly different from 1.5, let us say it is 1.6 would my system become unstable because I have done a pole 0 cancellation is a worry that you might have and we will see what happens here ok. So basically what we are seeing here is what is the amount of uncertainty that I can tolerate in this Alpha, how different can it be from this 1.5 while I still maintain stability is the is the question that I am going to ask. Of course we have already said that because this G closed loop uses this GP, it is not going to be equal to G desire okay so the loss in performance is already there, which is a difference between G closed loop and G desire in some sense.

(Refer Slide Time: 24:27)



So performance loss because of model plant mismatch is okay but I do not want the system to become unstable because of this plant model mismatch, so I should also judge whether there is a possibility of that happening. So to check that what we should do is GPC divided by 1 + GPC, same thing you do all this algebra and then you will get the denominator polynomial, and the denominator polynomials will turn out to be this. Now this is a quadratic and you know the quadratic root you will have minus B plus or minus root of V square - 4 AC by 2 A, and if A is positive this B term is the only thing that decides the location in terms of right half plane or left half plane.

And in this case in fact if this 13.5 - 3 Alpha is greater than 0 then we can guarantee that all the roots of this quadratic are in the left half plane because this will be positive, 27 is positive so B by 2 A is positive so the roots will be minus B by 2 A so that will be in the left half plane okay. So for this to be positive I should have 13.5-3 alpha greater than 0 so I could take 13.5 is greater than 3 Alpha that makes alpha less than 4.5. So as long as Alpha is less than 4.5 the roots of the denominator polynomial of this closed loop transfer function will all be in the left half plane and the system will be stable. So what this basically says is if this were 4.5 and I still guess this to be 1.5, I will still be stable, so there is tremendous amount of error that I can actually accommodate in the identification of the location of RHP 0.

So what is basically says if you do not have to really worry about pole 0 cancellations in this case unless of course you have grossly underestimated or overestimated the RHP locations, in other words it has to be very different from whatever value you have taken in the model ok, so this is how we look at how to choose and derive controller for inverse response system, the key idea is to enable us to design a controller we introduce some performance loss in our desire transfer function and then we were able to design a controller which turned out to be in this case a PID form. And then we have also discussed whether the pole 0 cancellation will have any impact and I showed here that you could grossly mis-estimate the RHP 0 and still stability problem will not be there.

Clearly performance comes down to 2 factors; 1st factor in actually the difficult to control dynamics which is the inverse response because of which we made some allowances in our G desire itself that is number 1. And number 2 if the G process is different from G model, the closed loop transfer function is going to be different from G desire that is the other loss of performance which is notionally the difference between GCL and G desire.

So you can see how this idea of direct synthesis allows us to clearly understand where performance limiting factors come in and how you avoid those performance limiting factors and still design a controller. The only thing is that rid of if you are going to get constraints J desire, you cannot choose any G desire you want, you still have to weigh certain rules and you are constrained and some other components are necessarily added to G desire which something you would not have added by yourself so that is something that is important to understand.

Another important thing as we go forward because you can see the slight shift in the way I start talking about controllers and control design. In the first part we did not make such a big deal about the difference between a process and model, but as soon as we came into this direct synthesis and started moving onto more advanced topics we talk more about the model and then we kind of differential between the model and the process and then we made the point the controller design is all based on notion of a model but the true effect of the controller if you want to study then in the closed loop we have to choose the process transfer function to study their effects.

So this notion of model and predicting something with the model or designing a controller based on a model is going to be a key idea that takes you from your standard undergrad controller to more sophisticated controller. So one last thing that I would light to address here is I do not want you to get the feeling that we are not using a model at all in your undergrad control, we do use some model where we use the model in designing the controller for example if you take (())(29:04) you use a model to figure out what is the limit of stability and then all your controller during parameters become a function of the model parameter in some sense.

Whereas slightly conceptual different is the controller itself having the direct model equations into its design like C is 1 over GM, G desire by 1 minus G desire is slightly different okay . So in the other case also the model parameters are very much care but they get buried into your PID controller, here it is more explicit. And as you go you will see as you go more and more advance in controlled topics it will become more and more explicit in the controller calculations themselves. So even here for example though I say the controller is 1 over GM and that is why the model explicitly comes ultimately we got a PID controller through all the parameters were used and implementation is PID.

So this is while it is little more explicit than your (())(30:04) it is not used in online controller capitation is, I mean when I say it I am talking about the models is not being used but in the next thing which we are going to talk about which is how do you control time delay system. You will see that the use of the model will be more explicit even in the controller calculation so you cannot simply live with the PID but on top of PID will see that there is some model related terms that come in the controller computation online so without the model we cannot implement the controller.

In the other case once the controller is designed we can directly implement it okay, so you are going to see that this explicit use of the model keeps increasing more and more when you go into the more advanced control topics and ultimately the kind of pinnacle of this is actually using the model in the online optimisation framework to do computation which is what model predictive controller is which we are going to talk about. So that is also nice progression in terms of ideas that I would like you guys to think about, so I will get back in the next lecture to talk about time delay system and what are the difficulties the time delay system introduced and how do we control time delay system, thank you.