Optimal Control, Guidance and Estimation

Prof. Radhakant Padhi

Department of Aerospace Engineering

Indian Institute of Science, Bangalore

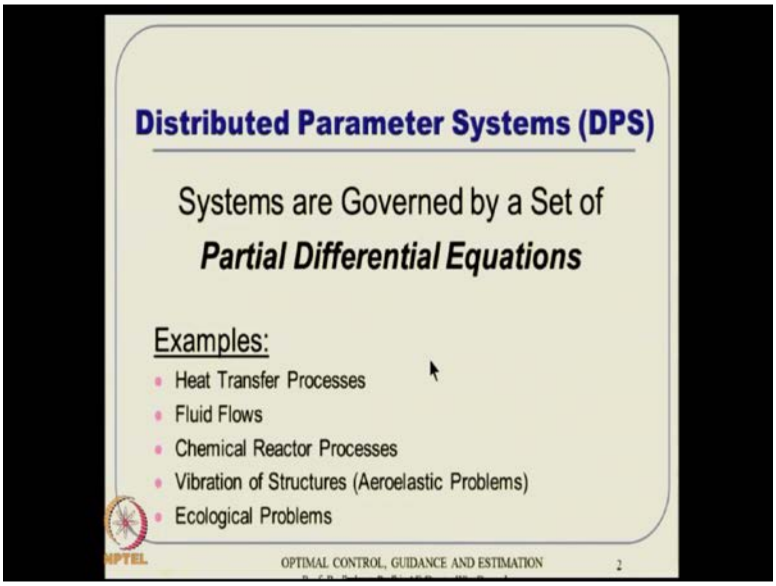
Lecture No. # 37

Optimal Control of Distributed Parameter Systems – I

Hello everybody. We will continue our lectures on this optimal control guidance and estimation course. So far, we have covered a variety of topics having both theory and applications as well. Towards the end of the course, I thought it is also a good idea to see a slightly different, it is very connected, but slightly extension of the topic in a different domain which is called distributed parameter system.

So, I thought, in two lectures I will cover some topics on optimal control of distributed parameters system. And especially, the research that we did and the results that we got some time there, and things like that actually.

(Refer Slide Time: 00:58)



So, let us get started. This is the topic what we want to discuss in these two lectures. This is something called distributed parameters systems. Essentially, we see these are the systems governed by a set of partial differential equations. And so far, we have not talked anything about that. We have talked everything about system dynamics in ordinary differential equations in state phase form. So, that means, these are the ordinary differential equations represent something called lumped parameters system; whereas, this partial differential equations are PDEs. They represent this distributive parameter system. And such class of system arises very naturally in a different variety of application problems.

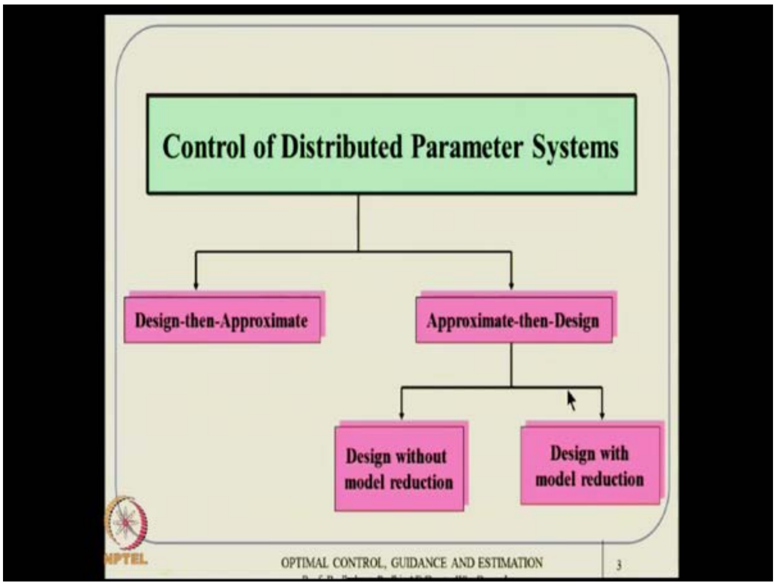
First thing is, let us say we talk about heat transfer process. You can talk about something like a lumped parameter behavior of heat transfer process. By the way, what is lumped parameter system? It talks about something like a system where there is no relative dynamics between different molecules of the system. That means, if one particle goes everybody goes together and things like that. They do not relatively disperse out, they do not, I mean there is no relative dynamics between the two molecules of the system. That is lumped parameter system. Whereas, the distributed parameter systems, that has a spatial behavior also in addition to temporal behaviors, that time evaluation, it also has spatial behavior and hence we need more than one independent variable to describe the system dynamics. That is why we need P D Es.

And then, the example problems include heat transfer process. The moment there is heat somewhere it does not, it has to dissipate every everywhere in all directions. So, that has to be talked about that sense. Then the fluid flow problems, both in liquid as well as gas form. And then we have these chemical reactor processes where this chemical reaction takes place throughout the reactor basically. So, that the dynamics of that is typically governed by partial differential equations.

And there is something like vibration of structures. And in aerospace may be aero elastic problems, like that. So, we see, there are various problems including probably ecological problems also. Sometimes, the behavior of these small animals in the jungle, in search of the pray mate all sort of things. They go in different directions. And then, if we really want to see the dynamics of this entire behavior, of this the entire colony, then probably we need P D E, partial differential equations also.

If you see, there are varieties of problems that cannot be kind of categorized as long parameter system. We have to talk about system dynamics in P D Es only. Now the question is, how do we control the systems? So, in how do we optimally control the system, these are the questions that arise naturally and then there are techniques available also. And we will see one or two techniques as we go along in these two lectures.

(Refer Slide Time: 03:58)



And hence, when somebody talks about control of distributed parameters systems, two things comes into mind. Remember, these are something called infinite dimensional system also. Means, if you see really want to have the behavior of all molecules of the systems independently, let us say in some sense, through constant equation for the relative behavior. But each of the molecules will take their independent dynamics, then, obviously, we have infinite molecules in a system. And hence it is an infinite system, infinite dimensional system.

And if you really want to kind of approximate it through sum set of approximate lumped parameters system. Then through this model of behavior first mode, second mode, third mode, things like that, then also we need an infinite number of modes actually. So, in that sense also it is an infinite dimensional system.

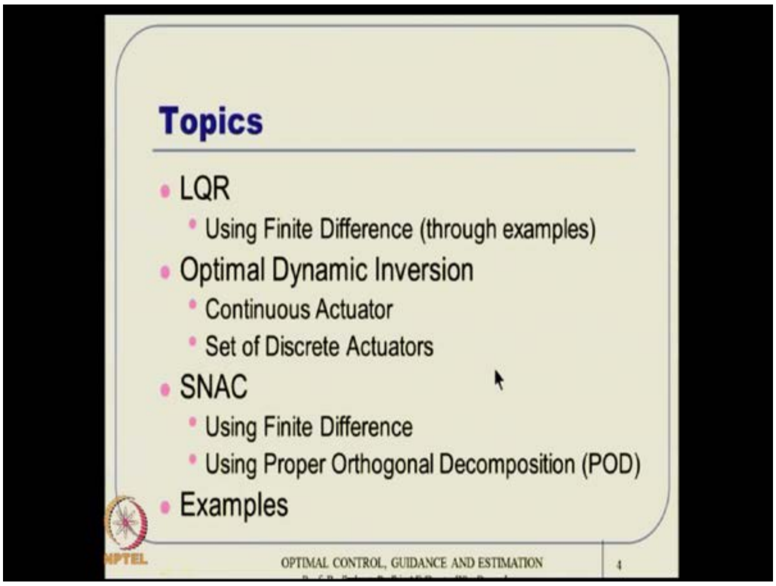
So, that cannot be done. I mean, that cannot be handled. So, there are ideas like, something like, design and then approximate. That means, you invoke this infinite dimensional operator theory and then do all sort of algebra math. And finally, come up with a infinite dimensional controller also. And, obviously, you can implement that or that the feedback necessity is also infinite dimensional. So then you cannot implement it that way.

So finally, we have to approximate. So, first you design and then you approximate. On the other end, naturally the other thing comes, approximate and then design. So, this approach what you see is Design-then-Approximate. It is typically favored by math people. The mathematics people will typically like to have this, whereas, engineering people will typically like to have this.

Theoretically, this is more elegant because they do not talk about throwing away behavior to begin with at all. And the implementation is subjected to only your computational power and things like that. If the implementation, practical difficulty your reason is and computational power. Whereas, this one, the system dynamics itself is truncated to begin with and hence there is an approximation to begin with. But anyway, so this is typically favored by engineering people, and then this itself can be divided into two ways. One is design with model deduction design without model deduction. Means, you design some sort of a, you put a large uniform finite blade sort of thing. And then, do not do any model deductions. Just directly use that and go ahead and design the controller. That is one way of doing this. But then it also requires a large number of rid points to do the problem. To solve the problems in a satisfactory way and hence this is also not good in general.

So, what people think is, we will approximate, but then introduce these model deduction concepts also, so that we have to handle this low dimensional lumped parameters approximation of the system dynamics. And then carry out the engineering design or the control design based on that reduced order model actually. So, these are the various categories that comes to mind. And we will typically see this side of the story, engineering side of the story. Both design without model deduction and model deduction also.

(Refer Slide Time: 07:04)



So, this is how it is and the topics that these two lectures typically will talk about L Q R design. So, if the system dynamics happens to be linear; that means, if the P D E itself is linear, then it will result in approximate lumped parameter system which is also linear. And cost function is quadratic, then it also results in a final dimensional cost function which is also quadratic. So, that will have a compatible L Q R formulation. And because in a L Q R formulation, we can go ahead and implement it. And solve it and implement through Riccati equation.

So, we will not talk about philosophy here. There you know the philosophy. So, we will just demonstrate this idea through some examples. Just one example I will take. So then, we will have some, I mean, your ideas will be clear actually. And then we will diagnose slightly little bit, I think it is a slight diagnose in some sense, but it is over-lapping with optimal control in a way.

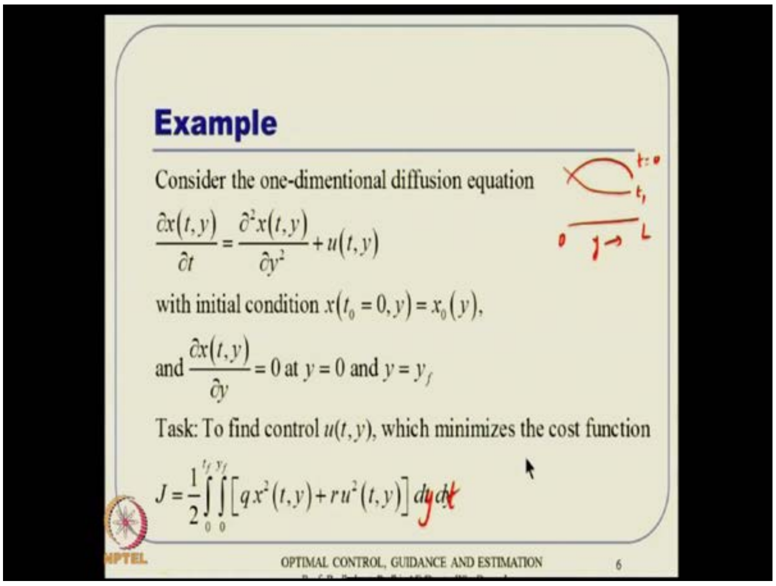
So, here I will talk about something called optimal dynamic inversion. Again, that is something that, we proposed some time back. So, that is not really optimal control, but there is optimality expects that comes on the way, and variational calculus is also used on the way.

So, these two we will see. Especially, this lecture will cover these two. And in the next lecture, we will come back to this single network adoptic critic. That is something; we talked about that sometime back. And in that frame work, we would like to solve this problem for optimal control of non-linearity symmetry parameter systems. Non-linear, I mean, P D E sort of cost function. I mean, system dynamics and a necessary cost function using a directly, like finite difference or using this model deduction ideas. Like proper orthogonal decomposition, and then just synthesize a controller. And then obviously, on the way we will give examples also.

So, in this lecture, the first two topics and next lecture I will cover this S N A C based optimal control design. So, next we have started. First thing is optimal control of linear distributed parameter systems using L Q R and finite difference. This is a very simplistic idea that comes to mind and probably that is the... I mean, more than that amount, it may not happen.

So, let us just have a small demonstration of the idea through an example. And through that I think everything will be clear to you.

(Refer Slide Time: 09:35)



So, let us take a simple linear P D E in this form. Del x by del t is del square x by del y square plus u of t y. So, this x is the state. But it is a scalar problem. x is a single state problem. I mean, x is a single state. However, it is a function of both t and y. So, t is the time part, how it evolves in time. And so if you have something like y, y varies from let us say, 0 to l. That is the y part of it. So, x is there everywhere along y. But then the behavior of that, suppose you want to take t equal to 0, then something it will like this.

This is probably t equal to 0. Then t equal to, let us say little bit later t plus delta t. This is something like, t 1 let us say. It can be different different things.

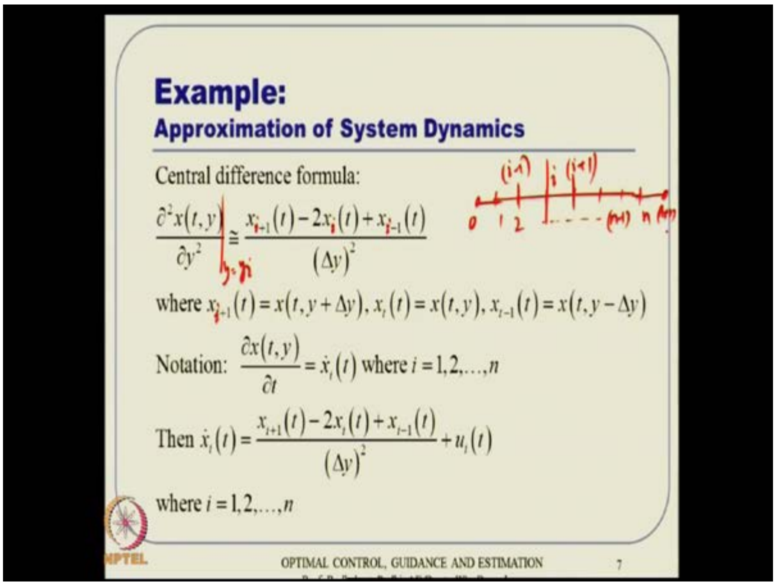
So, this is the type of problem that we are talking. And we are also assuming that control is available throughout the domain, 0 to L. The control is a continuous control (( )). Typically, that is one of the difficulties of distributed parameter systems.

We cannot have a continuous control x n throughout the domain. That is, implementation says, it is not possible in general. All right, so, we have a system dynamics of this del x by del t equal to del square x by del y square plus u of t y. Still it is a linear differential equation linear P D Es. And as initial condition when t equal to 0, we take some profile. Remember t equal 0, it has, it cannot be just a number. It has to be a profile across y. So, something like that. And the boundary conditions must also be given in P D Es.

So, at y equal to 0 and y equal to f what we are assuming is that the derivative is 0 for special derivative 0. And there is something called, thing in a normal boundary condition. Now, the task is to find a control, u of t y which minimizes the cost function this way. Again the quadratic cost function q and r are scalars essentially here. And then, this is the, remember this is a double integral here. Integrated with respect to y and then also integrated with respect to time.

So, this is the thing that we want to minimize. Well, ideally speaking it should be y first and t next. On this, either way mathematically both are equivalent anyway.

(Refer Slide Time: 12:28)



So, this is what it is actually. Now, how do we go about it. The very simplistic idea is, well, I will do this spatial discretization, let us say. I will do this spatial discretization. There is some grid points I will select. 1, 2 like that up to n minus 1 and n. I will select to that way. (( )) n minus 1 here and n here.

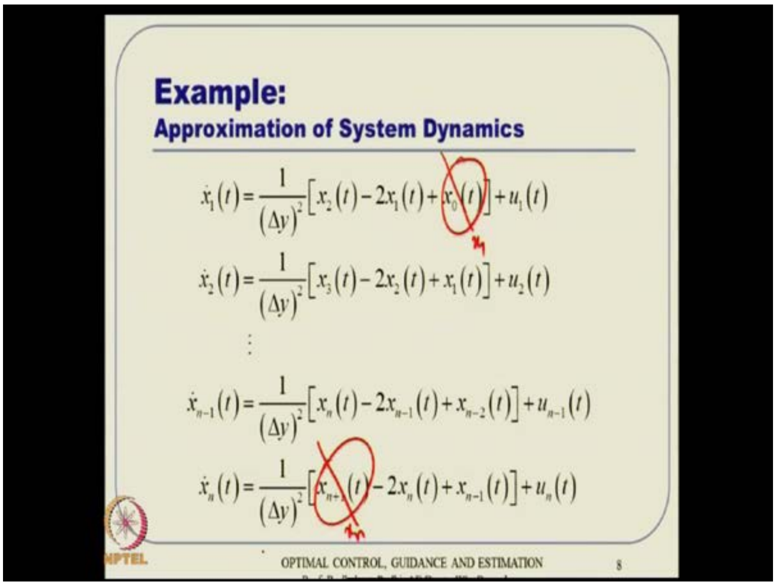
Now, what does it tell us, suppose I want to have this at any point of time, I fixed node, let us say, i. And that, at point i, I want to see this partial differential equation, I mean this partial derivative. How do I approximate that? I will approximate that using, a central difference formula. And central difference formula tells me that at any point of y equal to y i, these are all i's basically, this i then i, i.

All right, so, I will take 1 plus here, 1 minus here and this is central difference formula. We know it very clearly, this is i plus 1 and this is i minus 1. So, this can be, very easily seen. At any grid point i, I can approximate that in this way.

These are all print mistakes actually. x i plus 1. Anyway, so now we include this notation. del x by del t is x i dot, because after we discretize there is no for special derivatives for say. And these are all node point values now. And then we can talk about something like partial derivative with respect to time is x i dot.

So, if you do that, then what it turns out, that x i dot, I can represent coming one part, one portion from here, and then u i. coming from system dynamics here. So, then if I repeat this exercise for 1 to n, what we get everywhere, I will put this kind of formula. Then what will I get?

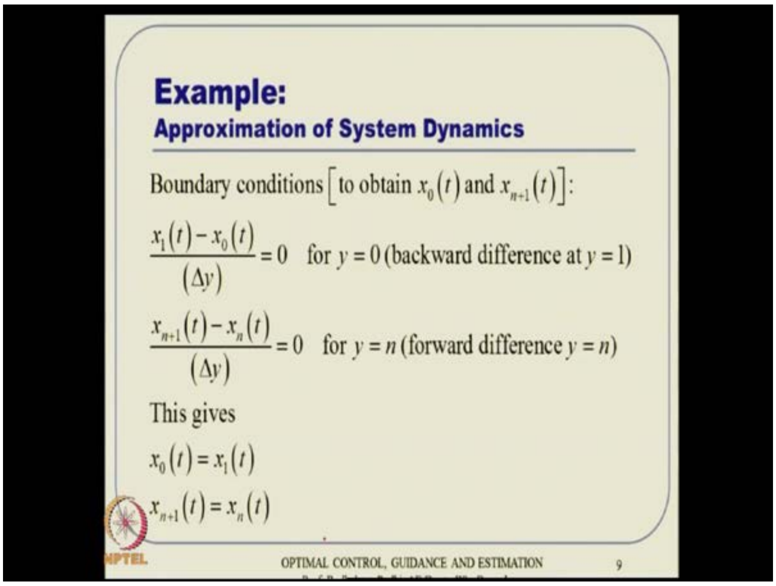
(Refer Slide Time: 15:07)



I will get something like x 1 dot, x 2 dot, x 3 dot up to x n dot. The thing to note here is something like, I mean, everything is alright, but other than these two. These two values are outside x 0 and x n plus 1 when you see this, we have a grid point from 1 to n, 0 is not there. So, what we do is, we will approximate that there is a fictitious node point we will take, all that is 0. Extend that little bit. And here, another fictitious node point. Call that as n plus 1.

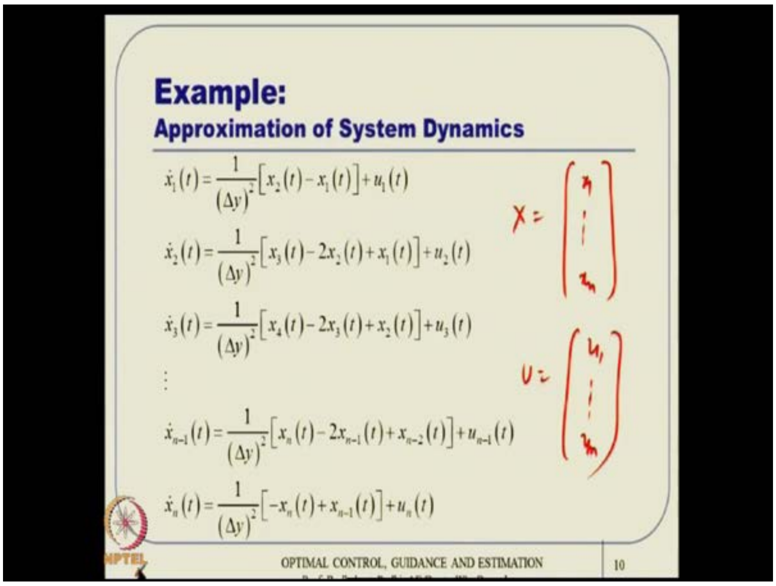
So, that is necessary for writing the system dynamics at grid point 1 and n. And the question is how do we get a value for that? These are fictitious node points anyway. This is where the boundary condition comes very handy.

(Refer Slide Time: 15:58)



And then, we tell at y equal to 0, I will use something like backward difference at y equal to... So, for y equal to 0, I will use backward difference at y equal to 1. Here I will use backward difference at y equal to 1. And then here, I will use forward difference at y equal to n. So that, then I will get these boundary conditions implemented this way. So, this will tell me that because this is 0, then x 1 has to be x 0 and x n has to be x, I mean x n plus 1 has to be x, so then if you go back and substitute this. This is nothing but this is from boundary condition. What we got is nothing but x 1. And this is nothing but something like x n. So now, we can simplify the first and last equation and then combine them together.

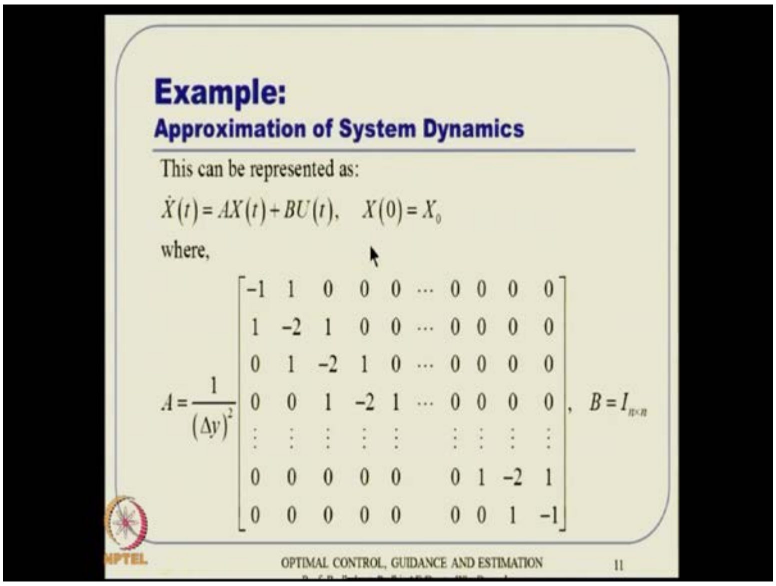
(Refer Slide Time: 16:52)



Then it results that first equation and last equation. Typically, first equation happens to be this way. Last equation happens to be that way. But all other things will have a uniform structure.

So now, you can clearly see that if I really want to combine everything, I can combine it this way, because now I can define a state vector and a control vector also right? I mean, I can define a state vector with my grid point values x 1, x 2 to up to x n. And then I can define a control vector which has same grid point values. Say, something like u n.

(Refer Slide Time: 17:32)



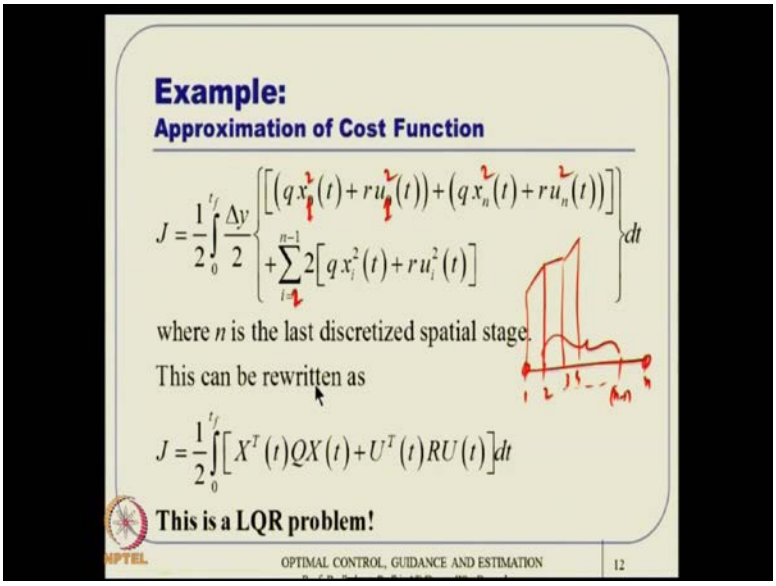
So, if I define these two, then I, with respect to these two variables, can write it this way. x dot is equal to a x plus b u. For a n, we can we define it that way. So, we have got an equivalent representation of the P D E in terms of O D Es, with respect to the grid point values. So, the more the number of grid points, the better the system dynamics representation. But beyond a certain point it may not be necessary also. And also, it turns out that it not only depends on the system behavior system dynamics, it also depends on the initial condition. How many grid points you need and things like that, that way. We will give a small example. That will make your idea clear.

Now, this is also very, I mean, somebody interested here, you can quickly evaluate the Eigen values of this matrix a. And you can realize that this is actually a stable system behavior, because this what system dynamics that we started with is a dissipative system. It represents a dissipative system. So, obviously, it is a system dynamics that will represent in stable behavior of this system dynamics and lumped parameter system. That means, the Eigen values of A will all remain in the left hand side. You can quickly verify it.

Anyway, in system dynamics we have got some representation in O D Es. How about cost function? And that is also easier, if in the cost function all that you have to do is just use trapezoidal rule, let us say. We have this cost function to begin with.

So, we use this spatial this integral. Whatever you get, the spatial part of it. This spatial integral will use trapezoidal rule and trapezoidal rule will give us this one.

(Refer Slide Time: 19:04)



This is like, trapezoidal rule tells us that, is nothing but delta y by 2, h by 2 into first node values and last node values. And everything else should be multiplied with two. Half h into a plus b plus all these a 1, I mean, a 1, a 2 like that, that way. So, you can revise this trapezoidal rule. We have also talked about that in our one of our earlier classes. Review of numerical method. I think, lecture number three probably. You can look at it.

Anyway, so, using trapezoidal rule, you can put it back. And then, it results in an equivalent cost function in this form. Now, combining the values, I mean variables and all that. And remember this 0. Forever you get a 0 that represents, the x 0 represents is equal to x 1. And x n plus 1 represents n and things like that.

So, details you can see or you can derive it yourself. But ultimately it happens that you can write the cost function in this form. Probably, this is a small thing here. I think, if the fictitious nodes are not required here, it is the directly node point. This is 1 and 1 and n. And probably this will start from 2. Then it is all right probably.

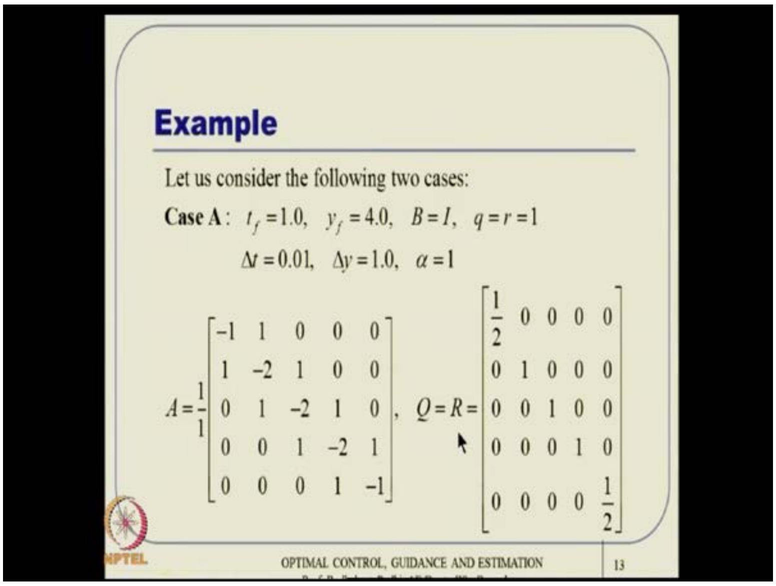
So, if you see this in a picture. This dynamics only requires a grid point. But evaluation of integrals, this starts from 1, 2 all the way up to n minus 1 and n. So, the first grid point and last grid point are taken as it is. So, this is 1 and this is 1. And this is square also, by the way. So, these terms and then all these things will be multiplied by this 2, because it essentially talks about area evaluation and things like that. So, this will talk about that and then this way 3 and then this 4, like that.

So, essentially it evaluates the area under the curve and things like that. That is why it happens like that. So, coming back to that, this is 2. So, from 2 to n minus 1, it is twice the same value, I mean, the integral integrant value. And first and last point, we do not worry about it. Now, all these things can be again arranged, rearranged rather. And then you can try to represent in this form.

So, once you represent in this form. And the system dynamics can be represented that way, obviously it formulates an L Q R problem. System dynamics is linear and cost function is quadratic. So, this is an L Q R problem and we know how to solve it.

So, that is one way of doing that. And then you can see what it gives. It gives solution at the grid point values only. It does not give anywhere else. That is alright, still we get some idea about what is happening at the grid point.

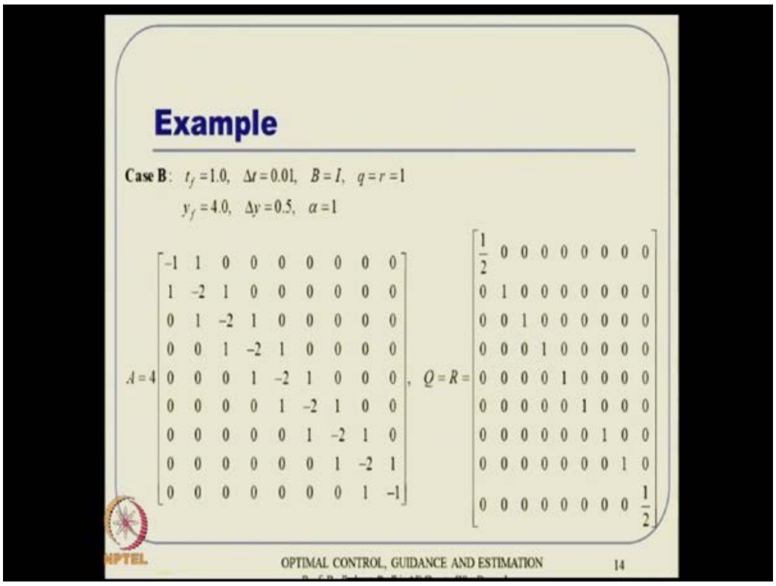
(Refer Slide Time: 22:13)



An example problem, if you take a finite time, L Q R problem also requires either t f is infinity or t f is finite time. You remember that. So, this example, this solution was the results that we are getting, I mean, that we have included, is with respect to final time - finite final time. So, we have this t f is 1, y f 4 and all these values, numerical values are there. So, I suggest that you verify these matrices that you are getting here. And then you code it. And then, using MATLAB of L Q R function you can actually recover the results yourself.

So, we have got the A matrix, we have got the B identity and we got q and r this way. So, we incorporate L Q R function, and solve it.

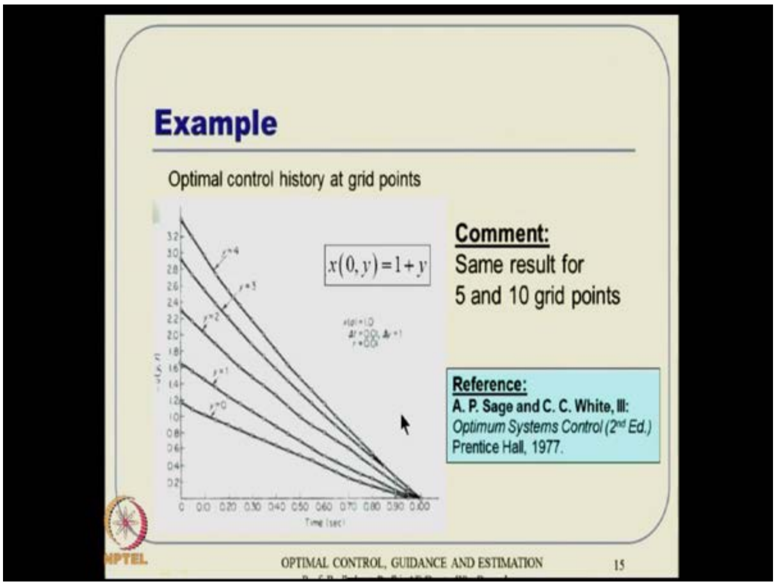
(Refer Slide Time: 22:59)



So, the what remember MATLAB L Q R is give towards infinite time. Here we are talking about finite time L Q R. So, probably we have to do a little bit coding to get it then. So, you know this final s f equal to 0; in this case, because the outside thing is not there.

So, p p f is equal to 0, but p dot is that Riccati equation. And then you have to integrate it backwards, from 1 to 0, and then implement it forward like that. We do all that and then the case b we have taken a slightly different way. Case one, you can see the dimensionalities increasing. That means, the number of nodes are increased. So, here the delta y is 1 and there the delta y is 0.5. Now, let us see what difference you get.

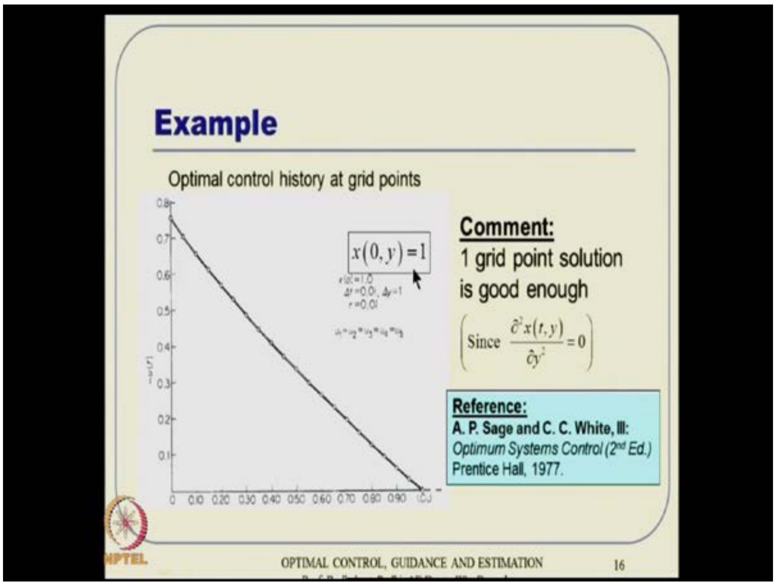
(Refer Slide Time: 23:50)



Now, surprisingly, there is no difference. Either you take it as one or I mean, either 5 grid points or 10 grid points does not matter. This is that you have taken, I mean what you see is taken from here. This particular book, where some of the concepts in this lecture, I have taken from this book.

So, this particular sort of behavior you can observe. And it really does not matter whether you have 5 grid points or 10 grid points. Both will be the same. And that it happens because this is a linear profile here. For which the special derivative turns out to be 0. If you take del square x by del y square, then this is nothing but 0. So, initially, as long as you have a linear profile as initial conditions, this kind of behavior is expected.

(Refer Slide Time: 24:39)



In fact, if you have a constant initial condition, even 1 grid point will do the job. That is why it is. Just the initial condition is constant everywhere. Then if you take 10 grid points, solve it, 5 grid points solve it that does not matter.

So again, I mean this depends on that kind of thing. But when you get a solution, you do not want to confine yourself to any particular kind of initial condition or initial condition profile. You let it be arbitrary, and then you may need a large number of grid points to get a reasonably good solution.

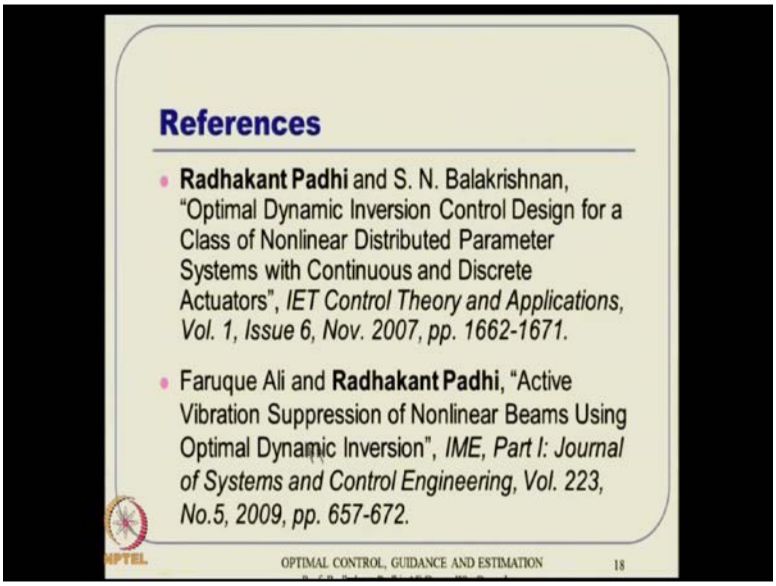
The whole idea is, start with course grid point. Let us say 5 grid points like this. And then go to 10, then go to 15, 20 like that. And some where you will see that the results are not different. That we can say that, that is the final grid point I will see late and then carry on.

And the typical example in a real life example problem, you will probably require thousands in thousands of grid points. I mean, that is the drawback of that. And hence, this dimension of these matrices will blow up, land up with high, very high dimensional representation O D representation of the PD E phenomenon.

So, that is the problem there. Essentially, it outlines the idea that, we given a distributed parameter system; system dynamics in the form of PDE. We can actually do it. This now, it is a question of how to do it better, basically that way.

So, rest of the time on this lecture, let me digressive slightly different topic. This is something that I want to discuss as in the frame work of what we called as optimal dynamic inversion, it is not really optimal control. It is largely into the, it will fall in to the frame work of dynamic inversion rather. But there is an optimality concept included in that and hence there is a weakling for optimal control also basically.

(Refer Slide Time: 26:24)



Largely, the material is taken from here and then especially the very first one which happened appeared in 2007. And following subsequently, have extended the work in different class of problems in a real, little more real life problem of vibration suppression of non-linear beams. It is essentially a structural control problem using this approach.

This particular thing lays down the theory lies, lays down the ideas and then solves an example problem that I will talk today. But this also you can see very similar concept. But it is used for a different problem. Name of that, the difference is, now if you talk about something like heat dissipation or liq flu. I mean, something like stable system dynamics fluid flow or heat dissipation, like that. These problems are relatively rather easier. They are something like, some parabolic distributes, I mean parabolic differential equation.

But when you go to vibration, these are typically, I think elliptic or hyperbolic differential equation, they do not have, they do not die out, so easily I mean, they will continue to vibrate until there is a, unless there is a good amount of dumping and all that.

So, in that sense it becomes a little more challenging problem. Anyway, we are bothered about the ideas here. So, we will concentrate on the first one. So, in this one, there are two different things here. One is using continues actuator. This is more theoretically elegance, and then using discrete actuator also which is practically more relevant.

So, if you have guarantee of continuous actuator that may (( )) anything that you can do throughout the special domain very next to it, something you can do, very next to it something you can do like that. You have distributed actuators, then you can do many good things. And that those are basically theoretically more elegant. So this is, first we will start with problems like that with continuous actuators.

(Refer Slide Time: 28:26)

The motivation is something like this. Quickly, to summarize, distributed parameter systems are usually difficult to control in general. Existing techniques: Design-Then-Approximate or Approximate-Then-Design.

And then this is the relatively new technique. Compared to what you see here, this is what is available. Approx Design-Then-Approximate is functional analysis approach. This is typically, mathematics people will want it. I would like to see that. And then, this Approximate-Then-Design is typically this spatial discretization. That is what we just talked about through an example problem. And then there are other ideas like Galerkin Projection Approach through this model analysis and all that.

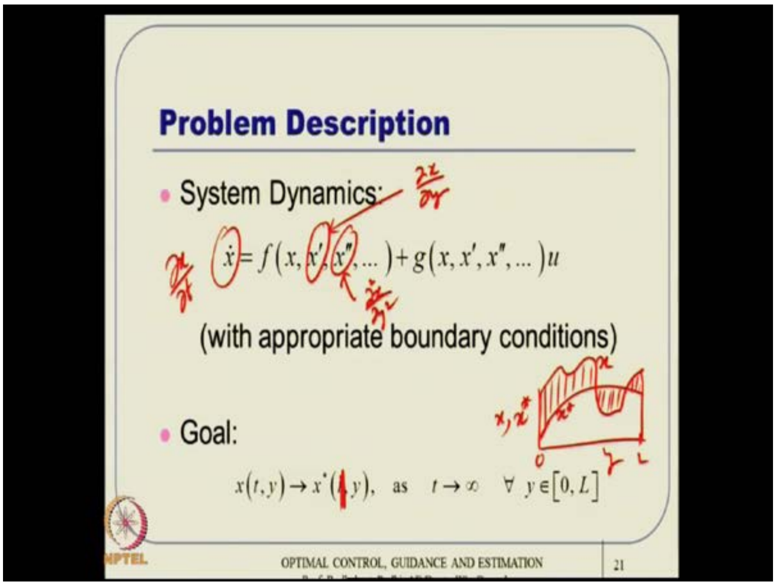
And this model analysis can be extended possibly through the P O D basis functions and things like that which we are going to talk about in the next class. Proper orthogonality composition like that, where if there is a the idea of model deduction.

So, like this if you can bring in, then this is Approximate-Then-Design. But here is a technique what we want to clean is something like, falls info the D-T-A category, but without math complexity. Typically, this is mathematically complex analysis and hence it is not very much appreciated in engineering domain.

But here is a D-T-A approach - Design-Then-Approximate category without too much of math complexity. The whole idea here is use something like dynamic inversion. Again, this details about dynamic inversion. Somebody who is interested, you can see the lectures in my other course and NPTEL problems. They are there.

Then here, we bring some variational optimization theory, and that is what is used in optimal control. So, we put it back together to propose a new technique. And then we can also claying the steady state convergence can also be proved as long as we have this, this continuous actuators sort of thing.

(Refer Slide Time: 30:22)



All right, let gets started to simplify things, we want this system dynamics in this form. Again, it is a kind of control defines form. So, x dot represents, del x by del t. It is not really something like lumped parameter system. This one is something like del x by del t.

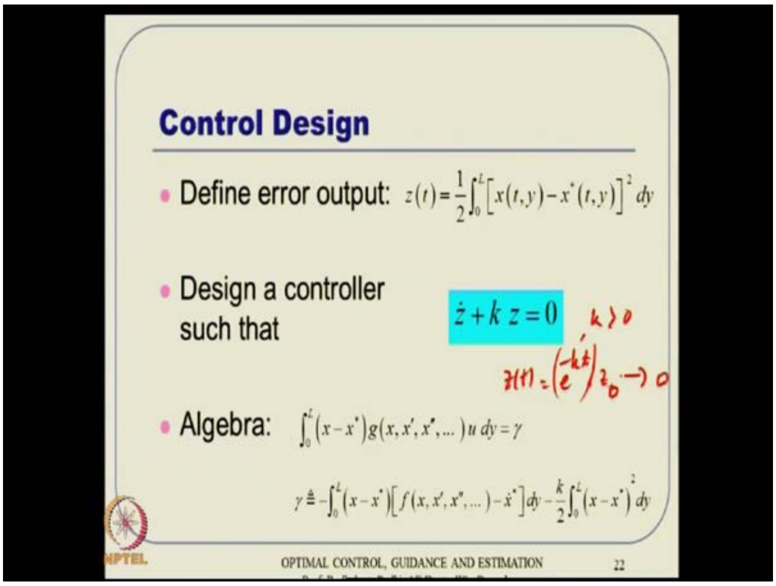
And when you see the x prime like this, this is something like, del x by del y, when you see this one, this is something like del square x by del y square.

So, these are the notations that are used here. And obviously, we assumed that there are appropriate boundary conditions available depends on the situation. Sometimes value is fixed, sometimes derivate is fixed. And one end can be value fixed, other end can be derivative fixed depending on the application.

And objective is that, x of (t, y) to go to something like x star of (t, y), as t goes to infinity; that means, this x star of... Well, theoretically speaking, this t does not mean too much of sense because t goes to infinity already. So, we have something like x star of y. Let us say, I want it to happen something like this actually. y goes from 0 to L.

And I want these to be a behave like, this is something my y star. Sorry, x star is a function of y. So, this my y, and this my x star let us say. So, as time evolves, any arbitrary recell profile should finally combine conversial. That is the ultimate objective.

(Refer Slide Time: 32:11).



So, how do you do that. First of all, you define an error. Error is an integral value we integrated all over the spatial domain. So, x minus x star, whatever is the error. Suppose this is not the case, but your x is somewhere different. Something I do not know what form. It is something like this. This is my x, this is my x star. Then what you really see is this difference. What you see this difference, it has to go to 0 with time.

So, how do you make sure that the difference goes to 0. First you define something like error quantity, z of t is integrated square error over spatial domain. And then if this goes to 0, remember, everywhere it will go to 0, because this is a quadratic function. If some of z is pulled to 0, then everywhere else this error will go to 0. And hence the objective will be met. Whatever object you are talking about.

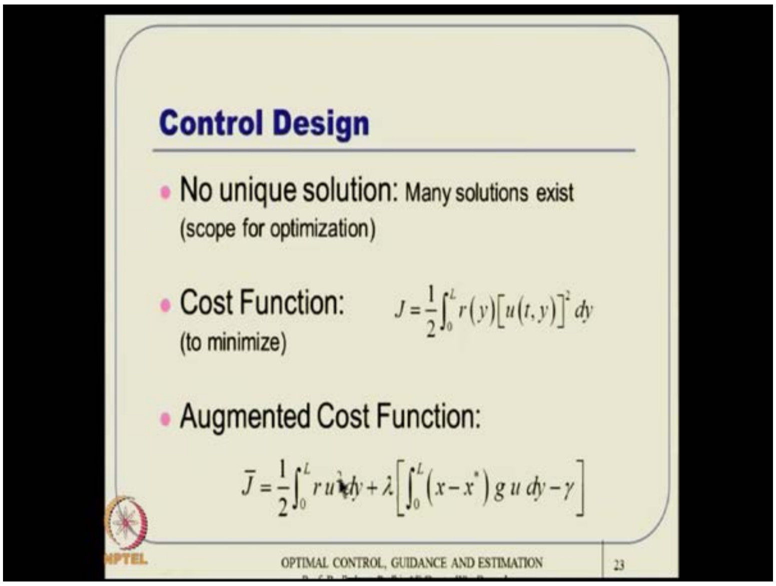
Anyway, so design a controller such that, so what is the idea of I mean, following the idea of dynamic inversion, till we will design a controller such that this error dynamics will be satisfied. k is a positive definite matrix in general or if it is a scalar it is a positive number. k is a positive definite matrix in general, depending on how many terms are there in that.

But, just to make sure that ideas are simple, you tell this is scalar. So, k is a positive number. So, we enforce this error dynamics so that, the solution of these, obviously, is e to the power minus k t into z naught. Initial condition is z naught. So this, obviously, will go to 0 because of the exponential term that is sitting here.

So, that is why we want to enforce this error dynamics here. Now, it is a, rest of the things typically math. So, from this equation, you put the definition of z. z is like that. z dot, which is del z by del t. And remember, this is a spatial integral. What we are talking about is a time derivative. So, this time derivative can be pushed inside the integral and then you can put. Here is x dot. And x dot can be brought in that all the system dynamics that we know here and carry out the subsequent algebra. And it will ultimately result in this kind of an equation. It will give us something like this, where gamma is defined something like that.

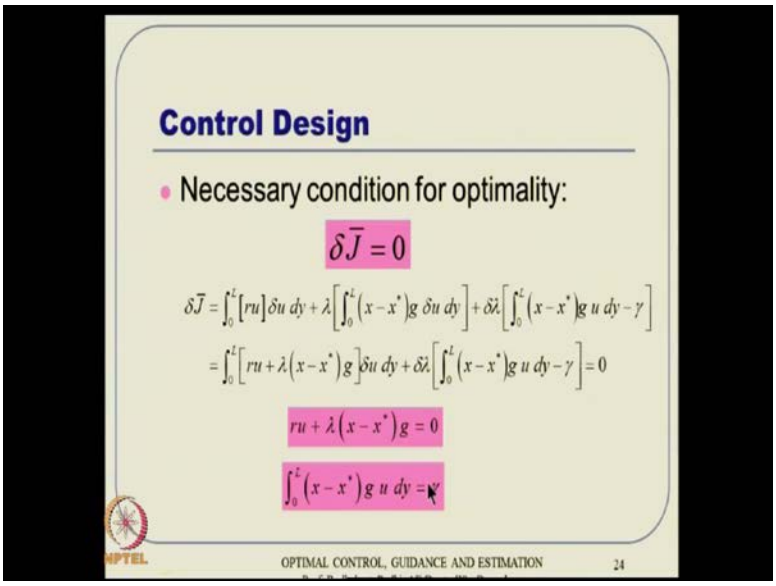
Now, the question here is, we have a control to solve from this equation here. And if you really see that this equation, what you see in the right hand side, is a scalar. But what is u? u is a controller that acts throughout, u is something like a profile. That means, if you see in a grid points sense; that is actually infinite number of points. That means, this equation what you see here, if you really want to solve for u, it turns out it is a very highly under constrained problem. Then you have to do some further algebra to get a solution for u, because there is no unique solution really.

(Refer Slide Time: 35:14)



So now, what you tell, if you can do that, there are many solutions. Then we want one solution which is control minimizing also. We do not want kind of keep on getting some solutions arbitrarily. We want that particular solution which will be minimum magnitude. So obviously, the idea here is to minimize this cost function. So, we want to minimize these cost function, subject to this constrained equation. Then you can have this augmented cost function coming from this part. I mean, this r times u square coming here, plus lambda times all that.

(Refer Slide Time: 35:53)

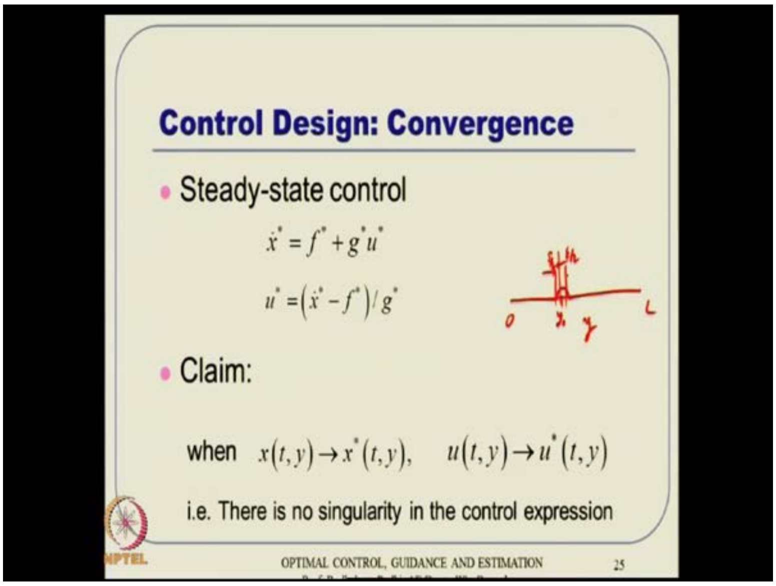


And then, subsequently, the necessary condition will tell you that first variation has to go to 0. Then there is the first variation. This is where the variational calculation is seen. The first variation, we have to expand based on this. This expression of J bar, do it carefully. You can derive it yourself also term by term you have to do, and then this has to be equal to 0.

Now, if this has to be equal to 0, this expression. And we know that variations cannot go to 0. That means, del u and del lambda cannot go to 0, because they have to satisfy this for arbitrarily taken, arbitrarily large number of variations this conditions has to be satisfied. So, hence the variation themselves cannot be 0, and hence if you excite the calculus of variation fundamental result. If such a situation happens, then the coefficients have to go to 0 basically.

We have talked about that theorem in one of the early classes. So, this has to go to 0. So, that mean this equation we will get it here. The various, this del u cannot go to 0. So, rest of the things will go to 0. So, that is how we will get that. No, no sorry that is how you will get this coefficient, this coefficient has to be at 0. And then, that is how we will get. So, from here, what you get is, this integral. This quantity, up to this, has to be equal to gamma, this cannot go to 0. So, the coefficient has to be 0. What you see in square bracket, has to go to 0. What you see in square bracket has to go to 0, like that.

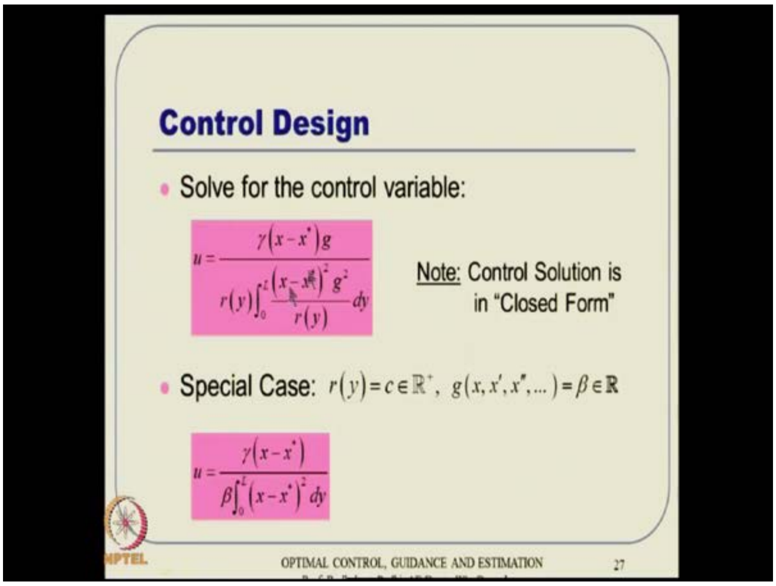
(Refer Slide Time: 37:25)



Once you put that, then the, I mean, you got a solution ready in a way, but because you have to solve this two equations for lambda and u. And then you get the solution. I will give the expression in a moment. But on the steady state what happens? When the steady state is reached already, then this has to be like this. This has to satisfy the same system dynamics. Otherwise, if the profiles do not satisfy the system dynamics, naturally then there will be deviations.

So obviously that does not you do not want to happen, so obviously this condition has to be satisfied. And the claim here is, when x goes to x star, u should go to u star also. Then only it can be maintained there. The error can be maintained it 0 basically. So, from this, whatever solution we will get it, goes to 0.

(Refer Slide Time: 38:07)



And the solution turns out to be something like this this form, actually and in a special case, if you solve these two equations. And you can see the details in the paper. If you are unable to see that, and then you can come up with this solution, and another very special case, where r of y is just a constant. You do not give differential emphasis of controller values, magnitudes through the special domain. Then it will result in somewhat little more simplified expression where g turns out to be a constant quantity and r turns out to be constant quantity as well. Then it will result in something like this. So, you can note that the control solution is actually happening in closed form. So, that is good advantage.

Now, the claim here is, if you get a steady state control like this. And you have a time varying control like this, whatever either this way or that way. Then and, I mean, as these profile develops towards x star x, x goes to x star. Then u should also go to u star; that means, there is no singularity in the control expression. Remember, this is not singular, as long as g star is not singular.

And obviously, system dynamics will be well behaved. Otherwise, the system dynamics itself is not well behaved. Modeling has some problem in general. So, if we are assuming that this expression has no problem, then you will go to that expression. And hence, there is no problem.

(Refer Slide Time: 39:34)

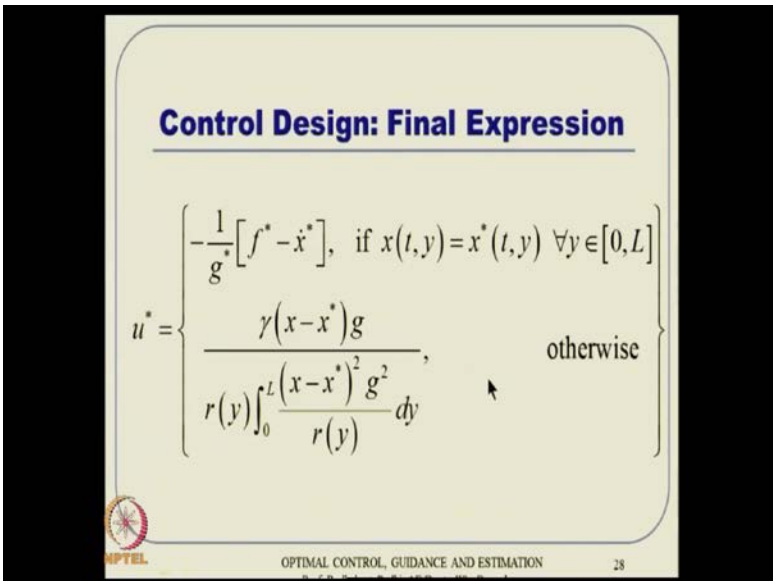
The behavior is,.. The proof is also given here. You can tell that at a very special case, what do you do is, is a very special case. You take a 0 to L in y dimension. Tell everywhere else it is 0 or there is a small domain, and around which it is not 0. Let that be something like that y naught.

So then, around that y naught, what is happening and all that, this analysis you can carry out. I think I will not go through the details here. It is not work probably. But you can see the previous and then carryout the algebra yourself. And then realize that ultimately this u bar which is the everest control in that small domain. Whatever which we are coming, as long as this is something like, the philosophy is like this. You construct a y naught.

And construct to, I mean your analyzing something like, epsilon y 2 this side and epsilon y 2 the other right side.

So, what will happen in this situation? And ultimately come up with some expression. And the claim here is, when epsilon tends to 0, then x will turn to x star obviously. And then, you will also, this u bar which is everest control over this domain will also approach to that this expression u star. And when epsilon, it tends to 0, obviously, it will merge there. That is the whole idea there. Details you can see in a paper.

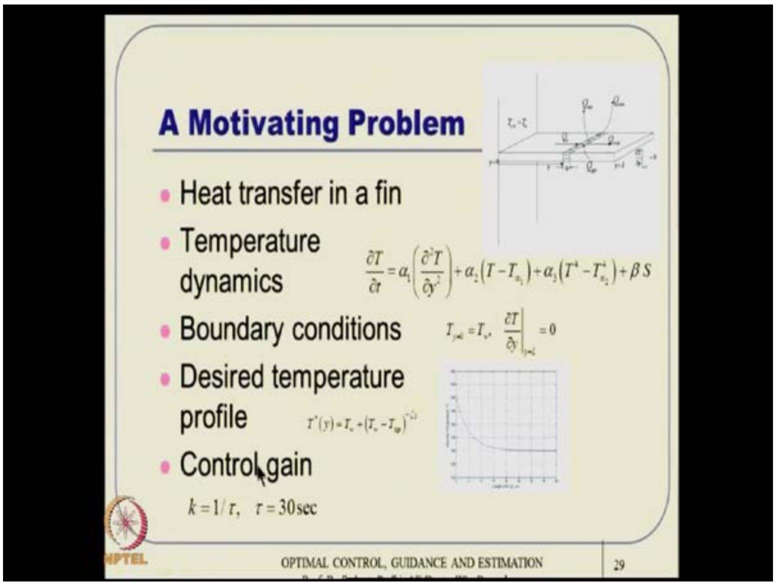
(Refer Slide Time: 40:55)



All right, so, this is the concept here, then finally the control expression, something like when x is equal to x star or a very close domain. x is very close to x star, then you can switch over to this simplified expression. Otherwise, you can implement this way. And remember, these two are equivalent expression. So, when this equivalent expression you do not want to, kind of, unnecessarily keep on computing.

So, that is why this is the simplified form here. Otherwise, there is no singularity force here. You can still keep on implementing this, nothing will happen? which is not the case when you have discrete actuators which I am going to show you that very shortly.

(Refer Slide Time: 41:28)



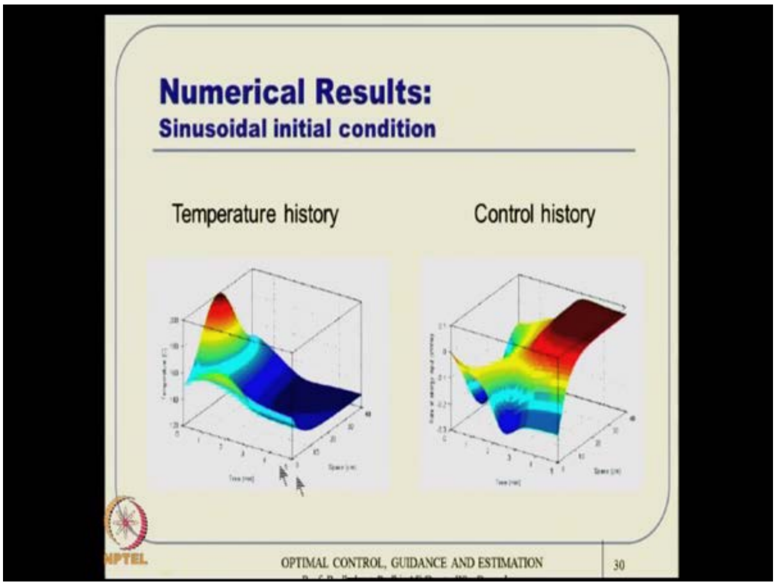
Now, a small motivating problem, this is a heat transfer in a fin. This is a fin here, this is a heat transfer. And here we are assuming all sort of heat transfers. That means, conduction, convection and radiation. And there is a source which is coming through the wall and all that.

So, this kind of modeling will excite non-linear behavior also through that. And depending on the situation, how much (( )) will reflect on the, depend on the alpha 3 quantity. And ultimately, what you really need is a steady state profile. And steady profile we need something like this. Obviously, there is a root. This is something like fin for heat dissipation.

Many times this, I mean, this heat dissipation fins are used in various mechanical devices and all that. So, inside this there is a tube sort a thing. And these fins are protruding out just to be dissipate heat, because there is a source inside that. So obviously, the wall temperature is expected to be slightly greater than that. It is kind of unnatural to assume that it will be throughout the same spatial domain it will be same. We can do that with heavy emphasis on control and all that. But practically it may not be able to do and not be that easy to do also. So, we assume something like this, which is very much reasonable.

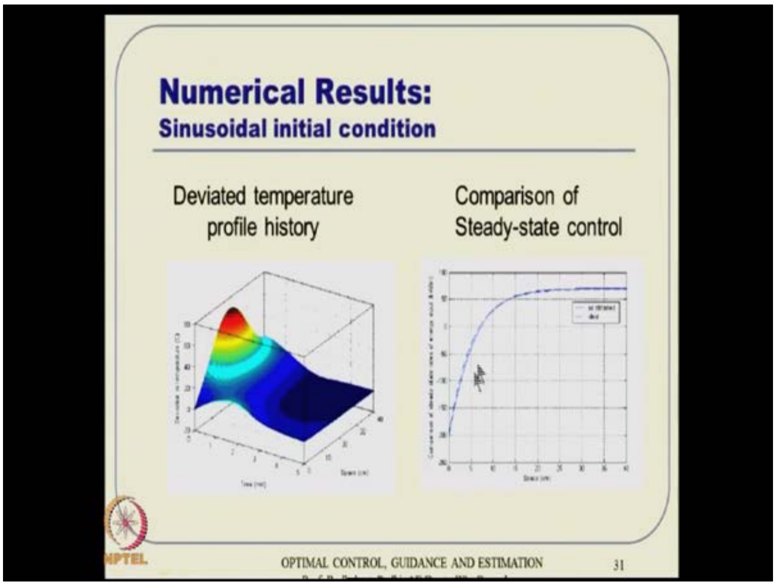
The wall temperature is something like 350 degree centigrade, sorry, 150 degree centigrade. And slowly it will come to something like 130 degree time. I mean, yes with spatial domain sort of thing. 20 centimeter away it is 130.

(Refer Slide Time: 43:00)



So, if you heard that, then the adoptive tuning parameters are this. This is the kind of results. If you start with initial profile, arbitrary, whatever it is and ultimately it will come to a profile that you want, this is the profile. This is the space y. So, this is what you want it in this form. So, it is happening. And associated control history has to be given like that. So, this is with respect to a sinusoidal profile as initial condition basically.

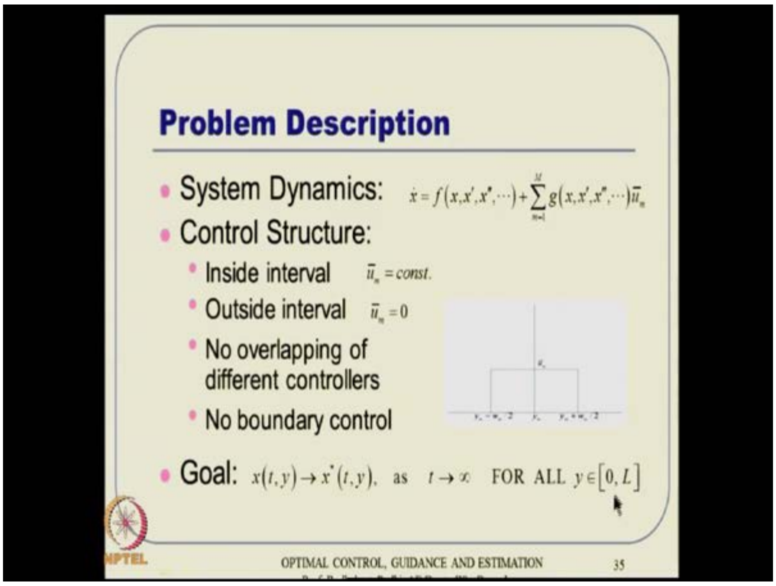
(Refer Slide Time: 43:25)



Now, something like, whether that is happening or not happening, you can put that derivation history also, there are of the temperature profile and all that. And you can see, under steady state, how closely there are merging actually here, a little. They come and exactly kind of do what you want to do. And some arbitrary profile, if it has random initial conditions, then also it will come out; that this is the control part of it. But it will also come out that, the error goes to be 0 everywhere. Again it will be what you really want.

Now, going to the next part of it, which is like distributed parameter system control using O D Is with a set of discrete actuators. This is more practically relevant, because continuous actuators, even though you get nice results like that, how do you actually implement this controller throughout the time. This is evolution of time, and this is evolution in space. So, every time you have a nice mathematical solution, but may not to be possible to implement. So here, we want to address that concern and then tell how are discrete set of actuators at set of discrete actuators.

(Refer Slide Time: 44:26)

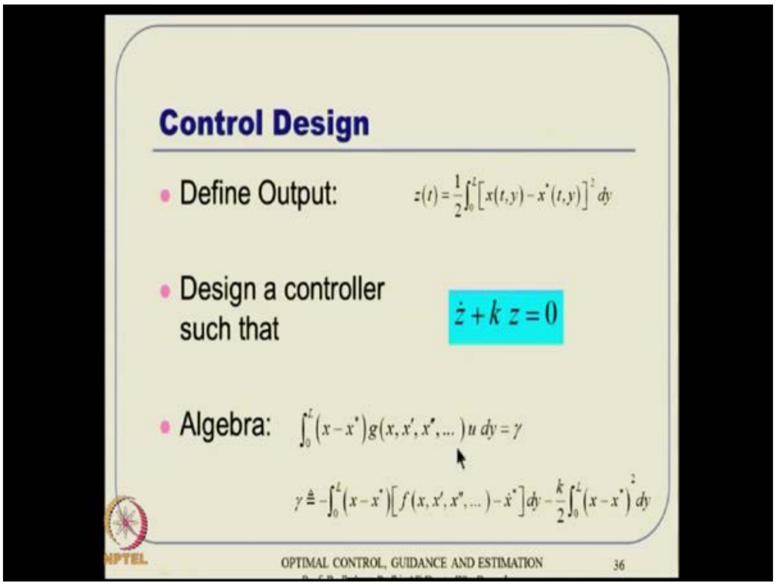


So, we have fixed locations. We have this controller 1 v r, 2 v r, 3 v r like that. Then what you do? Now, at m, y equal to y m, analyze the situation. We have this finite width controller sort of thing which it will everywhere else may not be there, but it is a set of discrete actuators at y equal y m, we have something like a constant control, u m bar, which is y m minus w m by 2, y m plus w m by 2. That means, the width of the controller is something is like w.

All right, so if you have that kind of thing, so then what to do? u m bar is constant and outside the interval it is 0 within this interval, it is some value, outside that is 0. And also, we assume that there is no overlapping influence. Obviously, that is also a practical constraint. How can you have a overlapping actuators? That means, this next controller will be situated far away from this one. So, the actuation says these are all independent. And also, you assume that there is no boundary control action, which is actually rather a little bit strong assumption.

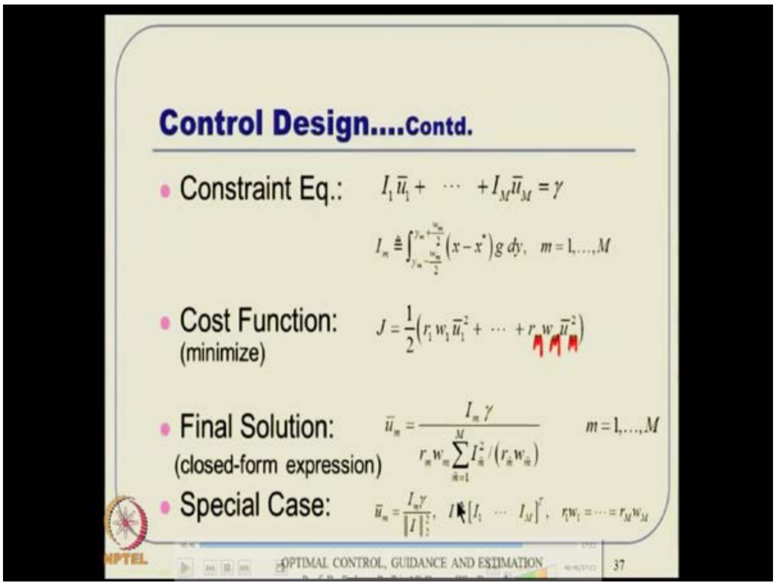
Typically, a boundary control is a subject of topic. It is even more relevant in partial differential equation. Suppose you want to control a vibrating string, probably you want to hold it at one hand and then vibrate it at one hand; one end of the string, so that it will ultimately take it to that side. Those are boundary control problems. But here we have to assume that there is no boundary control. Everything happens inside the spatial domain in discrete sense. Object x will remain same, x will go to x star for all y.

(Refer Slide Time: 46:06)



And then again, we similarly we define an output vector. And you also design a controller, such that this aerodynamics is satisfied. So, until this point, everything remains same. Again, we will land up with some constraint equation like that, and here also remember, you are talking about single equation, but you have a many number of controllers, but which are finite. So, u is not like spatially distributed control infinite dimension, but we have a set of discrete actuators, c 1 bar, u 2 bar things like that.

(Refer Slide Time: 46:36)



So, if you expand this expression, and then write this equation in terms of u 1 bar, u 2 bar up to u m bar. So, this is the expression that you can write. So, at grid points 1 to 2, 3, 4 up to M, capital M, we have this control actuations, which a small width, w. And in that, we have assuming there is a constant control actuation.

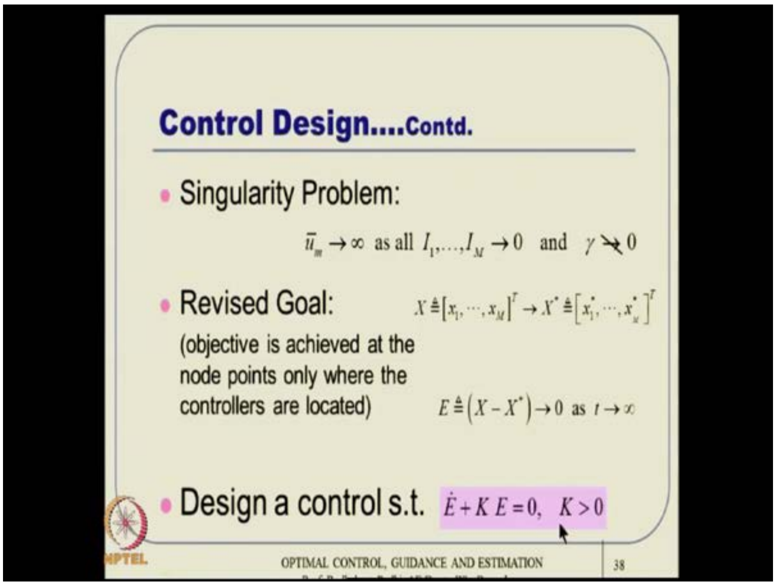
So, the I Ms can be computed that way. But u is or my control input variables that I want to compute. Cost function, obviously, this is also a under constrained problem because there are several freedoms and constraints being lesser. We want to have a cost function optimization. So, this is a cost function again, we assume like that. Again, the minimum values of control throughout the domain. But we can have relatively why it up or down, depending the values of r 1 to r n alright, well there is a small print mistake again. This is to be capital M.

And the final solution turns out to be like this. And if you assume that this r 1 w 1 is equal to r 2 w 2 and things like that. They, in combined sense, they can be equal. For example, if you assume all width of the actuators are same and all weightings are same. That means, r 1 is equal to r 2 equal to 3 like that and w 1 equal w 2 equal w 3 like that. Then, all these values are also any way satisfied. But in general, the mathematical condition required is something like this, r 1 w 1 is equal to r 2 w 2 like that.

If this condition is satisfied and this moment of inertia is this vector I 1, I 2 up to I M, then, if this control turns out to be of this form in a special case, if this condition is satisfied; if this not satisfied, then this the one.

So, this is what it is. So, now we want to have a solution, so which will give minimize this cost function subject to this constraint equation. And the solution is also available in close form.

(Refer Slide Time: 48:48)



Now, there is a problem here. The problem turns out to be like that when as all I's goes to 0 and gamma is non-zero, then u m will go to infinity. If you see this, when will that happen? Now remember, the expression for y and all that i's and all that when X goes to X star, then I will go to 0. And X going to X star is our objective, we cannot escape from that, that means that has to happen is asymptotically.

So, when this starts happening? Our objective starts happening. Then we are starting to have some problem. But it has to have, because you remember, this denominator, fortunately, there is a norm square sort of thing. That means, this norm will go to 0 only when all these I's go to 0 together. Not that one I will go to 0, every something happen. No. All I's have to go to 0. That means, everywhere in that this domain that where the control actuation is happening, this width w 1, w 2, w 3 up to w m, everywhere those segments have to go to 0. Then only this problem will happen.

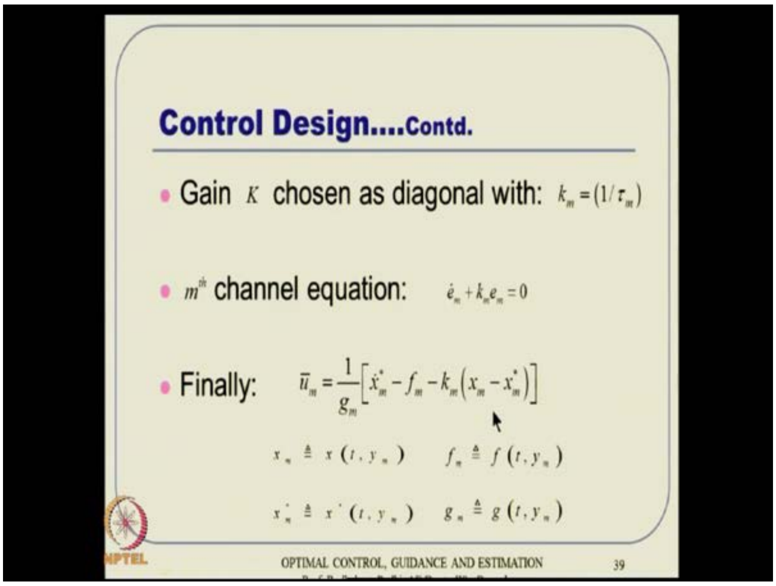
Now, this happens when the objective is made. So, that point of time it will, if the objective is met, then why formulating and then getting ambitious, and telling that through this control actuation, I will be able to control in other domain also. It may not happen because wherever control is located, there I should see some errors at least. If nowhere else I see an error, then it becomes some sort of an observability problem, and error is not there. So, control remains silent, because the things I mean the objective is met.

And if you really want to control it through the errors in the other domain, because the errors are 0 here; that means, the effect control effectiveness is 0 at that point of time, and that point in space. So, every control starts blowing off. It goes to infinity; that is the physical reason for that.

So, then you tell, if that is the case and that point of time will have a revised objective. We will not aim that everywhere it should go to 0, but only those points where the control where the actuator is located at that point we will continue to hold the state values at the desired look, desired values.

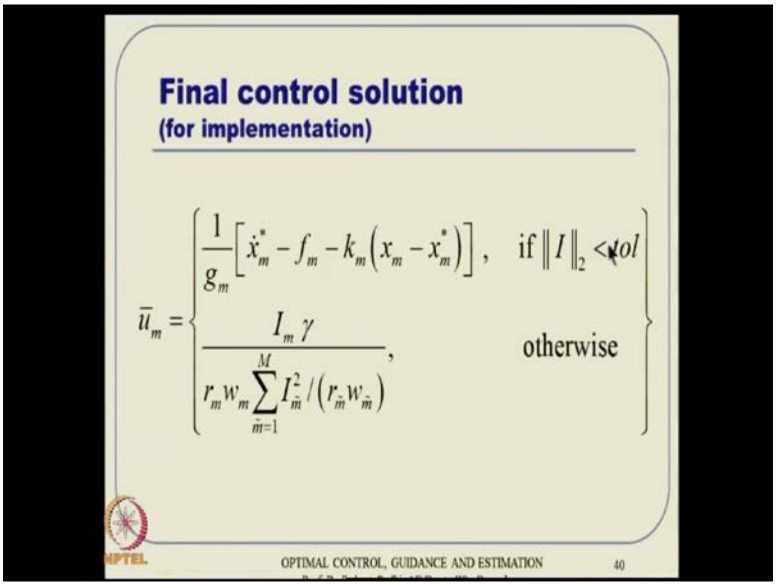
If you have that, then if you define a state vector like this, x 1 to x m and x dot is something like that, and tell my X has to go to X dot in discrete sense. It may not happen everywhere. I do not want to see that. But in this domain because I do not see any error in the state, I will continue to hold that; that is the objective there. So, if you have that, then again you can have this aerodynamics getting enforced here, like dynamics inversion sort of ideas.

(Refer Slide Time: 51:25)



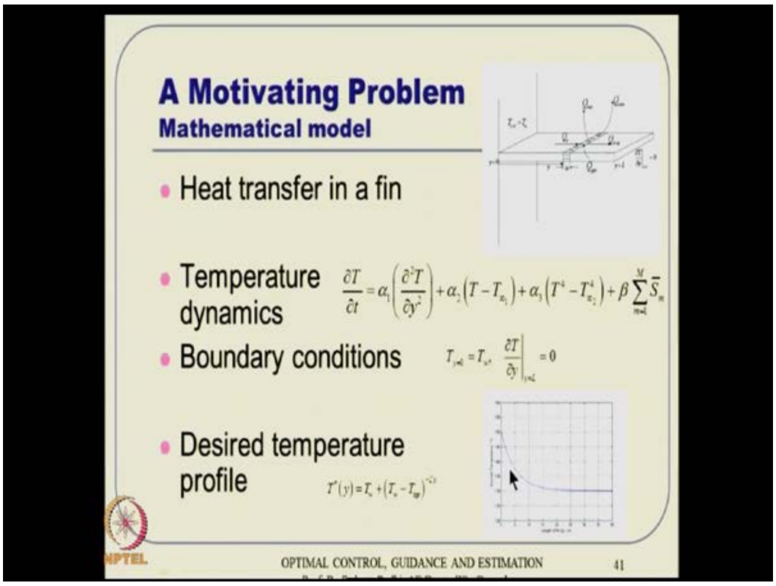
You can in m th channel, this equation will sit and hence you will get a control like this. And this, the other control does not hold on to this by the way. It is a different objective. So, we have to take a call sometime, when to sit.

(Refer Slide Time: 51:40)



So, in implementation sense, this is something like this. If I 2, this norm of I is better than some tolerance value, then use this, but if it is less than some tolerance value shift to this. And this shifting does not happen in a smooth way. So, it would take a call of this number of tolerance that you want to put there.

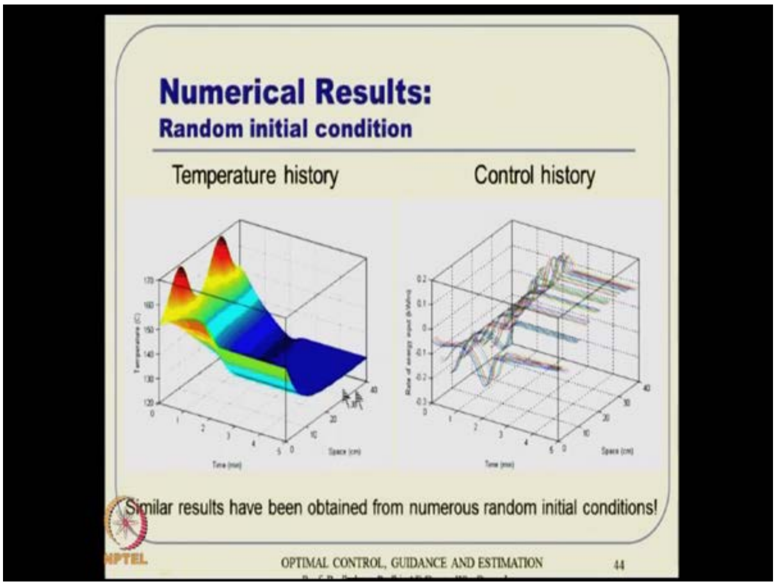
(Refer Slide Time: 52:01)



So again going back to the same heat transfer problem, because it contains non-linear terms, and this is also a miss boundary condition. Remember, at y equal to 0 there is a wall temperature, but at the free end, the derivative is 0. That means, this free end is insulated, that I forgot to tell, but this is what it is? One side it is regional boundary condition, other side it is normal boundary condition.

So, desired temperature again remains same. So, system dynamics is same, objective is same. Only that the control (( )) is not continuous, but it is a finite set of actuators.

(Refer Slide Time: 52:35)



So, now, using this these formulas what we just derived and we assume 5 control constraints are there. And remember, everywhere it is constant. Whatever this grid point we are talking about, this has to be constant. And in between there is no control. Control happens only here a little bit, here a little bit, here a little bit, like that. And here you see some sort of a discontinuity. That is where there is a control ceasing. We want to switch from one control formula to the other formula. And also, see there it is happening, but not in a very good way, there is some sort of waviness. The error is not going to 0 everywhere. Some sort of, it is roughly happening, but not happening in a very good way which is also expected because you really want to control an infinite dimensional system through a finite dimensional control. Obviously, your capability is lesser.

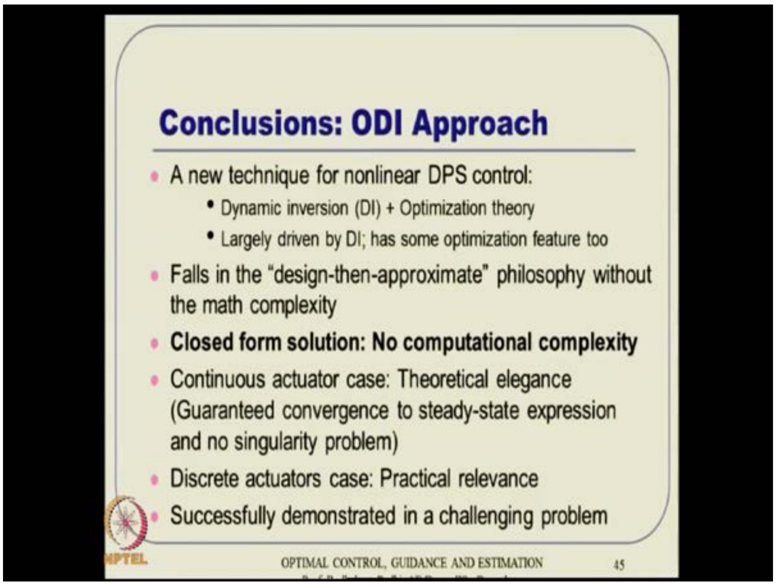
So, what about increase and doing an experiment, again these are numerical experiments anyway, so doing an experiment with 10 actuators rather, instead of 5. So, somewhere this wavy nature has to go, and it does happen. If you take 10 actuators and the wavy nature is much more smoother. You can put much lesser tolerance value because remember the singularity happens only when this goes 0 everywhere. All of this control, I mean, this integral value has to 0. Then only this control will go to infinity otherwise no.

This is much better. The more and more number of control actions you have, we have that many freedom and hence we can have much better job engineering sense basically.

All right, so, that is natural. Expected sort of behavior. Again this is a random profile, and we stick to 10 actuators again, because of this behavior. And then if it is a random behavior profile, again it is able to cut off all that this is the control behavior and finally, you see the temperature, desired temperature is met.

And this kind of repeated, see this is a, once you write a code with a randomized initial condition which can be done using Fourier series. You can run your program repeatedly, every time it can pop up some randomized initial condition, and then let you simulation happen through your control design decision. Every time we will end up with something here, that exercise you have done many times, and the many initial condition you have verified, so similar results have been obtained from numerous random initial condition.

(Refer Slide Time: 55:01)



So, the conclusion sense of this particular approach is actually a new technique which did not exist in the literature we proposed for the first time. And essentially it deals with this, it proposes a new idea of distributed parameter system control using dynamic inversion which is a non-linear control theory, the control synthesis theory using feedback linearization plus optimization theory. These two are coming together to propose a new idea there, falls in the design then approximate philosophy without too much of math complexity. Once you understand dynamic inversion ideas, which again I emphasize, if you do not know, then I suggest that you see you one of I think, couple of my lectures in my other course on dynamic version. It will even make more sense.

So, it actually falls in the Design-Then-Approximate, because I mean what we are telling is, in the double of mind we are not starting with this finite number of grid function. Start with a whole special domain as it is due to some special integral. Ultimately, you have a formula and then while we implement you discretize. So, that falls under the Design-Then-Approximate, but does not it completely avoids this infinite dimensional operation theory. And hence it is not mathematically complex.

Also leads to closed form solution and hence there is no computational complexity issue. And then, in continuous actuator case, there is theoretical elegance; that means guaranteed convergence is there. And there is no control singularity problem. However, discrete actuators case can also be solved which is practical relevance. And successfully it has been demonstrated for non-linear program. The heat transfer problem contains radiation terms also, which is both power of temperature, so with that I think I will stop this lecture. Thank you.

.