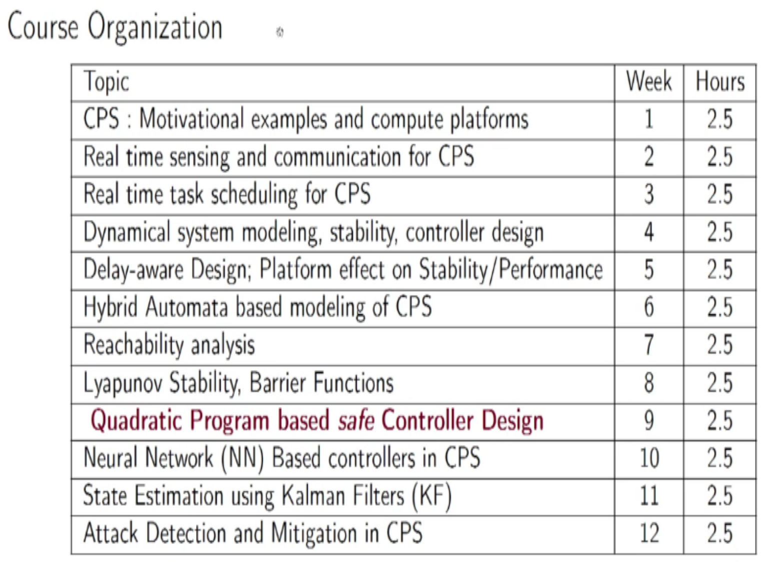
Foundations of Cyber Physical Systems

Prof. Soumyajit Dey
Department of Computer Science and Engineering
Indian Institute of Technology – Kharagpur

Lecture - 43
Quadratic Program Based Safe Controller Design

(Refer Slide Time: 00:30)



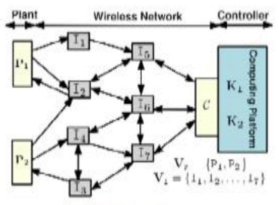
Course Organization

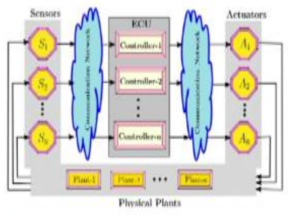
Topic	Week	Hours
CPS : Motivational examples and compute platforms	1	2.5
Real time sensing and communication for CPS	2	2.5
Real time task scheduling for CPS	3	2.5
Dynamical system modeling, stability, controller design	4	2.5
Delay-aware Design; Platform effect on Stability/Performance	5	2.5
Hybrid Automata based modeling of CPS	6	2.5
Reachability analysis	7	2.5
Lyapunov Stability, Barrier Functions	8	2.5
Quadratic Program based *safe* Controller Design	9	2.5
Neural Network (NN) Based controllers in CPS	10	2.5
State Estimation using Kalman Filters (KF)	11	2.5
Attack Detection and Mitigation in CPS	12	2.5

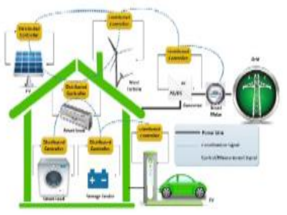
Hello and welcome back to this lecture series on Foundations of Cyber Physical Systems. So today we will be starting with our coverage here on quadratic program based safe control design.

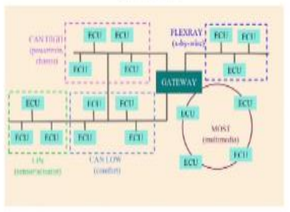
(Refer Slide Time: 00:42)

CPS Examples



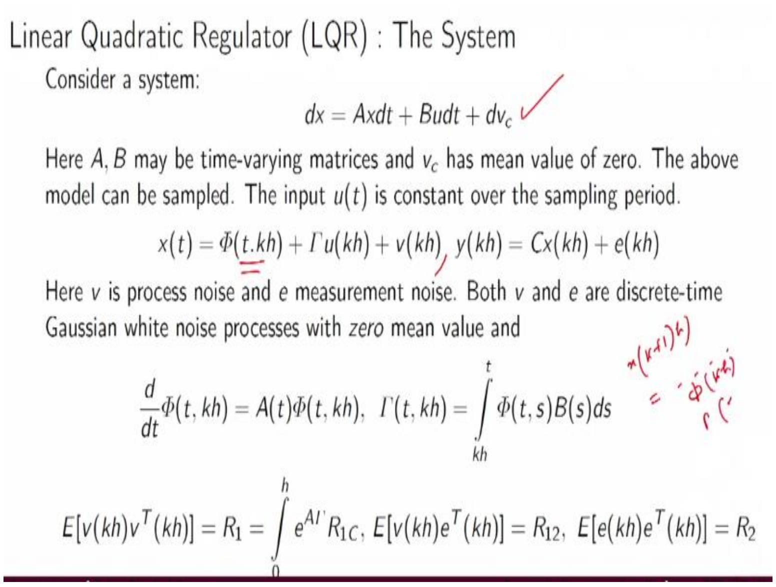






So before going into any of these let us first understand that what we mean by all these. So we have we have talked about some methods of controller design and that was the pole placement based method and let us also appreciate that there can be several other methods of control design and one of them is designing optimal controllers. So what that would really mean is that you have a cost function which captures your control objective and the cost function can be expressed in several ways.

One of the ways is a quadratic program and we will see what such programs mean. And so essentially the controller design boils down to solving that optimality condition or that or that objective function or basically maximizing or minimizing that objective function at every instance of the of the control iteration. So we set up a cost function which captures the control objective and we figure out what should be the control input so that this cost function is optimized in that control iteration. So that is that is the basic idea of designing such controllers **(Refer Slide Time: 01:53)**



Linear Quadratic Regulator (LQR) : The System

Consider a system:

$$dx = Axdt + Budt + dv_c$$

Here A, B may be time-varying matrices and v_c has mean value of zero. The above model can be sampled. The input $u(t)$ is constant over the sampling period.

$$x(t) = \Phi(\underline{t.kh}) + \Gamma u(kh) + v(kh), \; y(kh) = Cx(kh) + e(kh)$$

Here v is process noise and e measurement noise. Both v and e are discrete-time Gaussian white noise processes with *zero* mean value and

$$\frac{d}{dt}\Phi(t, kh) = A(t)\Phi(t, kh), \; \Gamma(t, kh) = \int_{kh}^{t} \Phi(t, s)B(s)ds$$

$x((k+1)h) = \Phi(kh)$ $\Gamma($

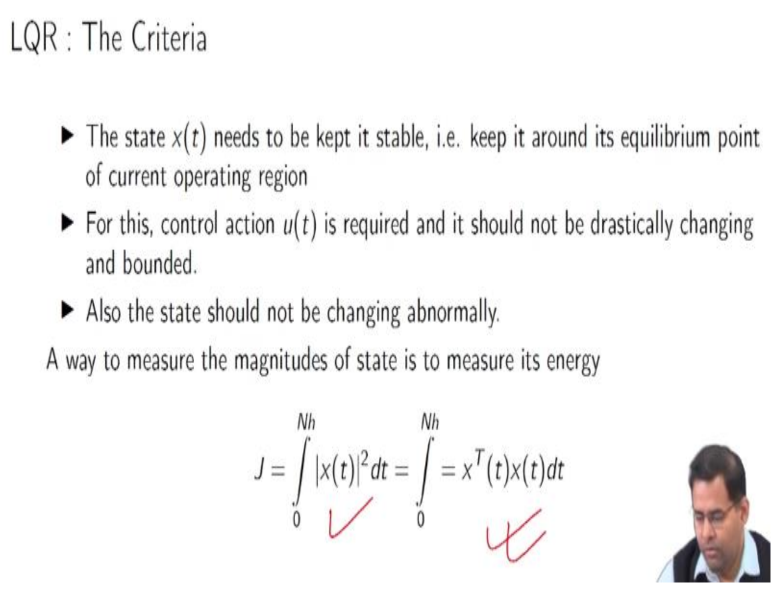
$$E[v(kh)v^T(kh)] = R_1 = \int_0^h e^{At'} R_{1C}, \; E[v(kh)e^T(kh)] = R_{12}, \; E[e(kh)e^T(kh)] = R_2$$

and we will we will talk about one such class of system called Linear Quadratic Regulators. So, we start with a system assumption. So, we have let this be the system assumption here okay, where you have A and B in general they can be like time varying matrices okay like we discussed earlier at the start of our course. And v_c is a process noise with a mean value that is zero right. And if you can also sample the system and if you sample the system you will typically get difference equation like this where the time instance are given like this right, that t is equal to k h and then so a different such time instances.

So you have so let me just figure this thing in a better way so let us say x at k plus one h will relate how to the values of Φ and Γ at kh. So that is what we are trying to talk about here okay so this is y yeah and this is a y equation okay now this v is your process noise and e is your measurement noise here. And for both we will assume that they are gaussian noises with the zero mean okay and we will have these usual relations between phi and a and gamma and phi etcetera right.

And also what we will assume is well how do I parameterize these noise signals we will parameterize them as zero mean gaussian white noise processes for which the mean is zero and their variance is given by R_1 one here for the process noise okay expectation of this and this and the variance of the measurement noise is R_2 and the covariance of variance and of the measurement and process noise together is R_{12}. So these are the terminologies we will be using in this control design objective here okay.

(Refer Slide Time: 04:08)



LQR : The Criteria

- The state $x(t)$ needs to be kept it stable, i.e. keep it around its equilibrium point of current operating region
- For this, control action $u(t)$ is required and it should not be drastically changing and bounded.
- Also the state should not be changing abnormally.

A way to measure the magnitudes of state is to measure its energy

$$J = \int_0^{Nh} |x(t)|^2 dt = \int_0^{Nh} = x^T(t)x(t)dt$$

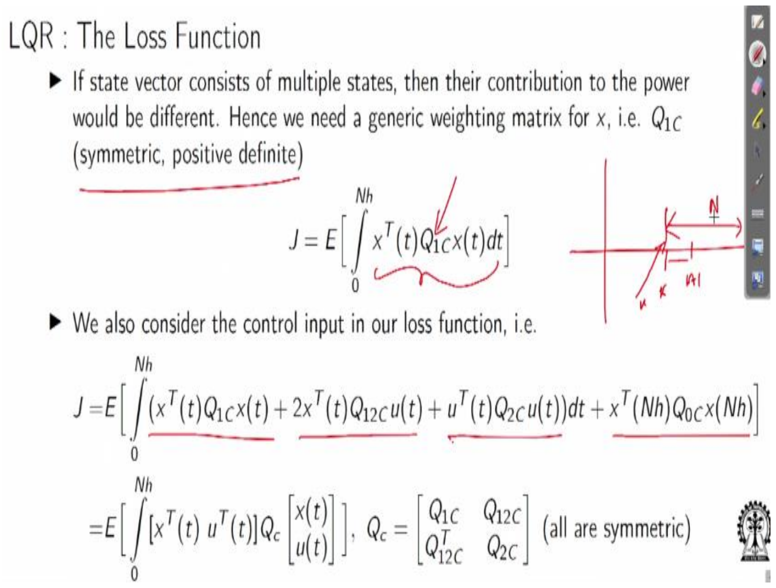
Now let us understand what is linear quadratic regulator like I said the general class of controllers we are going to talking about here is there must be a cost function which will kind of capture the systems control objective and in every iteration the choice of control input is with the objective that you want to optimize this cost function and this thing will go on in every iteration of the system.

the way we look at optimizing the control here is that we have a linear system and for that the state needs to be stable that means it should be hovering around an equilibrium point which is the current operating inside the current operating region and so x should not deviate too much okay assuming in general that this deviation is non null, it should not deviates too much right, and you are operating around an equilibrium of zero okay.
So that means the the value of x can be a bit above or below zero. So it can be positive and negative both. So I would like to minimize this square sum right to, so that is the idea. Also for health of the control system and for many practical considerations I want to write the control input to be changing drastically because no system no physical system likes drastic changes right.

There are inertial constraints there are also constraints like actuator saturation and stuff so all you want is that the system must steer in its trajectory very smoothly and it should not wear too much around the set point and the control input should not be changing in a very ad hoc drastic manner okay so that change should also be smooth right. So the way to encapsulate these ideas in a nice cost function can be as follows that well, you have something like this which is capturing that well what is the what is the value of this state is it deviating too much, if it is deviating too much you know the value will be high or low to too low below I mean in the negative sense.

So well why not take a square sum like this, because in a way as we have seen while we studied about Lyapunov and all that we have seen that well something like this a quadratic form of x is also in a way denoting the energy stored in the system right. And the system in a stable state will like to be steered towards its to a to lower energy states.

(Refer Slide Time: 06:46)



LQR : The Loss Function

- If state vector consists of multiple states, then their contribution to the power would be different. Hence we need a generic weighting matrix for x, i.e. Q_{1C} (symmetric, positive definite)

$$J = E\left[\int_0^{Nh} x^T(t) Q_{1C} x(t) dt\right]$$

- We also consider the control input in our loss function, i.e.

$$J = E\left[\int_0^{Nh} (x^T(t)Q_{1C}x(t) + 2x^T(t)Q_{12C}u(t) + u^T(t)Q_{2C}u(t))dt + x^T(Nh)Q_{0C}x(Nh)\right]$$

$$= E\left[\int_0^{Nh} [x^T(t)\ u^T(t)]Q_c \begin{bmatrix} x(t) \\ u(t) \end{bmatrix}\right],\ Q_c = \begin{bmatrix} Q_{1C} & Q_{12C} \\ Q_{12C}^T & Q_{2C} \end{bmatrix} \text{ (all are symmetric)}$$

So in general if the state vector has multiple states then their contribution to this to this energy indicator can be different right. So ideally we will like to have this in a nice quadratic form like this so its not just x square but rather you have x transpose x with a weight matrix in between Q_{1C} and the what is the coefficients of the weight matrix doing it is assigning let us say more weight to some variable x which is more sensitive less weight to some variable which is less sensitive something like that right. So that is and it will be a symmetric positive definite.

So that is how we actually define cost function which is going to capture this thing that well x must not be varying too much from the target equilibrium location okay. Now like I said that I will also like to have the control input not changing drastically. So we can introduce a similar quadratic term in my control input, for my control input and once I do that then I will have this weighted quadratic term for x and I would also have a weighted quadratic term for the for the for the controller and for the for the control input right and overall we would like to minimize loss function which is kind of looking like this.

So here as you can see you have you have considered x you have considered u also and here you also have a weight matrix which is capturing x and u together here okay. And not only that I would like to also consider that well what is my state at the end of this control window. So we are we are actually when we are trying to take it as a summation we are considering not only at single instances but the cost function has to be kind of windowed because we are trying to see that well suppose I am trying to choose the control input right now but I am not only worried about what will be the effect of this control input at this kth time instant or in this interval of k to k plus one, but what is the effect in the long run up to some finite horizon of size n.

That means I want to capture mathematically that well how do I choose this u so that this cost function is optimized not only in this period but over a finite horizon that means let us say over the next n successive periods. I want to know about that. So that is why n comes as a parameter in this design you can choose n to be one two three the more is the value of n if you think what you are doing is you are unrolling your loop over n future iterations so that your current choice of u is more and more knowledgeable.

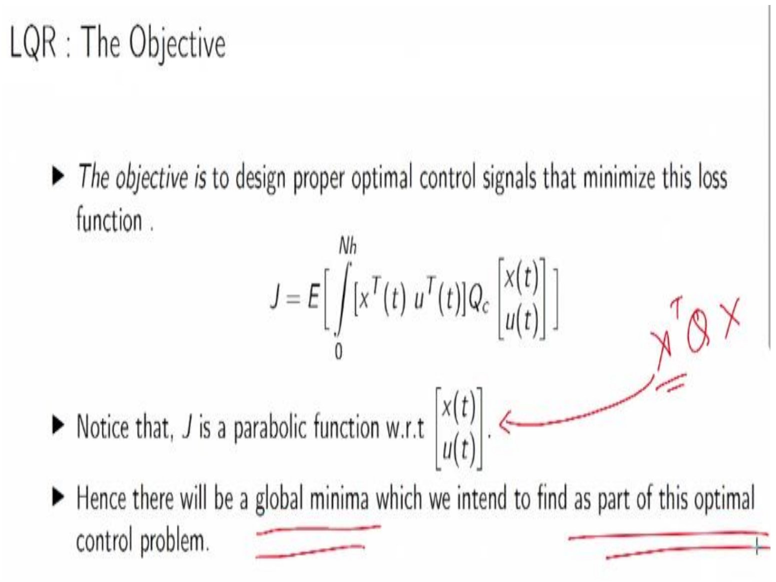
Because if you use a smaller iteration smaller n that means you are not really looking ahead into how the system will unfold based on your input now right the more you choose a higher value of n you are unfolding the system for a larger horizon and then you are really able to calculate that what is the effect of my system not only in the next time step what is the effect of my current control input not only in the next next time step but the next to next the even further and all these time steps okay. So based on that you choose a suitable horizon n for doing this integration and you you assume this quadratic terms which capture the interaction of x u and x and u together here okay.

And last but not the least as I also said that you also want to know that what well at the end of the horizon what is the final value of x. So for that you end this additional term here for this nth step and since at the after the nth step I am not considering the control any four any further so that is why I do not have a control term in the inner step but only only the final state because I am interested in this that currently I give an u. What really does my state become through this evaluation and finally into the nth step. So that is why you see this integration is covering this much and additionally you have this term here for the nth step.

So if you write this thing in kind of a shorthand, it will be like this where you consider x and u together as a vector okay and you can have an exp and you can have this integral over zero 0 to this horizon n and you have the state with x and u here and so basically you have a full quadratic expression on both x and u together. So x transpose u transpose here and x u here like that okay. Not only that the way we are writing it is in a nice manner we are trying to capture all these things together these four of them okay using Q_C which should be this these and all of these individual coefficients are symmetric and Q_C is the weight matrix.

All we are doing is we are writing this expression together in this nice form here. Now the question is well we said that we will integrate and all that but what are we really trying to look at. So we are trying to look at what is the average cost of this summation so we bring in an expectation here. Now what we will try to do is we will try to minimize the expected value of this objective function okay.

(Refer Slide Time: 13:15)



LQR : The Objective

- *The objective is* to design proper optimal control signals that minimize this loss function .

$$J = E\left[\int_0^{Nh} [x^T(t)\ u^T(t)] Q_c \begin{bmatrix} x(t) \\ u(t) \end{bmatrix}\right]$$

$x^T Q x$

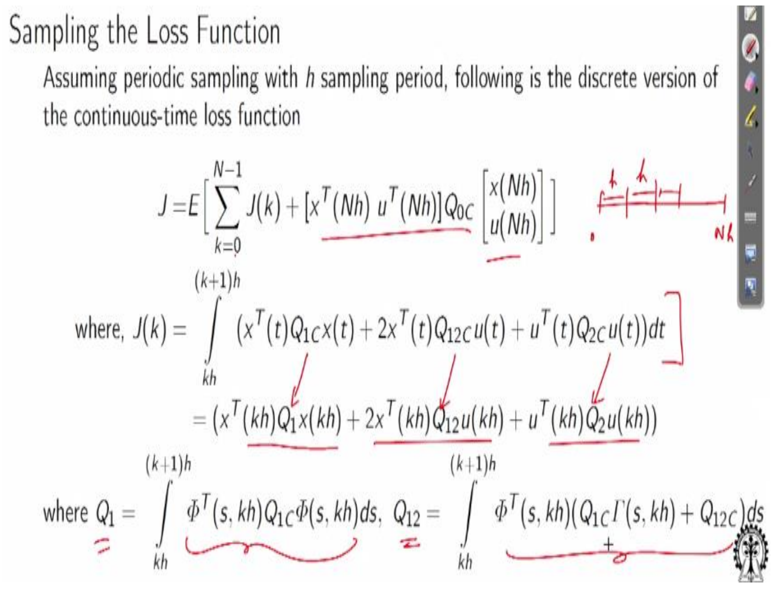
- Notice that, J is a parabolic function w.r.t $\begin{bmatrix} x(t) \\ u(t) \end{bmatrix}$.
- Hence there will be a global minima which we intend to find as part of this optimal control problem.

So that is why we say that the objective here is to design suitable optimal control signals that minimize this loss function and the loss function is nothing but expectation of this integral 0 zero to n h and this analytical form that we have okay. Now since this is a quadratic form j will have a parabolic function if we consider it in the space of x and u because you see that is a function of both x and u this is the state right.

You have a quadratic form which is x transpose Q x where x is x u right. So this is like a if if I plot it in a state space it will be a parabolic function. Now since this will be a parabolic function it will have a global minima I hope this is clear every parabola has a lowest dip right. And that is the global minima which we intend to find as part of this optimal control problem.

We want to figure out that well what is the control input values for which this function will reach the global minima and then that means if I choose those values then for that the we will actually have the system in its least energy state with the smoothest possible trajectory and the smallest possible control inputs okay.

(Refer Slide Time: 14:51)



Sampling the Loss Function

Assuming periodic sampling with h sampling period, following is the discrete version of the continuous-time loss function

$$J = E\left[\sum_{k=0}^{N-1} J(k) + \underline{[x^T(Nh)\ u^T(Nh)]}Q_{0C}\begin{bmatrix} x(Nh) \\ u(Nh) \end{bmatrix}\right]$$

$$\text{where, } J(k) = \int_{kh}^{(k+1)h} \left(x^T(t)Q_{1C}x(t) + 2x^T(t)Q_{12C}u(t) + u^T(t)Q_{2C}u(t)\right)dt$$

$$= \left(\underline{x^T(kh)Q_1x(kh)} + 2\underline{x^T(kh)Q_{12}u(kh)} + \underline{u^T(kh)Q_2u(kh)}\right)$$

$$\text{where } Q_1 = \int_{kh}^{(k+1)h} \underbrace{\Phi^T(s,kh)Q_{1C}\Phi(s,kh)}ds, \quad Q_{12} = \int_{kh}^{(k+1)h} \underbrace{\Phi^T(s,kh)(Q_{1C}\Gamma(s,kh) + Q_{12C})}ds$$

So let us get about this in the discrete domain. So what we will do is in the discrete domain we will change the integral to a summation and we are assuming the periodic sampling with h as the sampling period in a discretized version so this entire thing will now look like this okay so what is j is basically you are starting from k equal to zero to n minus one and you are summing it over this right, this j k where j k is this integral in a single step okay. And plus the last step this one that we talked about here okay.

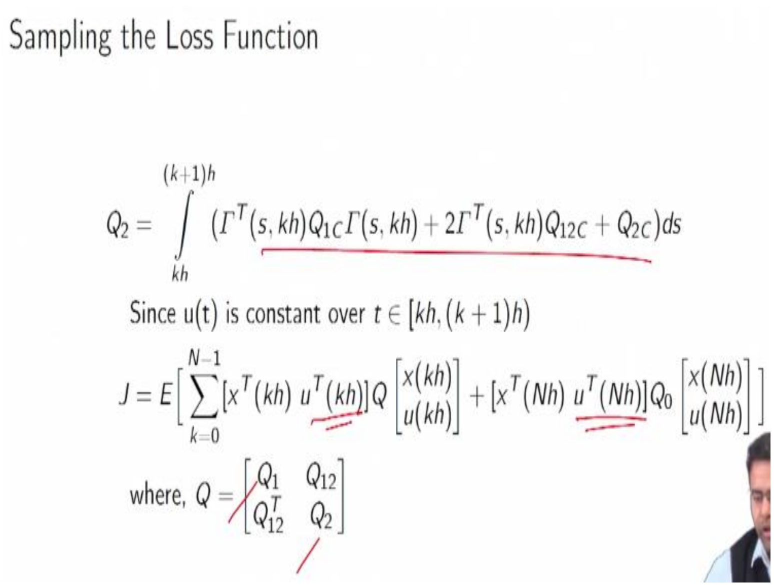
So let us first understand what the step and so we said that where we will evaluate it in a in a discrete domain so in the discrete domain that would really mean that I am going to sum up all these terms. So that means I do not integrate the full thing from zero to n h but rather I consider this small integrals in each of these sampling periods right, because this discretized with an h sampling period so consider this integrals in each of these sampling periods with this following terms like we already discussed right. And then that is so that is your j k that means the value of j in the kth instant. Now we are going to sum it over from zero to n minus one because essentially you are looking for the summation from zero to n h right.

So all you are doing is well you are looking to integrate over a continuous period from zero to n h. Now you say that well you break it into this spaces of h figure out this integrals in individual points. So those are your values of j at k k plus one k plus two up to k plus n minus one right. So that is why you summation you do summation from k equal to zero to n minus one and for each of these intervals you have this interval integrals from k h to k plus one h in this interval the integral value is j k from k plus one h to k plus two h the value is j k plus one so like that. And of course we know that if I do this integral you will get something in in in a in a nice discretized form.

So whatever we get that is what will come here okay and then there is the final step. So then from where just we do this integrations we come from the discrete continuous domain to the discrete domain. So whatever weight matrices we started with in the continuous domain, they will now get converted into the discrete domain variance following these integrals. So I hope this is clear So this was my continuous domain weight matrix now if we apply this integral this is how they will change. And here this phi gamma they are the usual symbols that we use for the systems matrices when we trans when we convert them from the continuous domain to the discrete domain.

So yeah, so this is phi one. Here we as you can see that we are saying j k is this which will be equal to I mean well the integration done it should be this I mean weight matrix over weight matrix is Q_1 with this weighting x and weight matrix Q_{12} in the discrete domain it is again weighting x and u and weight matrix Q_2 is weighting Q right its weighting u and what is Q_1 Q_2 and so this is your the discrete domain version of Q_{1C} here. This is your discrete domain version of these weights here right and you also have this Q_2 here.

(Refer Slide Time: 18:45)



Sampling the Loss Function

$$Q_2 = \int_{kh}^{(k+1)h} (\underline{\Gamma^T(s,kh)Q_{1C}\Gamma(s,kh) + 2\Gamma^T(s,kh)Q_{12C} + Q_{2C}})ds$$

Since u(t) is constant over $t \in [kh,(k+1)h)$

$$J = E\left[\sum_{k=0}^{N-1}[x^T(kh)\ \underline{u^T(kh)}]Q\begin{bmatrix}x(kh)\\u(kh)\end{bmatrix} + [x^T(Nh)\ \underline{u^T(Nh)}]Q_0\begin{bmatrix}x(Nh)\\u(Nh)\end{bmatrix}\right]$$

where, $Q = \begin{bmatrix}Q_1 & Q_{12}\\Q_{12}^T & Q_2\end{bmatrix}$

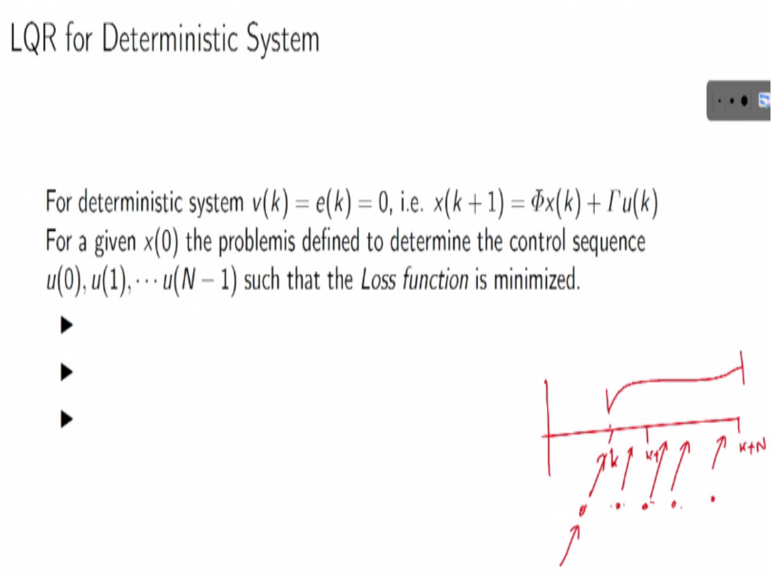
So let us understand of course this conversion can be easily done through some standard tools that we have and once this is done what is the idea is when we are doing when we are doing this optimization in the discrete domain then just like our older control designs what will happen is this u is going to be constant here over this interval right. So I can just free the u here right when I am doing the integration inside this interval right. So my expectation we will just have formulation like this the.

I mean I can just write this value of u as fixed to be u k h. So this should be u transpose k h sorry it is already here and it is in this form right, where we have already figured out that well in the discrete domain what should be the value of this individual weight matrices using this kind of expression that we have here. So just looking back so we started with this we started with this overall integral and then we said that we are doing it in a discrete space. So I will have samplings at every h point right, h sample, after h period. So at each of these intervals the value of j k will be found by integrating inside those sampling intervals.

So it should be this and then once we find the values of j or some form representing the value of j you put them in this summation. So that is the idea and you can figure out from this to this how we come by converting this from Q_{1C} to Q_1 one using this equation that we have just written. So we are not going into how this will come I think you can actually figure that out based on our previous coverages. So in overall let us focus here because these things can be done we can actually figure out them or even better let us think I mean if if you feel overwhelmed by all these continuous time to discrete conversion let us think that well you are given the system in the discrete domain directly okay.

So then you are you are starting your cost function design right from here okay. So you have the loss function design here and you have this state space formulation where with x and u, you have a quadratic form over which you have a summation and you also have the final state n for which you have this yeah. So note that this summation is only over the first term and at the end you add one term for the final values of x and u at the nth step horizon okay so this is your final j.

(Refer Slide Time: 21:43)



LQR for Deterministic System

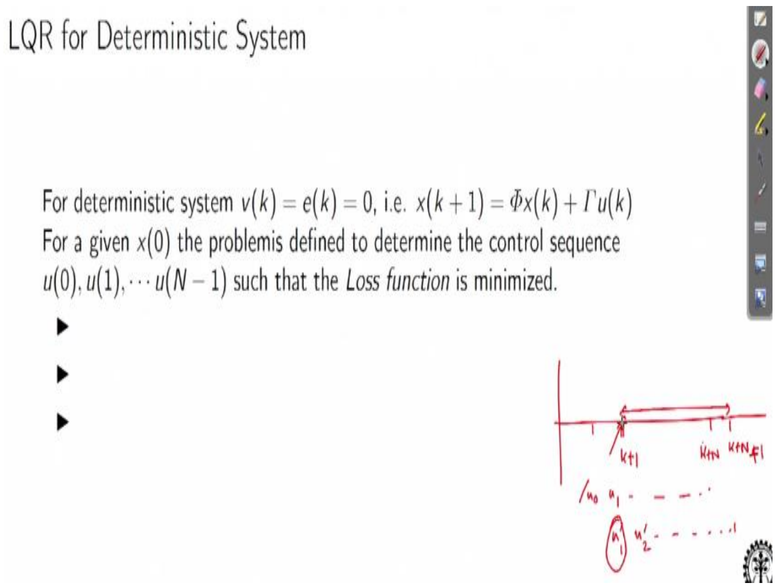
For deterministic system $v(k) = e(k) = 0$, i.e. $x(k+1) = \Phi x(k) + \Gamma u(k)$
For a given $x(0)$ the problemis defined to determine the control sequence $u(0), u(1), \cdots u(N-1)$ such that the *Loss function* is minimized.

Now the objective is that well I want to find out what would be these control values at each of these iterations in the horizon so that this function is minimized okay. Now this is a deterministic system we are considering with this value of noises v(k) and e(k) as zero here okay. So we are we are making it simple we are considering that well the system is simply evolving following this equation. So the noise terms are all zero and my objective here is that I want to determine what is this control sequence.

So what I mean by this is let us say what what we are really doing. So let us say this is k this is k plus one this is k plus n. So I am looking at how the cost function will change over this entire interval up to this n horizon and there will be n positions where this control updates will be given right from u zero to u n minus one here right. So in this cost function you see there are those n unknowns. I mean what should be the value of u zero u one u two etcetera and eventually I will get to some state here the point is how do I choose these values so that this overall cost function is minimized right.

So if I am somehow magically able to solve this then I get a sequence of control inputs. If I apply them starting from here I will minimize them. So that is how this optimization works that means I will solve this online and I will generate u zero to u n minus one among them I will apply u(0) because that is the control input relevant now and then the system will evolve okay.

(Refer Slide Time: 23:27)



LQR for Deterministic System

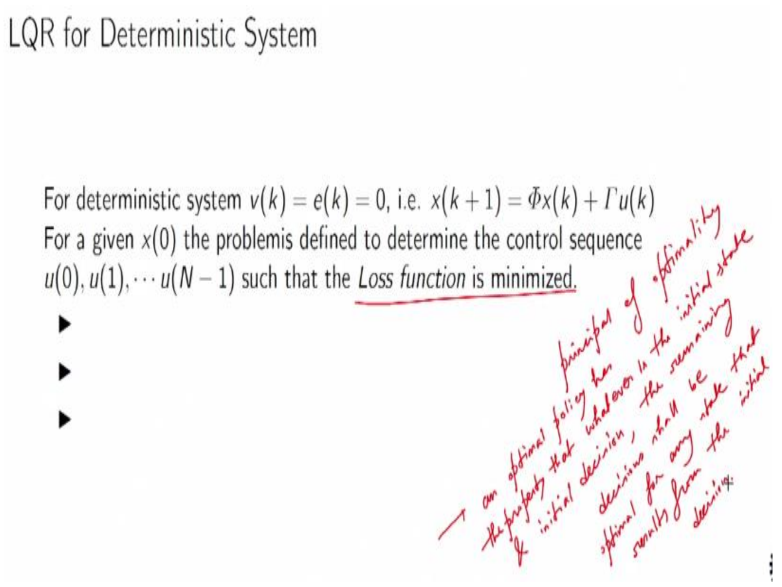
For deterministic system $v(k) = e(k) = 0$, i.e. $x(k+1) = \Phi x(k) + \Gamma u(k)$

For a given $x(0)$ the problemis defined to determine the control sequence $u(0), u(1), \cdots u(N-1)$ such that the *Loss function* is minimized.

So once the system evolves and I come from k to k plus one now again I have measurements of x. So again, so this was my k plus n and then here I have my k plus n plus one. So when I come here do I apply, so I computed u zero u one etcetera right up to u n minus one. Now what do I do at k plus 1, do I apply the old u(1)? No. Again I will form another cost function with the horizon n that means it will now be from k plus 1 to k plus n plus one okay. And now I will again compute a set of new control inputs u(1) prime, u(2) prime etcetera up to this horizon and I will apply the first of them right here. So see this is like a rolling optimization you figure out n inputs but you only use the first of them because that is what you should apply now.

You could have used the second one in the next step third one in the next step. But then why do not do I really do it? The reason is those values unlike already steered in the sense those values have been computed assuming that you have measurement of the current state. Whereas in the k plus 1th step you will receive a new measurement of y right. A new measurement of x we are assuming here so you will receive a new measurement of state for the system. So then instead of again applying u(1) and u(2) and u(3) like that in the in the k plus 1th step with the new measurement again you will create the optimization problem and you will again create a sequence of control inputs and you will apply the first of them. Of course we are assuming that you have enough processor to bandwidth available to do this and we we will work like that so so that is the idea here.

(Refer Slide Time: 25:35)



LQR for Deterministic System

For deterministic system $v(k) = e(k) = 0$, i.e. $x(k+1) = \Phi x(k) + \Gamma u(k)$
For a given $x(0)$ the problemis defined to determine the control sequence $u(0), u(1), \cdots u(N-1)$ such that the *Loss function* is minimized.

-
-
-

principal of optimality
→ an optimal policy has the property that whatever is the initial state & initial decision, the remaining decisions shall be optimal for any state that results from the initial decision.

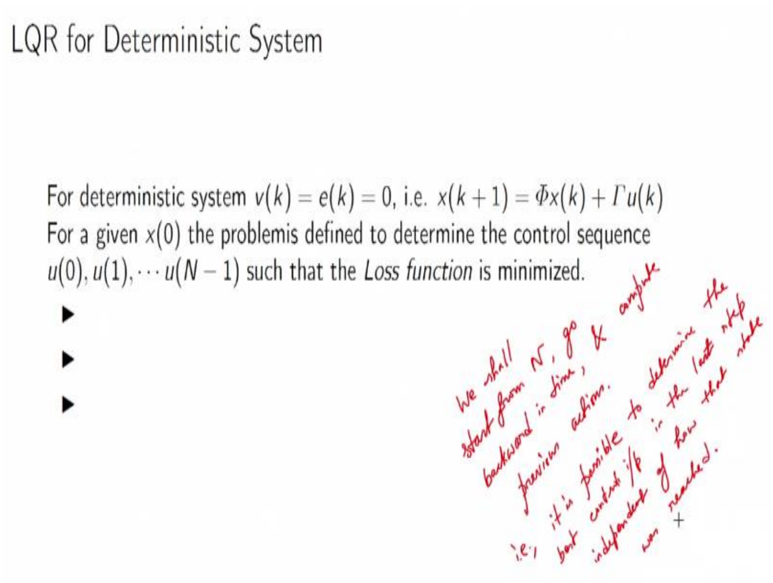
Now the question is that well how do I really go about solving this problem? The idea here is that while minimizing this function we will we will use some principles of optimality and dynamic programming. Now what this really states is that whatever is an optimal prop policy that I choose as an optimal property right now is that what I mean I would say that this policy is optimal in case whatever is my initial state and my decision and that was that has been taken it must be optimal with respect to the state resulting from the first decision. So let us let us understand what it really means. So let me just write this down so the idea is this that I say that this is the principle of optimality I have spoken and written it down.

So it means that well you you have a sequence of decisions to take right. Now I would say that well the first decision that you take in this chain of decisions should I will say that it is optimal if due to the decision wherever your system lands up to, the remaining decisions will also be optimal from this state where you have landed with the first decision. That means the first decision that I took that means the first control action that I took was not optimal only for that time point.

It was optimal for all future time points also that means it is not that due I that due to that control action the loss function got minimized only in that interval and had I not taken that action and I have taken something else then it would have facilitated future actions and because a different choice of this current action could have made future could have could have made a choice of future actions possible so that the overall loss would have been minimized. That is not the case. This is what is optimality. It means that whatever I choose now is something that based on that wherever I land up from that position whatever are my future actions those are also optimal okay.

Let us again understand that whatever I choose now as an action is not only optimal for this interval it is also optimal for future intervals. That means for this choice that I made now there exist choices in future and once I make those choices those choices along with this choice is the best possible way to minimize this function okay. I hope this is clear now. Now why does this principality of opt principle of optimality comes in play here. Because we will try to use this to figure out that well how to solve this problem and how to solve this idea of computing u here.

(Refer Slide Time: 30:57)



LQR for Deterministic System

For deterministic system $v(k) = e(k) = 0$, i.e. $x(k+1) = \Phi x(k) + \Gamma u(k)$
For a given $x(0)$ the problemis defined to determine the control sequence $u(0), u(1), \cdots u(N-1)$ such that the *Loss function* is minimized.

-
-
-

We shall start from N, go backward in time, & compute previous actions.
i.e., it is possible to determine the best control i/p in the last step independent of how that state was reached.

So what we will do is we will assume that initial decisions may have been optimal right and this that that helps us to do one thing. That helps us to start computing the control input from for future that means from the nth step and then going back in time towards the first decision okay. So because we will assume that this in this case we are going to compute the u sequence in such a way that they are all optimal okay that is why it will be positive. That is why we will take the strategy that we will compute the action at the last one and then we will roll back in time to compute the previous actions.

So there is the point we wanted to make it is possible to determine the best input in the last step independent of how the previous choices were made, because we are always assuming that those choices are all optimal okay, and those along with whatever optimal choice we make here are together together optimally reducing the loss function. So that is what we will do. So fine we will conclude our discussion today here and we will resume from this point in the next lecture.