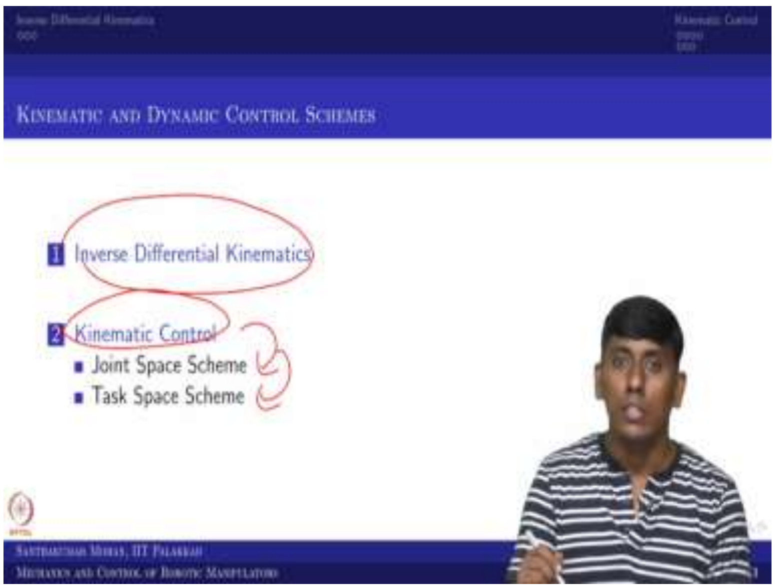
Mechanics and Control of Robotic Manipulators
Professor Santhakumar Mohan
Department of Mechanical Engineering
Indian Institute of Technology, Palakkad
Lecture – 38
Kinematic Control

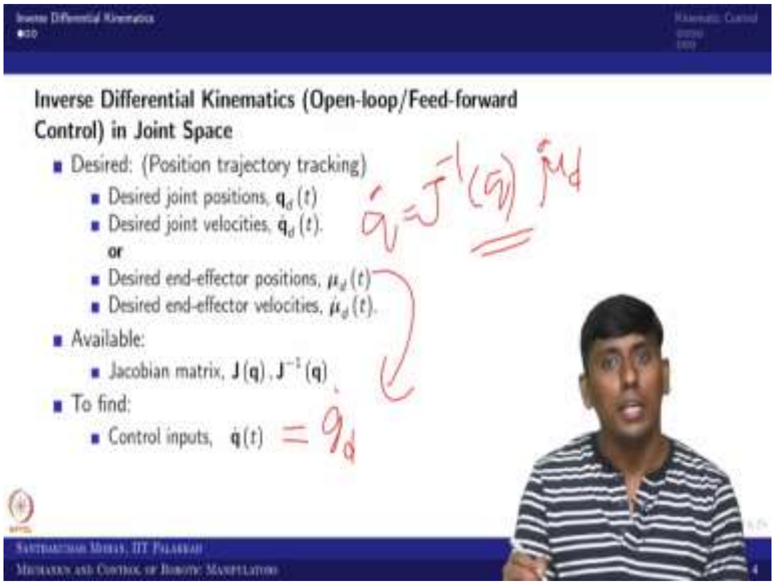
Hi, welcome back to mechanics and control of robotic manipulator. Last class we gave introduction to motion control and types. So, this particular class we would be starting with kinematic control as one of the types which we have seen in the last lecture.

(Refer Slide Time: 00:30)



So, here we start with inverse differential kinematics; then we will talk about kinematic control. Even this kinematic control can be done in either joint space or task space. So, here we are not talking about sensor space, because we are not going to work on real time system in this particular lecture or the course. So, that is why we have restricted only joint space and task space. So, let us move to the inverse differential kinematics straight away.

(Refer Slide Time: 00:54)



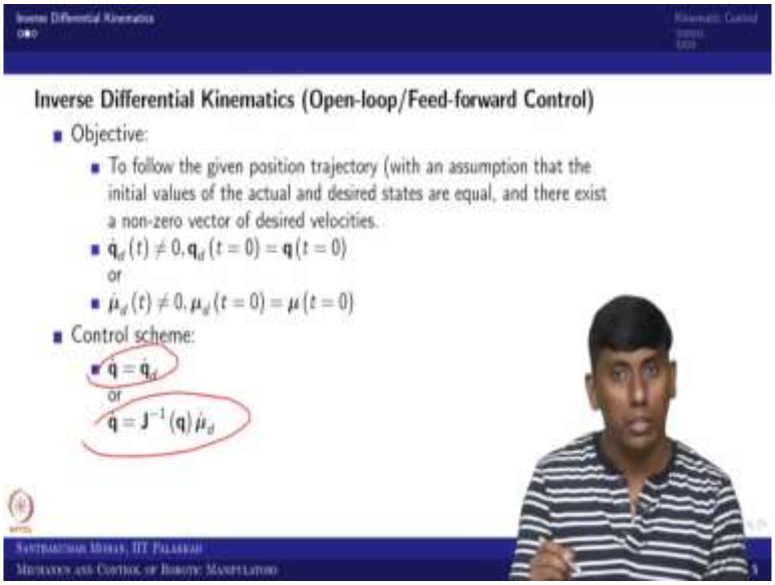
So, what inverse differential kinematics which we can consider as a feed-forward control; or we can say it is an open loop control. So, where we are end up? We end up with a joint space. So, in that case so what we are trying to see. So, the position trajectory is given; so, we have to follow that position trajectory tracking. So, what additional constraints would be given?

So, the q desired and q desired dot would be given to us. So, further so either this would be given, if it is a joint space; if it is a task space, the mu desired and mu desired dot would be given. So, the end effector position and velocity would be given. So, what we expected? We expect that the q of t tends to q desired; but that is not directly guaranteed. But what we can ensure if the q of t is equal; sorry, q dot of t is equal to q dot of desired.

So, that is what we are trying to see. So, available is Jacobian matrix, because this Jacobian matrix is required when we try to do end effector position trajectory tracking. So, then you can see what we need to find out? We need to find out the q dot. Since, it is inverse differential kinematics; we said that the q dot is nothing but q desired dot.

So, if it is like this; so, the J inverse of q into mu desire dot; so that would be q dot. So, this is what we are calling as a inverse differential kinematics. So, for that there are some conditions.

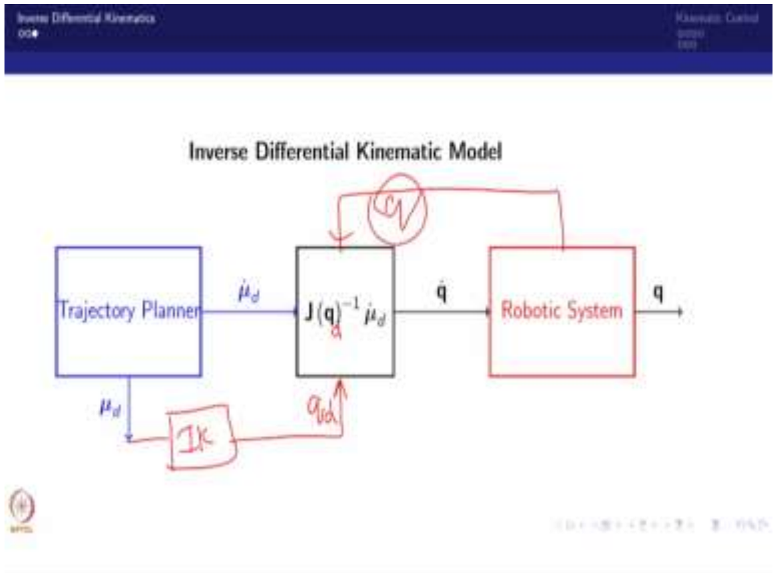
(Refer Slide Time: 02:32)

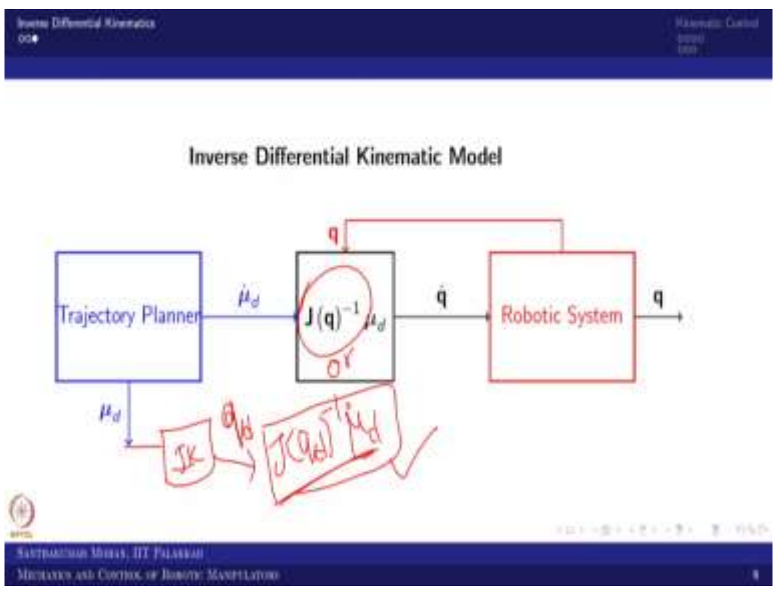


So, that conditions I am trying to put it here. So, what conditions? So, the objective here to follow the given position trajectory; however, what assumption explicitly given, so, with an assumption that the initial values of the actual and the desired states are equal. So, q of 0 and q desired of 0 are same; that is what we are written it.

So, further it should have non-zero desired velocity; then only the differential kinematics will work. So, if that is the case, so if it is in a task space, you can say the mu desired dot supposed to be non-zero. And mu desired at t equal to 0 and mu equal mu at t equal to 0; both supposed to be equal. So, if that is the case what we do the control of, we can directly write q dot as q desired dot or. So, q dot can be written as J inverse of mu desired dot. So, this is what as what you call the inverse differential kinematics; this we have seen already in the regular lecture.

(Refer Slide Time: 03:28)





So, here we will see in a block diagram based. So, the robotic system will give as per kinematic level. So, it would be given as q output; and what would be the input, so q dot; so this is what we assume. And the inverse differential kinematics means what happened? So, there would be a trajectory planner. This trajectory planner will give what?

So, we assume that it is like end effector; so, end effector position in trajectory. So, then this would be giving me mu desired dot and mu desired; so, these two would be available from the trajectory planner. But you need q dot. So, for that what we can use? So, you know that mu dot is J of q into q dot.

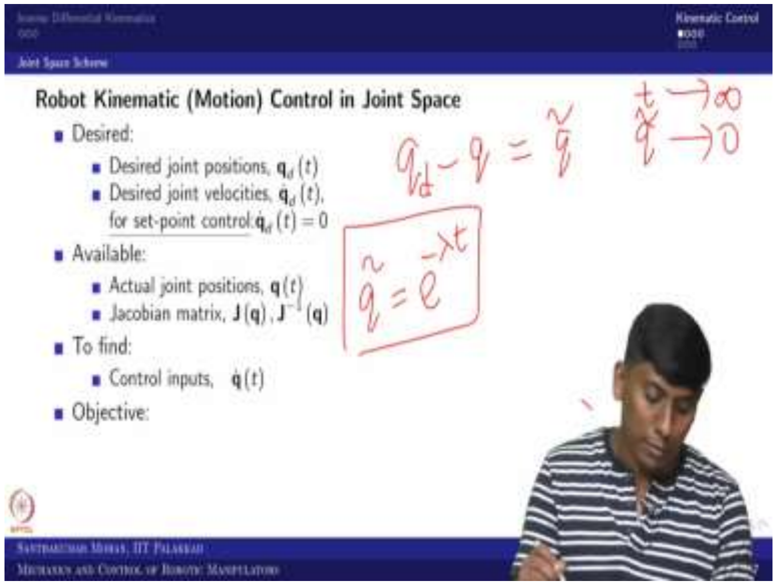
So, q dot can be written as J inverse of, so q into; sorry, so this one is mu dot. But here we are thinking about desired; so, this desired means this is also desired. So, then what we can do it? So, we can bring one Jacobian inverse; so, then you multiply with this. That would be equal to q dot; so that is what we have bringing it. So, now there are two way we can do it; so, we can do the inverse kinematics.

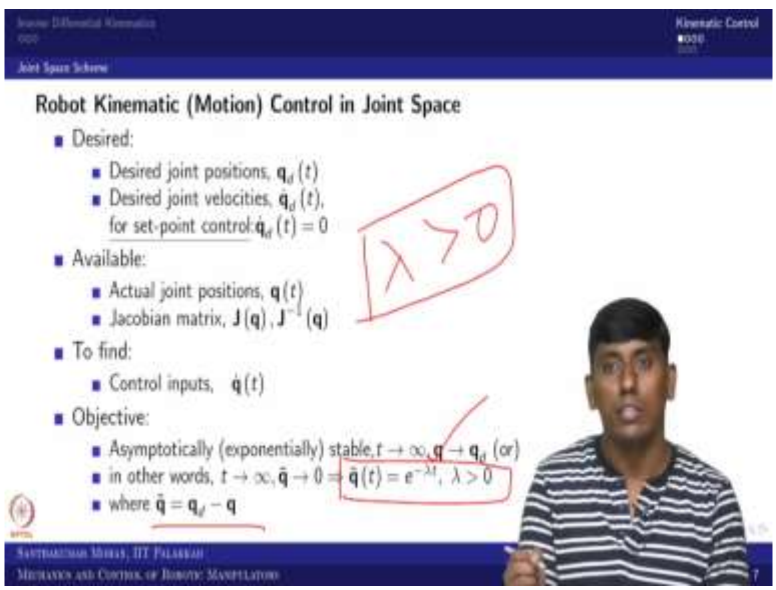
So, then we can do it here, so this way we can do it; so, the q desired can be fed. So, then this would be q desired, or we can take the q as the feedback; and we can directly as the J of q. So, either way we can do, so more commonly when we try for real time system. The q is easily obtainable, then we can consider as feedback.

But, if you think about strictly, so we have to do this; but we are more commonly using in this aspect. So, where we assume q is known to us; in the sense j of q inverse is exist to us. I already told strictly we can write this way, so J of q desired inverse into, so mu desired dot. So, this is the equation so for that we have to have; so, the inverse kinematics that will give. So, the q desired, so q desired will give. So, that is what we suppose to take as feedback.

But, more commonly the robotic community people, so they take it q; and then correct it the J. So, either way we can do it, so strict inverse differential kinematics means this; but you can use it either way around. So, I have used this one more commonly which was used in the book; so that is what I have written it here.

(Refer Slide Time: 06:33)





So, now we will move to the kinematic level control, where the joint space is the consideration. So, then the desired would be joint position, and joint velocity which nothing but q desired and q desired dot. So, if it is a set-point control, the q desired dot would become 0. So, then what one can thought about it? So, we can see what are the things supposed to be available?

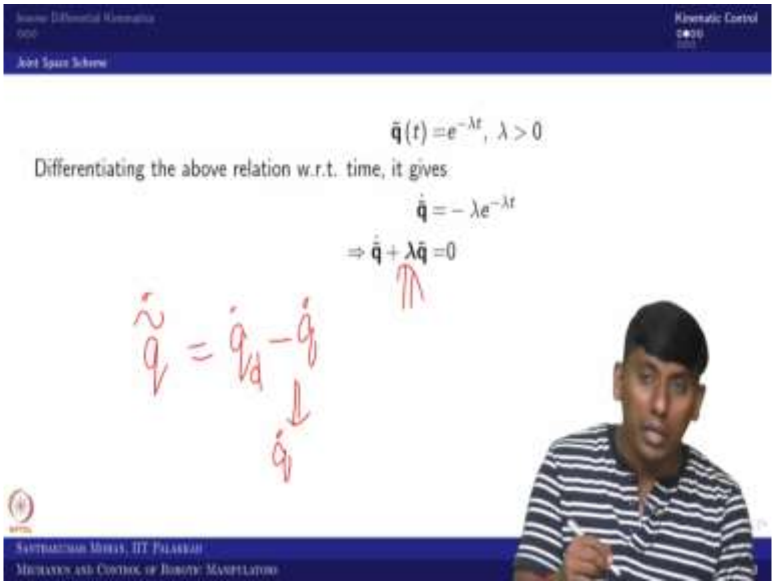
We if we think about kinematic or motion level control; the q of t which is nothing, but the actual joint position supposed to be available. Further, so if we are moving the spaces, so we may require Jacobian. But, if we talk about joint space, this may not be directly required. So, then what we are trying to find out? We are trying to find out what would be your actuator speed; so, joint velocity should be known. So, for that what we are trying to use?

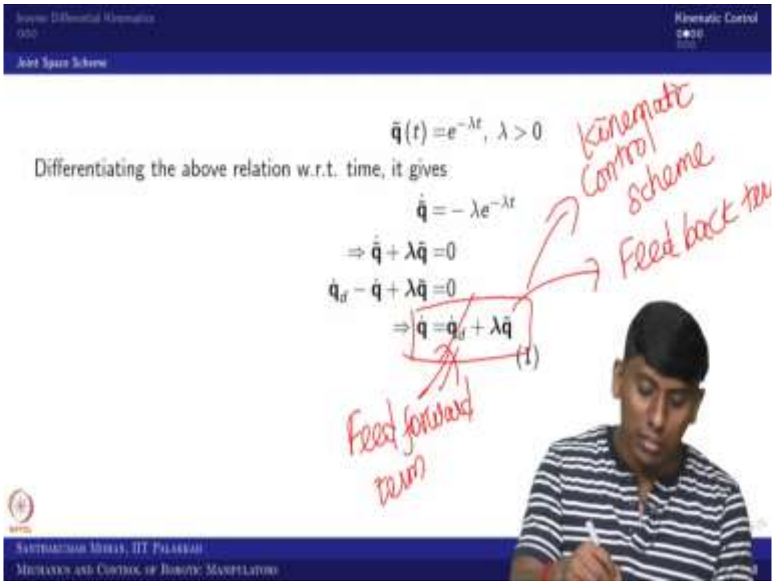
We are trying to say that some objective, where q desired minus q as I call q tilde. This q tilde supposed to go to 0, when t tends to infinity. So, this is what my objective, if I assume this is the objective; so, what I can choose. So, q tilde I can choose something like an exponential, some positive constant with t; if I assume as scalar, I can take as lambda.

Otherwise, I have to take lambda as a matrix with a diagonal value; but right now, I assume that this is the case. So, then what one can feel it? So, you can do it this in a simple aspect. So, for that what we are saying that it is exponentially stable. When t tends to infinity, the q tends to q desired; so, q tilde supposed to goes to 0, so what we define.

So, the q tilde we define in this way; so, where q tilde I am defining as q desired minus q. So, so that this can be incorporated, provided the lambda should be a positive constant.

(Refer Slide Time: 08:41)



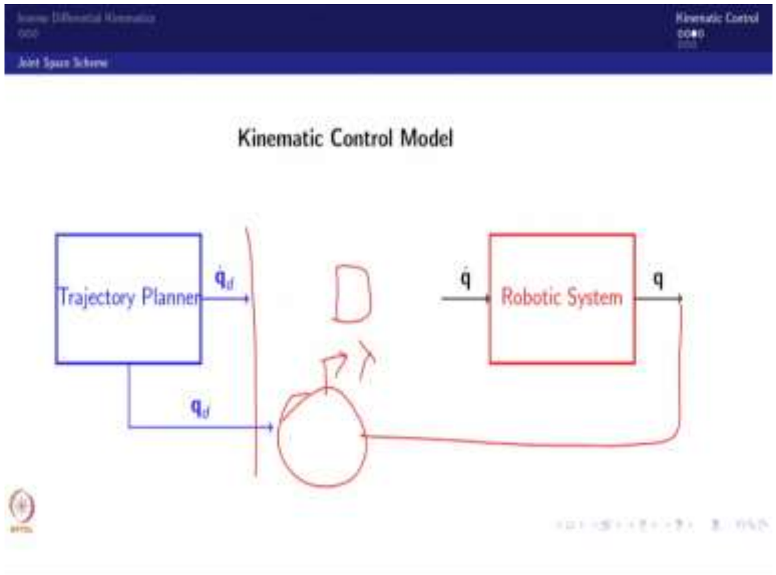


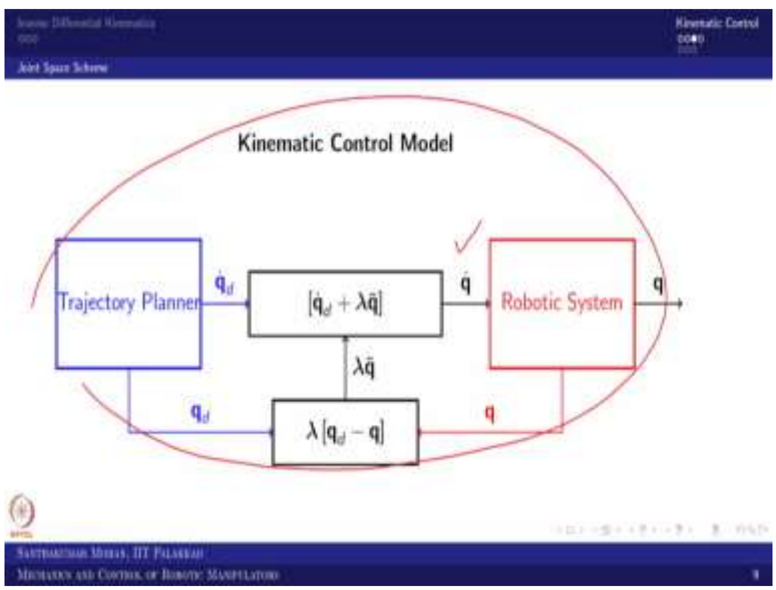
So, now in that case so we take this, and we differentiate it. So, what it will come? So, q tilde dot equal to minus lambda into e exponential of minus lambda t. But what is this? This is nothing but q tilde. So, now we can substitute that, so then I rewrite. So, q tilde dot plus lambda q tilde equal to 0. It is a simple first order system, where the lambda is a positive constant; so, then the error dynamics is like stable, it is like tends to zero.

So, that is what we are trying to do. So, now the q tilde dot I can write as q desire dot minus q dot; so, then the q dot is like required; I can rewrite, so, that what I am writing it here. So, now in that case the q dot I can write it as this; so, this is what my control law, so this is the control scheme. I can say this is a kinematic control scheme.

So, this is like feed, so the forward term; and this is like feed back term. So, in that case even the q desired dot is 0; or you do not know still this control will try to cope up and go further. So, in order to understand that what we are assume that this is not note, and we try to do a simple proportional control.

(Refer Slide Time: 10:14)



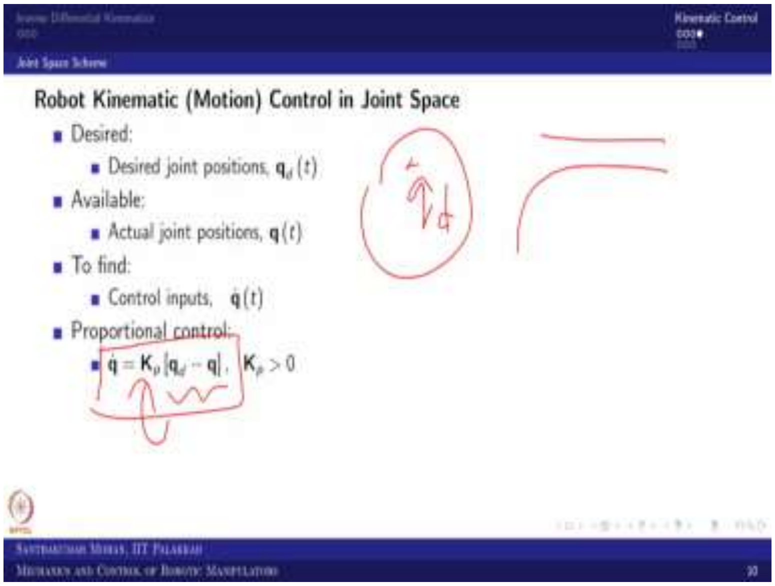


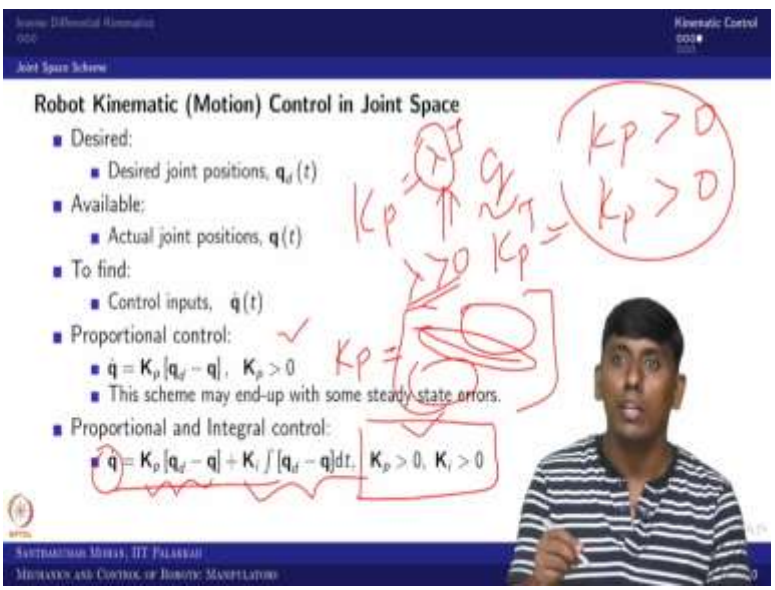
So, that is what the next idea. Before that we will see the same thing in a block diagram based; so where the robotic system will give these two. So, this is the output, and this is the input. So, now we are trying to see what would be the trajectory planner will give; since it is in a joint space. So, it will give q desired dot and q dot; sorry, q dot desired and q desired.

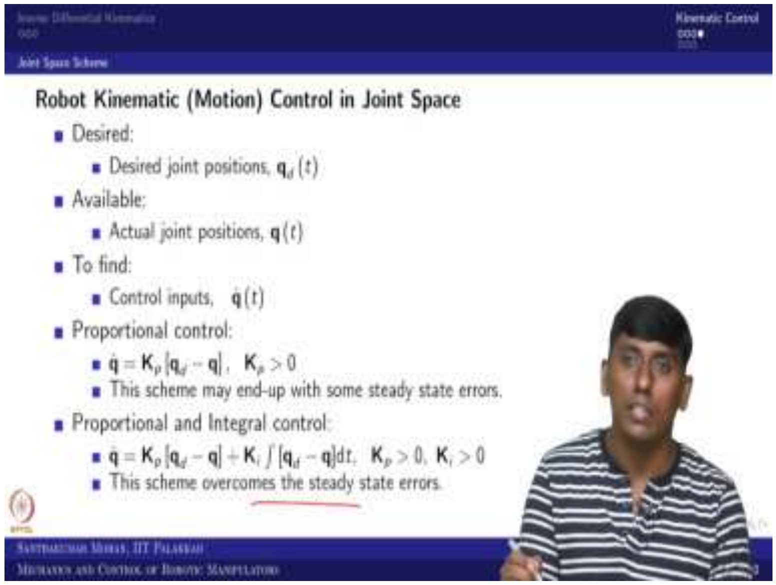
So, these all are input, and I take this and calculate the error, and multiply with lambda. So, then I can add this; so that is what I am trying to do. So, now I calculated the error; this is nothing but q tilde. So, then I can feed it that to some block; so, where q desired dot plus lambda q tilde would be the q dot as the input.

So, this is what we have seen in the previous; so, this is what we can say the kinematic control model. The same thing can be modified for the task space; but before going to see the task space.

(Refer Slide Time: 11:20)







We can see how we can expose to simple feedback with a without even feed forward. So, for that we assume the initial thing all are like same; only thing the q desired dot is like is not available to us. So, then what one can do? We can simply calculate the error; and we can make it a simple proportional constant.

Then we can consider this is the control law, this will work. Sometimes what happened this q desired dot is not there; so, what happened this would give some kind of steady state error. In order to avoid this steady state error, so what people try to do it? So, the q tilde goes to a second order dynamics in integral level. So, then we will bring the proportional and integral control; so, then that can be overcome.

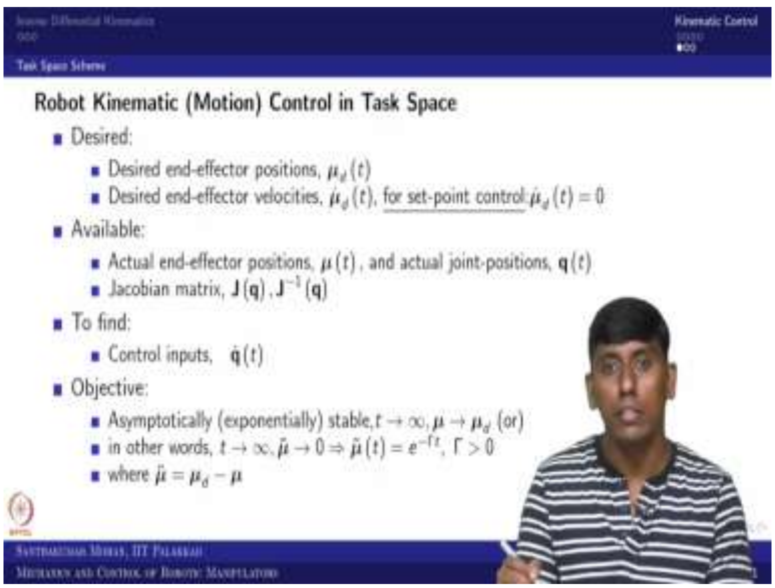
So, this is what we can say, so this may end up with some steady state error, as long there is a q desired dot is some value, so non-zero. Then this may end up with a steady state error, so in order to overcome that; so, people come up with the proportional and integral control. So, then you can see this is the way we can write it; so, then you can see that the q dot can be written as a proportional control and integral control.

So, further it provides that both control gains are like positive; so that is what you need to know. Since, it is q is a vector, so then this matrix suppose to be positive definite matrix. More or less like people try to use this Kp is positive; in addition to that this is symmetric, so this is like further extended. But most of the robotic researchers that too like when apply in real time; so they take this in a diagonal value.

So, positive values in the diagonal only, so off diagonal all are zero; so that way they take it. So, either Kp or Ki this is also one standard. So, some people some people even use to the other way around; so Kp take as the simple some scalar value into identity matrix; so, this is also possible. So, this lambda is a positive constant, provided this is a positive constant; then this control will work very well.

But the steady state error may vary based on the lambda choice or based on the positive definite matrix. So, this is what we can see as a kinematic level control; we will go to the dynamic sorry kinematic level. But, in task space, so for the task space what we can see this overcomes the steady state error.

(Refer Slide Time: 14:13)

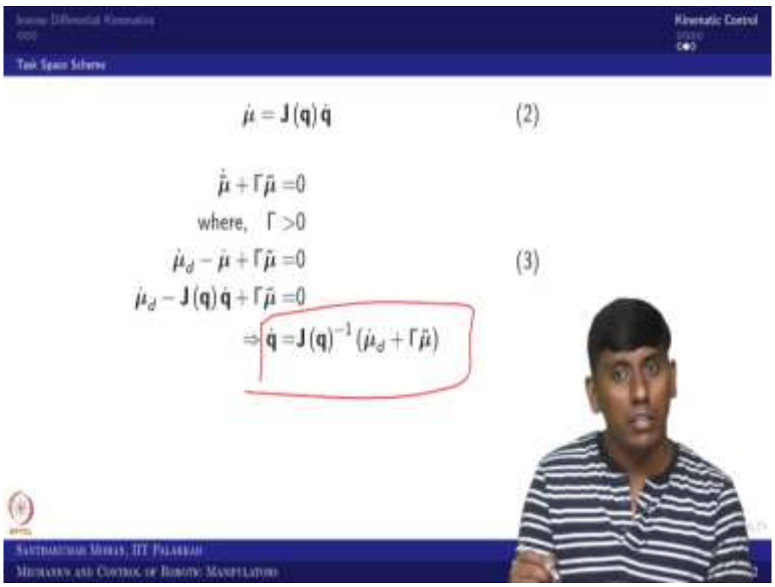


For the task space, what would be given? So, instead of desired joint position; it would be given desired end effector position; desired end effector velocity. And if it is a set-point, the desired end effector velocity would be 0. Then what we are expected? It would be available, so the actual end effector position and so actual joint position.

Because sometime the inverse kinematics is not possible straight away; so, then people try to use the actual joint position. So, in that case what one can see? So, the Jacobian matrix is necessary here. So, Jacobian and Jacobian inverse also exist; because this supposed to be exist, so, this determinant is non-zero. Or, if it is rectangular, then we have to see this J plus of q is exist, pseudo inverse suppose to exist.

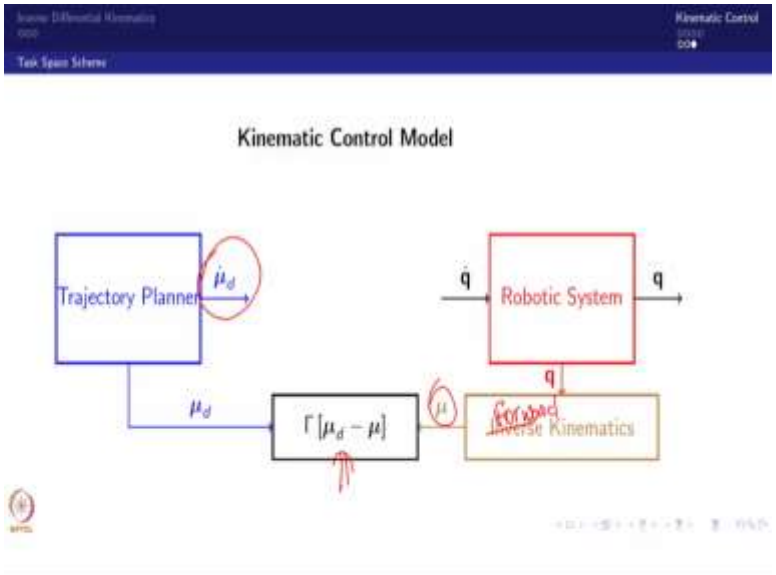
So, then what we need to find out the same thing we need to find out the q dot of t. So, how we can find the q dot of t? The same principle we will use; the objective here is mu tilde tends to 0, when t tends to infinity. So, or the other way around mu tends to mu desired, when t tends to infinity. So, if you apply mu tilde I can write it as, so exponential of minus gamma into t, where the gamma is positive constant. And again, mu tilde I have defined as mu desired as mu desired minus mu.

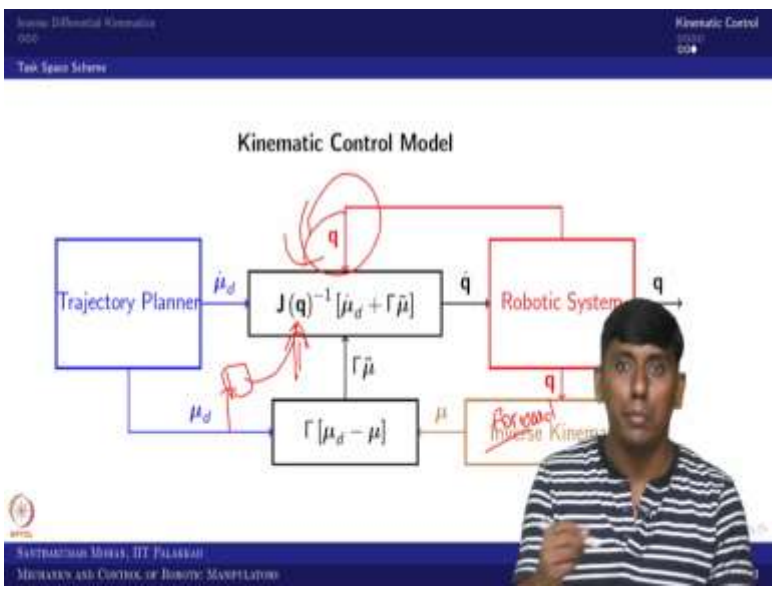
(Refer Slide Time: 15:46)



So, based on this I can write this equation. So, then this is known, and this is the equation which we will be obtaining based on that. So, then we can provided put the gamma is like positive; then we can get this. So, from there mu dot we can substitute as; so, this relation we can substitute; then we can find the q dot in this way. So, now this is the control law, so which is for the task space level. So, I have make it the same thing block diagram based; so only there is was small error.

(Refer Slide Time: 16:23)





So, I will correct it in the handout. So, the robotic system will give the output as q and input as q dot. So, then we can see the trajectory planner; the trajectory planner will give the mu desired dot and mu desired, so, now we need mu. So, mu we have to do it in a forward kinematics; so, this I have written as inverse; so this is written as forward kinematics.

Please correct it; I will correct it in the handout. So, once you know this forward kinematics, so the mu would be obtained; so, then calculate the mu tilde. So, once you calculate the mu tilde and mu desired dot is known; so, what we can do? We can make Jacobian inverse into. So, mu desired dot plus gamma into mu tilde; that is what we are trying to do.

So, this is so what we used it. So, so what one addition? So, you need this q, because this J of q is required. Some people here again the use from here the inverse kinematics and then this desired. But I prefer this is feedback, so better we can use q; so, then we can make it, so, that is what the whole idea. So, now I hope you are clear, so what is kinematic control and what all about task space scheme and joint space scheme?

And in the next lecture we will see some numerical examples along with a MATLAB simulation; so, that you would be getting more clarity on this kinematic control. Once that is clear to us, then we will move to dynamic control; and till then see you, bye, take care.