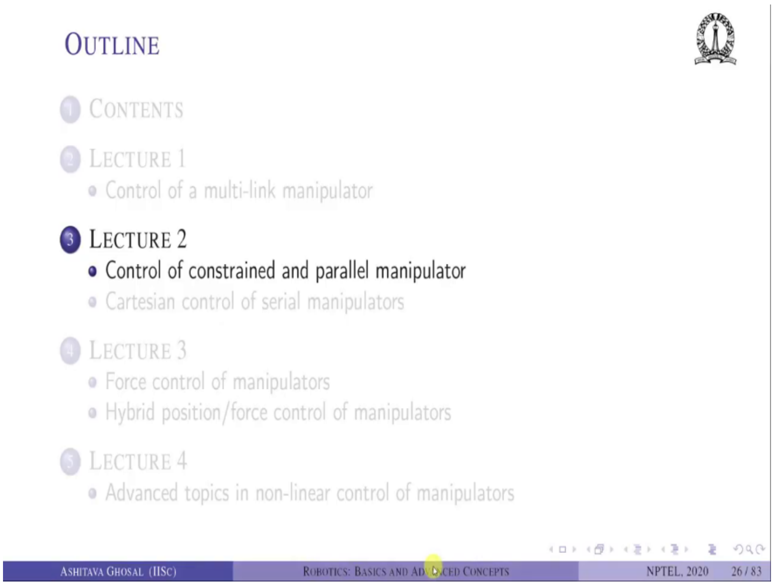
Robotics: Basics and Selected Advanced Concepts
Prof. Ashitava Ghosal
Department of Mechanical Engineering
Indian Institute of Science, Bengaluru

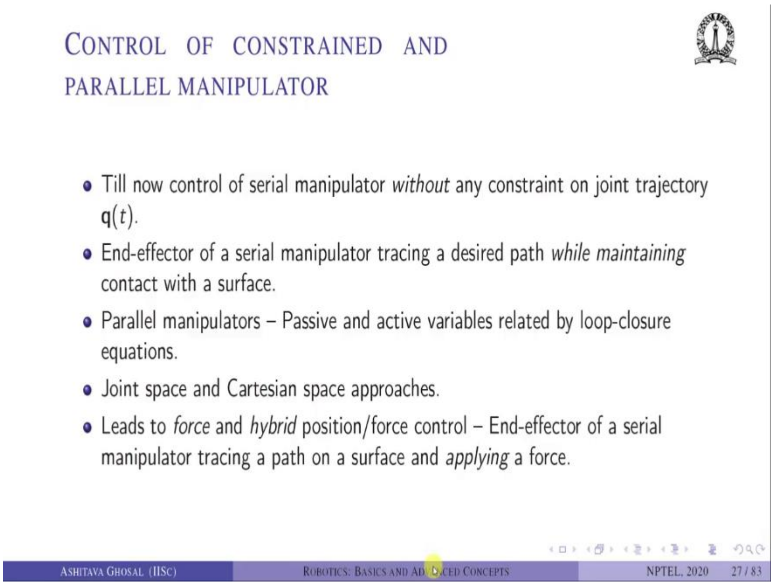
Lecture - 35
Control of constrained and parallel manipulator, Cartesian control of serial manipulators

(Refer Slide Time: 00:20)



Welcome to this NPTEL lectures on Robotics: Basic and Advanced Concepts. In this lecture, we will look at Control of constrained and parallel manipulators. In the previous lecture, we looked at control of multi link serial manipulators. The second topic which we will look at in this lecture is Cartesian control of serial manipulators ok.

(Refer Slide Time: 00:47)



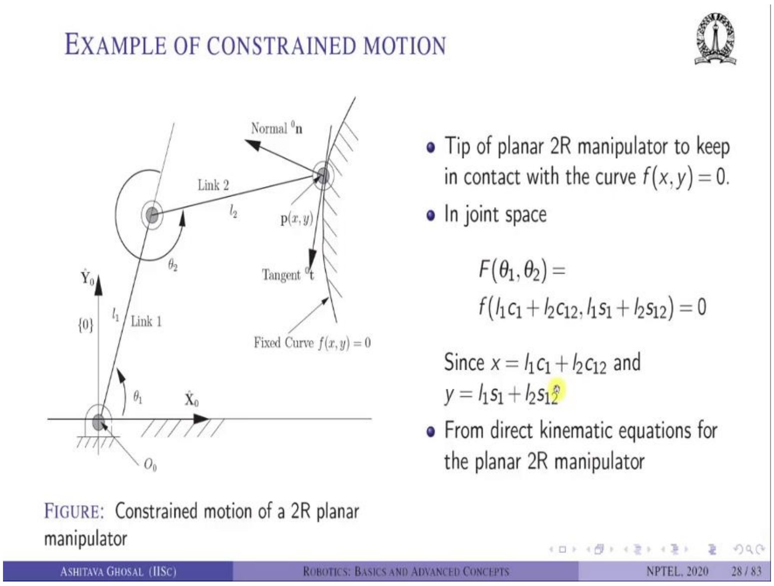
So, let us start. So, till now the control of serial manipulator without any constraint on joint trajectory was considered ok. So, we looked at $q_d(t)$ is given desired trajectory and then we tried to control the robot to achieve that $q_d(t)$. However, there was no inherent constraint on $q_d(t)$ or and $q(t)$, in many instances the end effector of a serial manipulator needs to face a desired path while maintaining contact with a surface ok.

So, for example, you can think of a robot carrying let us say a grinding tool, it wants to grind the surface ok, so it has to trace a path on that surface on which it is grinding, at the same time the grinding tool must press on the surface with some desired force ok. So, this is the topic of control of constrained serial manipulators.

Similar to this control of constrained serial manipulators, we also have parallel manipulators in which the passive and active joint variables are related by loop closure constrained equations ok. So, there also some constraints appear, but as you will see later it is different than control of constrained serial manipulator ok.

So, we can control the motion of a serial manipulator with some constraint either in joint space or Cartesian space ok. And as we will see later this leads to force and hybrid position force control of a serial robot ok. So, ultimate thing is as I gave the example we want the end effector of a serial manipulator tracing a path on the surface and also applying a desired force.

(Refer Slide Time: 02:55)



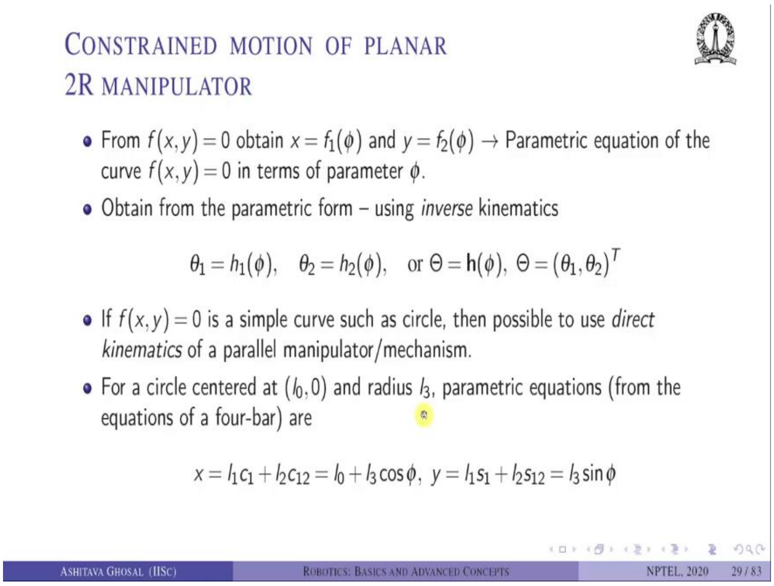
So let us look at this example of a constrained motion. So, we have this planar 2R manipulator so, this is θ_1 and θ_2 this is the X and Y axis this is the origin of the first coordinate its fixed coordinate system and we have link 1 and link 2. So, what we want is this point end effector point $P(x, y)$ to trace a curve which is given by $f(x, y) = 0$. So, this is the fixed curve in 3D space ok.

So, if you have a fixed curve in 3D space at any instant we will have a tangent vector along this tangent and a normal vector to the curve ok. So, in joint space what we what do we have? We have $f(x, y) = 0$ can be converted to some $F(\theta_1, \theta_2) = 0$ ok, how? Because $f(x, y)$ in the case of a 2R manipulator $x = l_1 \cos\theta_1 + l_2\cos(\theta_1 + \theta_2)$.

We have looked at the direct kinematics of a planar robot and we can obtain the x and y of the end effector point.

So, $x = l_1c_1 + l_2c_{12}$ and $y = l_1s_1 + l_2s_{12}$. So, this $f(x, y) = 0$ can be rewritten as some $F(\theta_1, \theta_2)$; and this is simply because we know the direct kinematics equations, for the planar 2R manipulator.

(Refer Slide Time: 04:35)



So, from $f(x,y) = 0$, we can also try and obtain x and y as a function of a parameter, ϕ ok, most of the time the equation of a curve in a plane can be written in terms of a parameter ϕ or in parametric form ok.

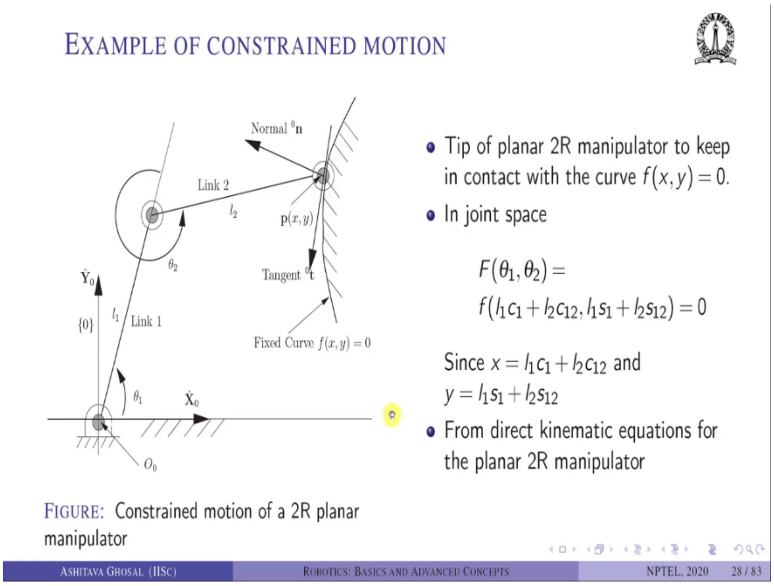
So, once we obtain x and y in terms of the parameter, we should be able to at least conceptually use inverse kinematics and find θ_1 and θ_2 as functions of that parameter ϕ.

So, basically what do we have? Not so easy to do all the time, but conceptually we can see θ_1 will be related to ϕ, θ_2 will also be related to ϕ again using inverse kinematics ok, or we will write this Θ vector as some vector function of ϕ.

So, this Θ vector this $(\theta_1, \theta_2)^T$.

So, if $f(x,y) = 0$ is a simple curve such as a circle, we can also use direct kinematics of a parallel manipulator or mechanism ok. So, what do we mean by this? So, consider that this curve is given as a circle centered at $(l_0, 0)$ and radius l_3.

(Refer Slide Time: 06:04)



So, this is just an example that this curve is nothing but a circular arc of radius l_3 and centered at $(l_0, 0)$. So, we have a point $(l_0, 0)$ with a radius this.

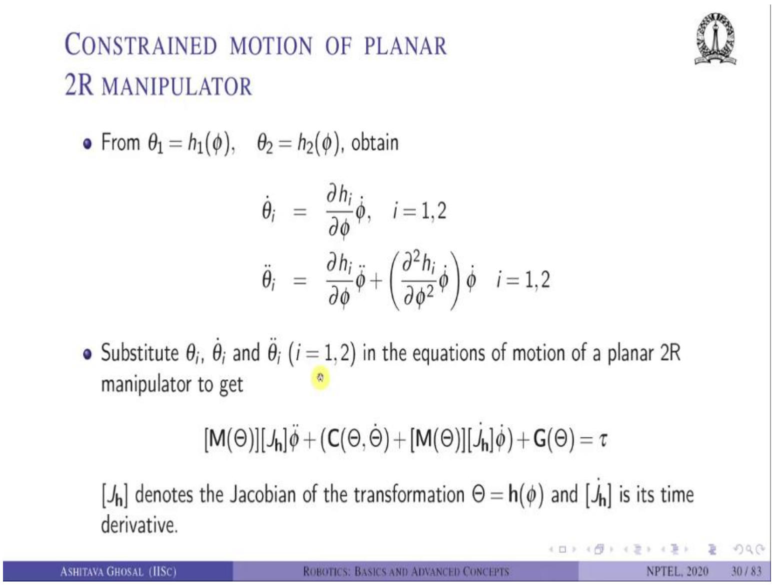
So, basically what do we have? We have one link, two link and then this point connected to the fixed point by another link.

So, it becomes a four bar mechanism ok. So, sometimes in four bar mechanism we can write $x = l_0 + l_3 \cos\phi$, $y = l_3 \sin\phi$ which is basically what we want. We want x as a function of a parameter ϕ and y as a function of parameter ϕ.

So, instead of worrying about θ_1 and θ_2 we can directly write in terms of a parameter ϕ, no big deal because we are just trying to find the equation of a circle.

The equation of a circle can be always written as in terms of cos and sin of an angle with some center (a, b) so, $a = l_0$, $b = 0$.

(Refer Slide Time: 07:13)



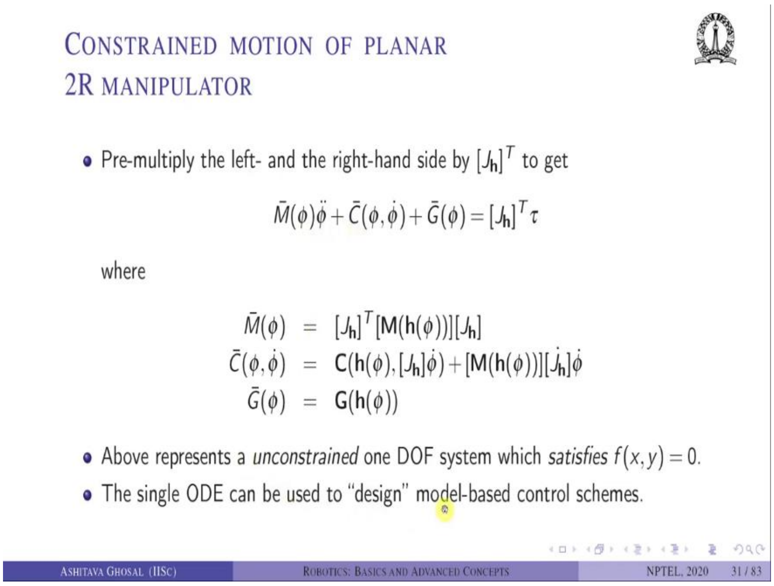
So, from $\theta_1 = h_1(\phi)$ and $\theta_2 = h_2(\phi)$. So, these two functions of ϕ, we can find $\dot{\theta}_1$ and $\ddot{\theta}_1$. So, basically nothing but use chain rule we take the partial derivatives of h_i with respect to ϕ times $\dot{\phi}$ that is $\dot{\theta}$. $\ddot{\theta}$ is take the derivative again.

So, we will get one term which is $\frac{\partial h_i}{\partial \phi}\ddot{\phi} + \left(\frac{\partial^2 h_i}{\partial \phi^2}\dot{\phi}\right)\dot{\phi}$. So, we can substitute this θ_i, $\dot{\theta}_i$, $\ddot{\theta}_i$ for $i = 1{,}2$ in the equations of motion. So, remember the equations of motion of the planar 2R will contains $[M]\ddot{\Theta}$ plus Coriolis term plus gravity term is equal to τ_1 and τ_2.

So, now theta and $\dot{\theta}_i$ and $\ddot{\theta}_i$ can be substituted. So for example, $\ddot{\Theta} = [J_h]\ddot{\phi}$, $\dot{\Theta} = [J_h]\dot{\phi}$. So, we can reorganize this equation as $[M(\Theta)][J_h]\ddot{\phi} + \left(C\left(\Theta, \dot{\Theta}\right) + [\mathrm{M}(\Theta)]\left[\dot{J_h}\right]\dot{\phi}\right) + G(\Theta) = \tau$.

So, think about it is not very hard, basically what we have done is we have taken θ as a function of this parameter ϕ ok, which was obtained either using inverse kinematics or using analogy to a four bar mechanism, if the curve was very like an arc of a circle and then, we convert everything from θ_1, θ_2, $\dot{\theta}_1$, $\dot{\theta}_2$, $\ddot{\theta}_1$, $\ddot{\theta}_2$ into thisϕ terms ϕ, $\dot{\phi}$ and $\ddot{\phi}$ and so on ok. So, $[J_h]$ denotes the Jacobian of the transformation θ to $h(\phi)$ and $\left[\dot{J_h}\right]$ is its time derivative straightforward.

(Refer Slide Time: 09:37)

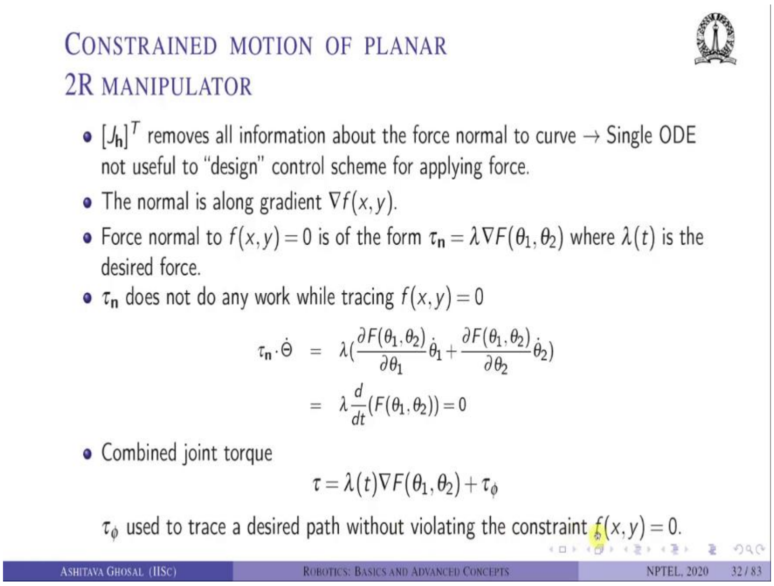


So, now we can pre multiply that previous equation by $[J_h]^T$. So, what happens when we pre multiply by $[J_h]^T$? So, we will have $\bar{M}(\phi)\ddot{\phi} + \bar{C}(\phi,\dot{\phi}) + \bar{G}(\phi) = [J_h]^T\tau$. So, what is this $\bar{M}(\phi)$? It is $[J_h]^T[M(h(\phi))][J_h]$.

So, $\bar{C}(\phi,\dot{\phi}) = C(h(\phi),[J_h]\dot{\phi}) + [M(h(\phi))][\dot{J_h}]\dot{\phi}$ an so on. So, what you can see here is this equation is nothing but an unconstrained 1 degree of freedom system, which satisfies $f(x,y) = 0$. We have converted first from x, y to θ_1, θ_2 and then again we have converted θ_1, θ_2 to ϕ and we have derived a differential equation in ϕ and its derivatives ok.

So, this is a single one ODE ok, in terms of phi and its derivatives and this single ODE can be used to "design" model-.based control schemes. Just like in the case of serial robot so, we have managed to get rid of the constraint that the tip of the robot traces that curve $f(x,y) = 0$.

(Refer Slide Time: 11:12)



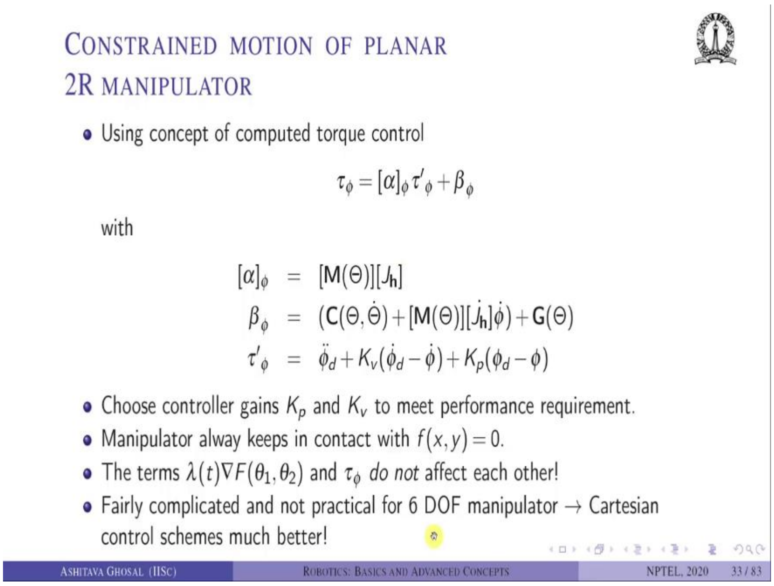
So, what have we done? Basically $[J_h]^T$ removes all information about the force normal to the surface. So, it is a single ODE not very useful to "design" controlled schemes for applying force ok. So, this quantity here this is like some kind of a projection. So, we are projecting from θs to ϕ ok. If you want to include the force which is normal to the surface, we can compute the gradient which $\nabla f(x, y)$.

The force normal to $f(x, y) = 0$ is of the form τ_n and proportional to this gradient. So, $\lambda(t)\nabla F(\theta_1, \theta_2)$, we do not want to write in terms of x and y we can go back right in terms of joint variables ok, so, where $\lambda(t)$ is the desired force. First thing you can see is this τ_n does not do any work while tracing $f(x, y)$, why? Because, $\tau_n \cdot \dot{\Theta}$.

So, work done by this normal force can be written in terms of $\lambda\left(\frac{\partial F(\theta_1,\theta_2)}{\theta_1}\dot{\theta}_1 + \frac{\partial F(\theta_1,\theta_2)}{\partial \theta_2}\dot{\theta}_2\right)$ and this is nothing but $\lambda \frac{d}{dt}\big(F(\theta_1, \theta_2)\big) = 0$ now, this itself is 0 ok. So, this is the equation of the curve in terms of θ_1, θ_2.

So, the normal force does not do any work while the tip is tracing the curve on this $f(x, y) = 0$. So, we can have a kind of combined force which is τ is λ times the gradient of this function plus τ along the ϕ direction or along the tangent ok. So, we can use this τ_ϕ to trace a desired path without violating the constraint $f(x, y) = 0$.

(Refer Slide Time: 13:25)



So, we go back to our basic notion of a computed torque control scheme. We can write $\tau_\phi = [\alpha]_\phi \tau'_\phi + \beta_\phi$, but what is the $[\alpha]_\phi$? that is $[M(\Theta)][J_h]$, $\beta_\phi = \left(C\left(\Theta,\dot{\Theta}\right) + [M(\Theta)]\left[\dot{J_h}\right]\dot{\phi}\right) + G(\Theta)$ and $\tau'_\phi = \ddot{\phi}_d + K_v\dot{e} + K_p e$, where $e = (\phi_d - \phi)$.

So, very similar ideas to what we did for the serial robot and computed torque control scheme. So, we can choose this controller gains K_p and K_v to meet the performance requirements. So, if you want overdamped so, we have to choose $K_v = 2\sqrt{K_p}$ and so on ok.

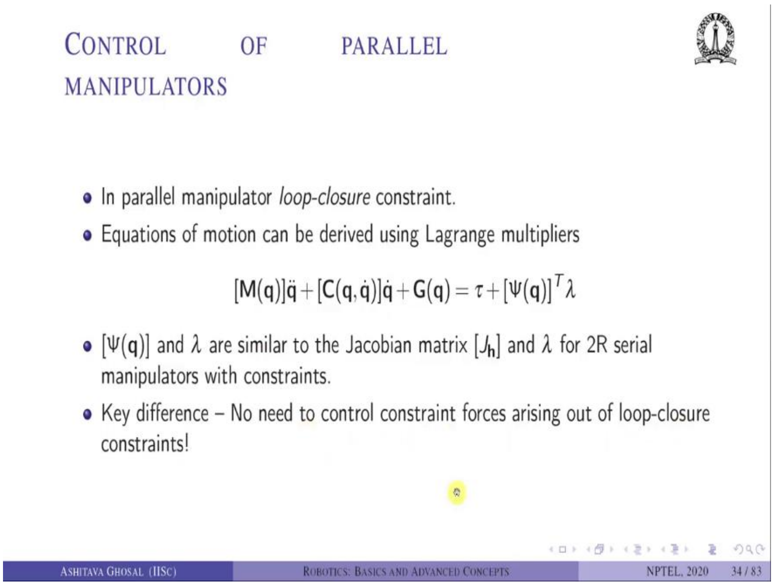
And this way we make sure that the manipulator always keeps in contact with $f(x,y) = 0$ why? Because, $f(x,y) = 0$ is always satisfied. We have derived this ϕ based on $f(x,y) = 0$. The terms $\lambda(t)\nabla F(\theta_1,\theta_2)$ and τ_ϕ do not affect each other because, one is in normal to the surface one is tangent to the surface ok.

So, what do we have? We have a controller which looks like this that the torque input is proportional to the gradient and plus the tangent direction τ_ϕ, where τ_ϕ is given by $[\alpha]_\phi \tau'_\phi + \beta_\phi$ like any other model based torque and then we can choose K_p and K_v to satisfy the performance requirements ok.

So, as we can see this is fairly complicated ok, it is not really practical for the 6 degree of freedom manipulator. If you have 6 degree of freedom manipulator I will not be able to get rid of these or obtain this ϕ and then get rid of θ_1 though θ_6 and do inverse kinematics for this complicated 6 degree of freedom robot ok.

But, we will see later that we can use something called as a Cartesian control schemes, remember we have done one simulation with Cartesian control and it is much much easier to do it using Cartesian coordinate.

(Refer Slide Time: 15:57)



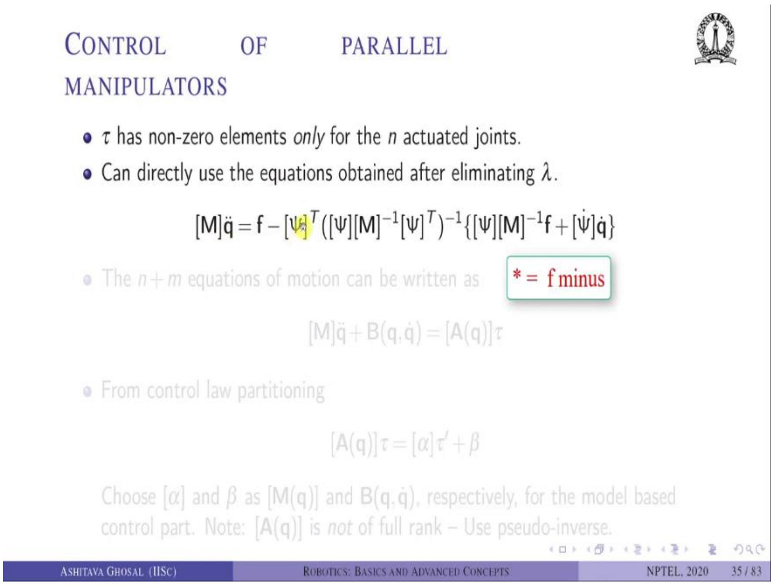
So, in parallel manipulator, the loop closure constraint is given by this ok, we have loop closure constraints. So, it is some $\eta(q)$ which is $\eta(\theta, \phi) = 0$, we can obtain the equations of motion in terms of Lagrange multipliers and q contains both actuated and passive joint variables, this $[\Psi]$ and λ are similar to the Jacobian matrix $[J_h]$ and λ for the 2R serial manipulators with constraints ok.

So, it is sort of similar idea, we have some variables which are related to the some other variables using that loop closure constraint equation. In the case of this 2R serial robot we are x and y related by $f(x, y) = 0$ and then we obtained this Jacobian of this h function, where h_1 was θ_1 related to some $h_1(\phi)$ and so on. So, similar ideas are occurring here which is why we are also considering control of parallel manipulators in the same lecture.

The key difference though is that we do not need to control constraint forces rising out of loop closure constraints ok. So, in the case of a 2R robot, we also wanted to control the force which is normal to the surface and if you think about it, the force which is normal to the surface happens because we are trying to trace a path on the surface or you are trying to satisfy that constraint ok.

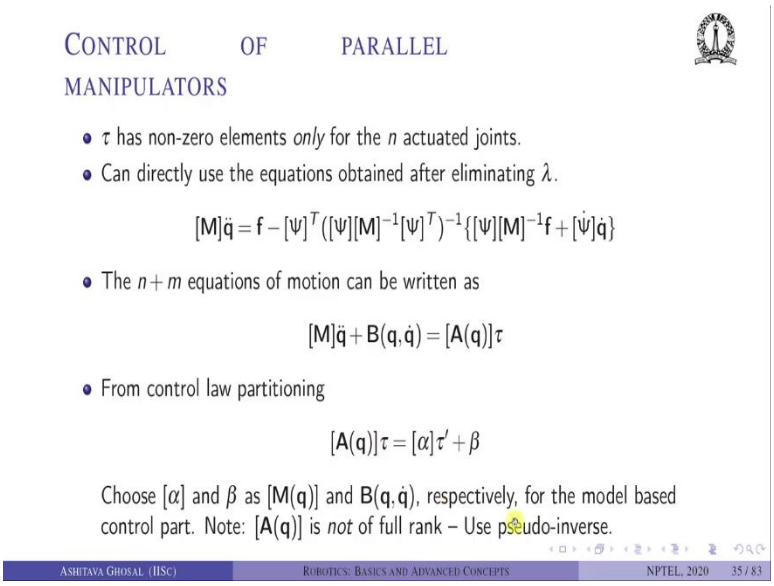
However, in parallel robots there is no such issue or notion trying to satisfy the constraint forces corresponding to the passive joints, which are related to the active joints using the loop closure constraints ok.

(Refer Slide Time: 17:53)



So, what do we have? We have τ as only non-zero elements only for the n actuated joints ok. So, again you can directly use the equations after eliminating λ. So, the equations of motion of a parallel robot remember we have obtained this as $[M]\ddot{q} = f - [\Psi]^T([\Psi][M]^{-1}[\Psi]^T)^{-1}\{[\Psi][M]^{-1}f + [\dot{\Psi}]\dot{q}\}$, where $[\Psi]$ was the matrix of $[K]$ and $[K^*]$ derived from the partial derivatives of the constraint equation; and $[\dot{\Psi}]$ was the derivative of the $[\Psi]$ matrix, the constraint matrix.

(Refer Slide Time: 18:42)

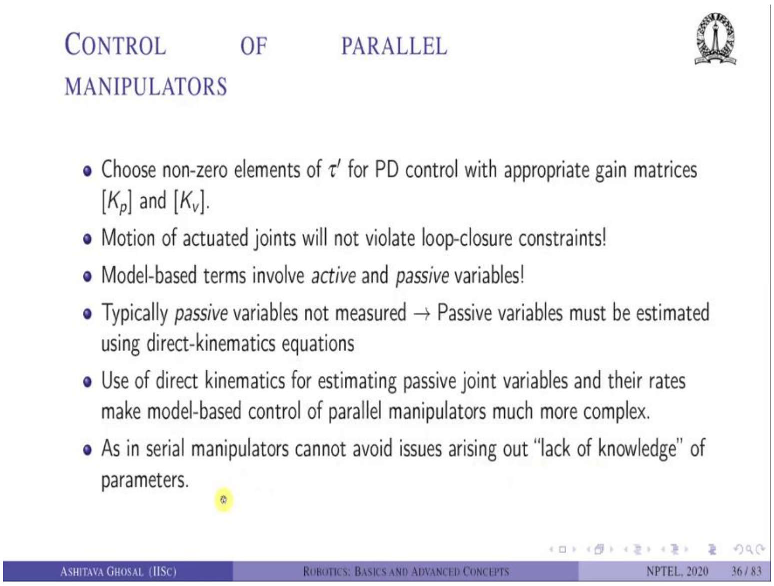


So, we had $m + n$ second order differential equations here after eliminating the λ and these $m + n$ equations of motion can be written as $[M]\ddot{q} + B(q, \dot{q})$ everything taken this side except this τ part which is in f, and if and we can write it at least conceptually as $[A(q)]\tau$. So, in the case of a normal serial robot we have $[M(q)] + C(q, \dot{q}) + G(q)$ and so on, which is all here and equal to τ, but now we have some $[A(q)]\tau$.

So, again from control law partitioning, we can now consider $[A(q)]\tau = [\alpha]\tau' + \beta$ and then we can choose $[\alpha]$ and β as $[M(q)]$ and $B(q, \dot{q})$ respectively. For the model based control part, one thing to note is this $[A(q)]$ is not of full rank, why? Because all the τ's are not there some of the τ's are 0 ok, the τ corresponding to the passage joint variables in q are not there are 0.

So, how do we find τ finally? The actuated joints are we need to use some kind of a pseudo inverse.

(Refer Slide Time: 20:05)



So, we choose non-zero elements of τ' for PD control with appropriate gain matrices $[K_p]$ and $[K_v]$ ok, τ' also does not contain all the τ's. The motion of the actuated joints will not violate the loop closure constraints, because we have used the loop closure constraint equations to derive the control scheme ok. The model based term involve active and passive variables so, $[M(q)]$. So, q contains both θ and ϕ, the actuated variables as well as the passive joint variables ok.

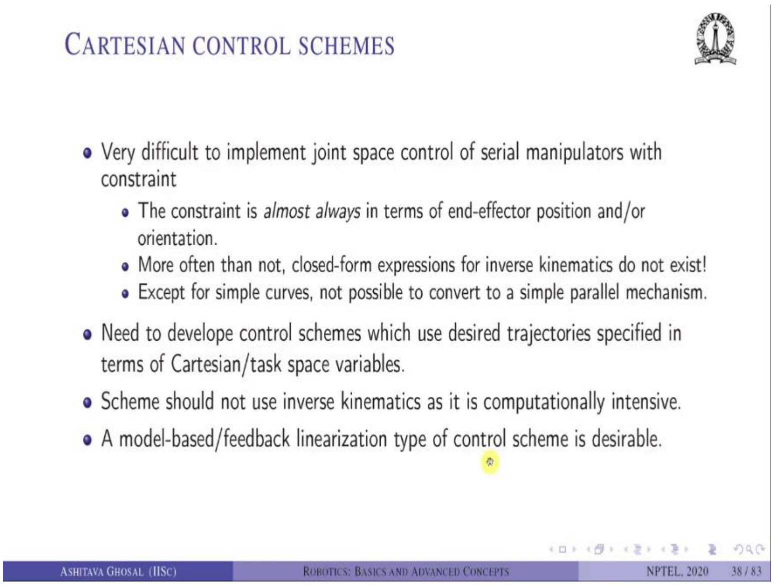
So, typically passive joint variables are not measured ok, why would you want to spend money on sensors and measure that passive joint variables. However, the passive joint variables can be estimated using direct-kinematics equation ok. So, the use of direct kinematics for estimating passive joints variables and their rates make the model based control of parallel manipulators much more complex ok.

Because see, we can measure θ but, I have to solve direct kinematics I have to estimate the passive variables and their rates, which is $[K]\dot{\theta} + [K^*]\dot{\phi} = 0$. Then, I have to worry about $[K^*]$ being not singular all kinds of complications arise, when we are trying to use model based control for parallel robots ok.

And also as in serial robots we cannot avoid problems or issues arising out of "lack of knowledge" of parameters ok. So, we will not know the link lengths there are many more link lengths, we do not know the friction of the joints both active and passive joints. So, estimating all these dynamics is much more complicated and causes issues ok.

So, as I said the joint based control of a robot trying to trace a path on a surface or basically constrained motion is difficult ok, especially for 6 degrees of freedom robot. So, this is done much easier using something called Cartesian control of serial manipulators. So, which is what we will discuss now.

(Refer Slide Time: 22:28)



So, as I mentioned it is very difficult to implement joint space control of serial manipulator with constraints ok. The constraint is almost always in terms of end effector position and or orientation so; the tip must trace some path. And more often than not, closed form expression for inverse kinematics do not exist ok, only for things like PUMA or robot with last three joints intersecting.

We can hope to get some expressions for the joint variables otherwise, it is some 8th degree or 16th degree polynomial and we have to use numerical techniques.

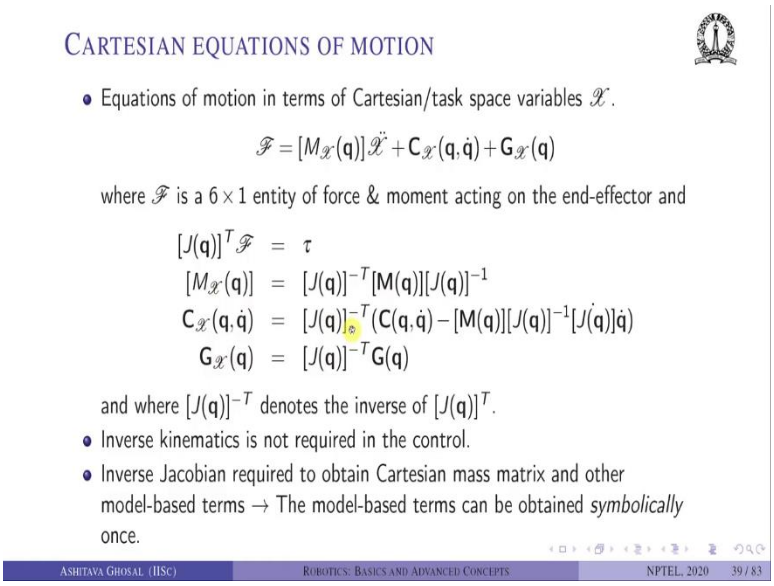
So and also for simple curves, it is not possible to convert to a simple parallel mechanism. So, for the curve as a circle centered at one point on some point of the X axis, we saw that it could become a four bar mechanism.

And hence again we can get rid of θ_1 and θ_2 in terms of a parameter ϕ, which is the rotation of this fictitious link ok. But that is not possible to do for complicated curves.

So, we need to develop control scheme which use desired trajectories specified in terms of Cartesian or task space variables. So, it is specified using x, y, z, the position of the end effector and the orientation of the end affector ok.

So, ideally we should not be using inverse kinematics, because inverse kinematics is computationally intensive. We would like a model based feedback linearization type of control scheme, but in terms of Cartesian variables position and orientation ok. So, can this be done? Yes ok.

(Refer Slide Time: 24:26)



So, basic is the step is that we need to write the equations of motion in terms of Cartesian or task space variables $\mathcal{X}$. So, basically $\mathcal{X}$ contains x, y, z which is a position and some representation of the orientation let us say three Euler angles. So, we can write the external force which is acting on the end effector and the moment which is this $\mathcal{F}$ script style F is $[M_{\mathcal{X}}(q)]\ddot{\mathcal{X}} + C_{\mathcal{X}}(q,\dot{q}) + G_{\mathcal{X}}(q)$.

So, the subscript $\mathcal{X}$ script X means that they are in terms of the Cartesian variables ok. This q and $\dot{q}$ can still stay there we do not need to actually use inverse kinematics, we will see later. We have already derived that this external force or moment which is acting at the end effector can be related to the joint torques by $\tau = [J(q)]^T\mathcal{F}$.

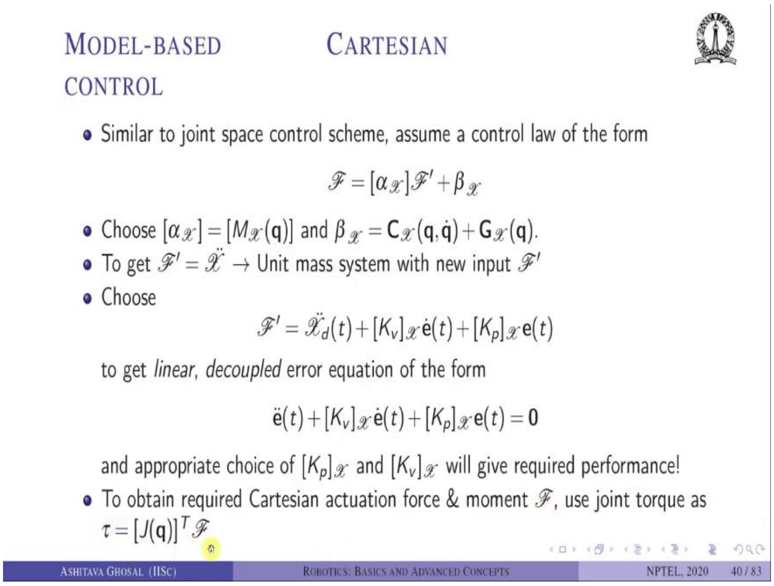
Remember in statics we looked at what is the force or the torque required at the joint to maintain equilibrium and some $[J(q)]^T\mathcal{F}$. Likewise using inverse kinematics we can show that this mass matrix in terms of the Cartesian variables can be written in terms of the mass matrix with joint variables using $[J(q)]^{-T}$ and $[J(q)]^{-1}$.

The Coriolis term in terms of Cartesian variables is of course much more complicated. So, it is $[J(q)]^{-T}\big(C(q,\dot{q}) - [M(q)][J(q)]^{-1}[\dot{J}(q)]\dot{q}\big)$ and gravity term is $[J(q)]^{-T}G(q)$. So, $[J(q)]^{-T}$ denotes the inverse of the transpose of the Jacobian matrix ok. So, the inverse kinematics is not required in the control, why? Because I am still leaving q, I am not trying to get rid of q.

I am writing q is inverse of $\mathcal{X}$, inverse kinematics of $\mathcal{X}$. We need to use inverse Jacobian to compute this left hand side ok. But, they can be computed symbolically and only once, I do not need to do it in the loop ok. So, if I have these expressions somehow. Let us say I have computed after standing lot of time in a computer using symbolic algebra.

I can obtain the mass matrix using Cartesian coordinates keeping q as it is, Coriolis term and gravity term ok. So, we have to use $[J(q)]^{-1}$. So, that requires a lot of effort, but it can be done once symbolically.

(Refer Slide Time: 27:19)



So now, let us continue we want to design our model based Cartesian controller. So, similar to the joint space control scheme we assume a control law of the form $[\alpha_{\mathcal{X}}]\mathcal{F}' + \beta_{\mathcal{X}}$, this is the external force and moment acting on the end effector. So, we choose $[\alpha_{\mathcal{X}}] =$

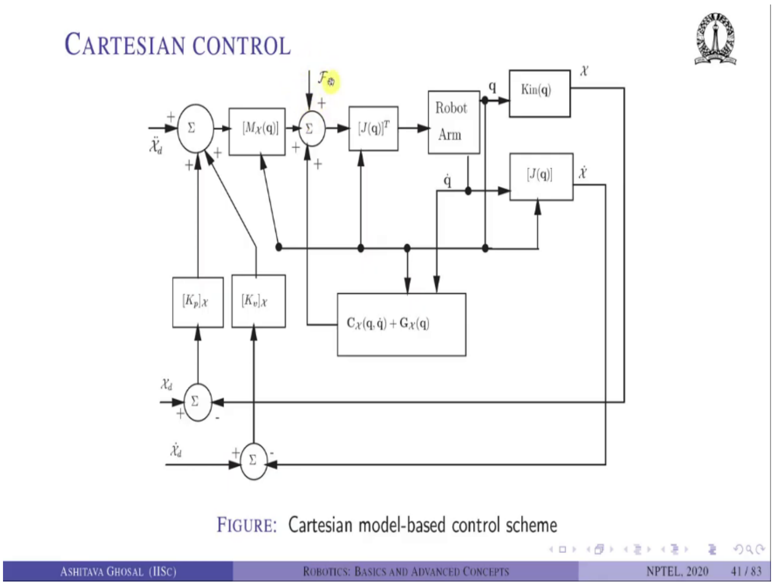
$[M_{\mathcal{X}}(q)]$, $\beta_{\mathcal{X}} = C_{\mathcal{X}}(q, \dot{q}) + G_{\mathcal{X}}(q)$ and then just like in the serial robot case we end up with $\mathcal{F}' = \ddot{\mathcal{X}}$.

And we can choose so, it is like a unit mass or a unit inertia being acted by a force $\mathcal{F}'$ or a moment $\mathcal{F}'$ contains both the translation force F and moment M. So, we choose this curly $\mathcal{F}' = \ddot{\mathcal{X}}_d(t) + [K_v]_{\mathcal{X}}\dot{e}(t) + [K_p]_{\mathcal{X}}e(t)$.

So, remember here $e(t)$ is a Cartesian error not the joint space error and we will substitute this $\mathcal{F}'$ back into this and then that into the equations of motion, we will get $\ddot{e}(t) + [K_v]_{\mathcal{X}}\dot{e}(t) + [K_p]_{\mathcal{X}}e(t) = 0$. So, we get a linear decoupled error equation of the form like this, exactly the same idea and same approach as we did for the serial robot ok, in joint space.

So, but finally we have to apply joint torques ok. There is an external force acting on the end effector, but there is no there are no rockets or something which allows you to apply forces at the end effector. So, we can only apply torques at the using motors at the joints. So, how do I convert this $\mathcal{F}$ to torque? We can use $\tau = [J(q)]^T\mathcal{F}$.

(Refer Slide Time: 29:20)



So, this is what the block diagram of a Cartesian model based control scheme look like ok. So, we start from this interior. So, this is the robot arm, the input is torque ok. So, we apply some voltage to the motors and we get some torque, we measure q and $\dot{q}$ using encoders and tachometers or some other device.

From q we can do direct kinematics, $Kin(q)$ means, direct kinematics. Using the direct kinematic formula of the robots we can get position and orientation of the end effector. So, from q and $\dot{q}$ we can do Jacobian and we can find the linear and angular velocity of the end effector.

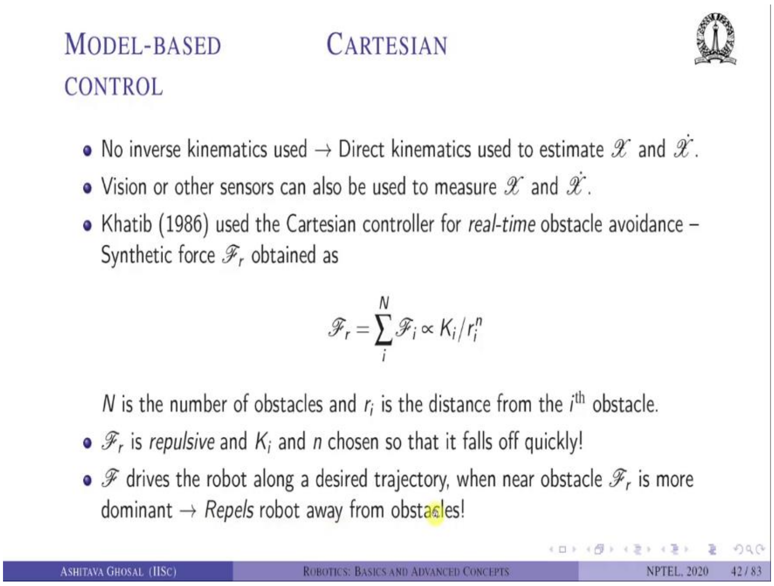
So, this is $\dot{\mathcal{X}}$ correct. Then with q and $\dot{q}$, we can substitute in this Cartesian Coriolis term and Cartesian gravity term, we can also use this q to obtain Jacobian transpose and then we can also obtain $[M_{\mathcal{X}}(q)]$ the Cartesian mass matrix.

So, this, this, this, this and this all these things are happening inside the computer ok, there are some programs which uses q and $\dot{q}$ to obtain the position and orientation of the end effector, the velocities at the end effector, the Cartesian Coriolis and gravity term, the Cartesian mass matrix and the Jacobian matrix of the robot.

So, we can also use this $\mathcal{X}$ and $\dot{\mathcal{X}}$ so, $\dot{\mathcal{X}}_d - \dot{\mathcal{X}}$ will give you this $\dot{e}(t)$ multiplied by a derivative gain $[K_v]_{\mathcal{X}}$, $\mathcal{X}_d - \mathcal{X}$ will give you the error in terms of Cartesian variables multiplied by proportional gain and we add this $\ddot{\mathcal{X}}_d$ and here we will get $\mathcal{F}'$. So, what do we have, here $\mathcal{F}'[M_{\mathcal{X}}(q)]$ plus all these things and eventually we get $\mathcal{F}$ here, which you multiply by $[J(q)]^T$ and you get torque ok.

So, as you can see in this block diagram we are never using inverse kinematics anywhere. We are using forward kinematics or direct kinematics we are also using Jacobian matrix. And as you can see we do not need to get rid of q because, q is available we can use the measured q to compute $[M_{\mathcal{X}}(q)]$, $[J(q)]$, $C_{\mathcal{X}}(q, \dot{q})$, $G_{\mathcal{X}}(q)$ and so on ok. So, we will look at this $\mathcal{F}_r$ a little later, but we can think of that at this place we can also add some external force which is I am denoting by $\mathcal{F}_r$ now.

(Refer Slide Time: 32:26)



So, no inverse kinematics used, direct kinematics used to estimate $\mathcal{X}$ and $\dot{\mathcal{X}}$. If you have vision or other sensor we do not even need to use direct kinematics, we can directly measure $\mathcal{X}$ and $\dot{\mathcal{X}}$. So, this scheme was proposed by Khatib and basically he said that you can use this scheme to do real time obstacle avoidance ok.

So, this was his idea. So, what he said is that we can add the synthetic force $\mathcal{F}_r$, at this place, this is a synthetic force it is a computed force and it looks like it is of this form.

So, this force is the summation of all these forces from N obstacles in the robot field or view and each of these forces is K_i/r_i^n, where r_i is the distance from the i^{th} obstacle.

So, it is like an inversely proportional to the distance from the obstacle. So, we generate a force which is inversely proportional to the distance of the obstacle, not directly inversely some to the power r_i^n.

So, if you now choose K_i and n properly so what do we have?.

We have a force which is as seen by the robot which is generated due to the obstacle and we make sure that this force $\mathcal{F}_r$ is repulsive and we choose K_i and n cleverly such that this repulsive force drops off quickly. So, if the robot is close to an obstacle it will see a large force, if it is far away from an obstacle it does not see anything.

So, this N should be more than 2 or 3 and K_i should be some appropriate number. So, what is happening? What is happening is this control force drives the robot along a desired trajectory ok. When near an obstacle this $\mathcal{F}_r$ is more dominant and it repels the robot away from the obstacle.

So, let us go back to this figure once more. So, at this place we have $\mathcal{F}$, $\mathcal{F}'[M_{\mathcal{X}}(q)]$ plus this $C_{\mathcal{X}}(q, \dot{q})$ and so on, at this place we have $\mathcal{F} + \mathcal{F}_r$. So, $\mathcal{F}$ is driving the robot to follow this desired trajectory $\mathcal{X}_d$, $\dot{\mathcal{X}}_d$ and $\ddot{\mathcal{X}}_d$, but when it comes close to an obstacle, $\mathcal{F}_r$ takes over. So, at this place it is $\mathcal{F}_r + \mathcal{F}$, but $\mathcal{F}_r$ is much long larger. So, it is a repulsive force ok.

So, it repels the robot away from the obstacle. So, this is what he meant as real time obstacle avoidance. So, as the controller is working, as the controller is making sure that the robot is following a desired trajectory, it will also sense the repulsive forces from the obstacle and act accordingly ok. So, this is very very popular and vast popular or even now popular for mobile robots.

So, if you have a mobile robot which is moving around and if it knows that there are these obstacles, it can find the path in some sense based on the control scheme which will avoid all these obstacles ok. So, with this I am going to stop here.

So, in this lecture we have looked at Cartesian control of a robot ok.

So, we are going to design a controller using the position and orientation of the end effector of the robot and also as I showed you we can make sure that it avoids the obstacles ok. In the next lecture, we will look at force control of manipulators and hybrid position force control of manipulators.