

NPTEL Online Certification Courses
Industrial Robotics: Theories for Implementation
Dr Arun Dayal Udai
Department of Mechanical Engineering
Indian Institute of Technology (ISM) Dhanbad
Week: 08
Lecture 34

Gravity Compensation and External Forces/Torques

Welcome back to the module Robot Statics. So, in the last class, you saw the link, link with forces and moments, and we could deduce the torque at the joint through its own weight. Today, we will see what gravity compensation is. Why is it required? What is its application? We will also see the external forces and torques and how they can be taken care of in the control algorithm while designing the robot and while applying a robot for a particular task. So, let us begin with today's lecture.

Introduction to Gravity Compensation



Minimum amount of joint torque/force that the joints actuator should provide even when the robot is stationary.

Applications of Gravity Compensation:

Lead-through programming of COBOT - [Demo video](#)

Controller implementations with floating robot model (uniform behavior in all directions)

Pre-requisite:

Identified model of the robot for its accurate CG locations with respect to the link, and its mass.

Obtained through precise CAD models and/or through identifications experiments with isolated link or after assembling the link to the robot.

So yes, what is gravity compensation? It is actually the minimum amount of joint torque and force that the joint actuators should provide, even when the robot is stationary. What does it mean, actually? Let us say your robot is stationary at a point even to remain there without even trying to move or do any task, So it needs some minimum amount of torque at its joint. So, to stay there due to gravity, there is some amount of torque that comes due to the link masses and the links are connected in series. Why do they need to be analysed so carefully? Because it is a three-dimensional structure. So, each and every link is now oriented in space anyway, like it is not necessarily in a plane, so it is not so trivial to calculate the joint torques. So, we will do using our vector matrix approach to do that. So, that is what we will be doing today in gravity compensation. So, application, what we see, modern robotics-you see, there is cobalt, there are

collaborative robots that work in close collaboration with humans, and they sometimes do not even have joint brakes to hold the robot in case it fails, and most of the time they work in impedance control of things in which it is not under a position control system. Where you have a brake, you have a single input, single output system. When you have controllers that run the joints only in position control mode, rather it is controlling in terms of joint torques. So, that is how it runs. So, it needs close attention to the torque that is arising due to the gravity part of it. So, lead through programming that is a cobalt.

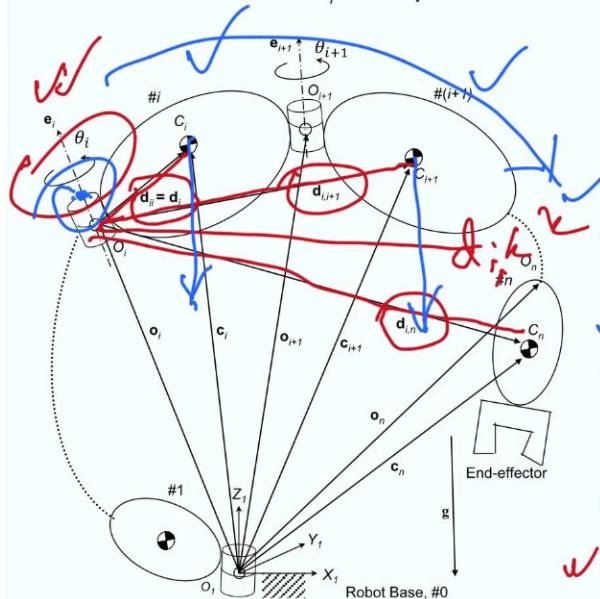


So, I will show you a quick video that I have recorded earlier in my lab. So, you see what I am trying to do. It is a freely floating robot that floats in the air because it is gravity-compensated, so I don't feel the masses of the link. When I am able to move and manipulate the robot using my hand, I am taking it to a different location on the path which I am trying to follow. So, this is led through programming. When I can drag the robot to the desired location, I teach those points, and the robot will, in turn, record those and repeat them. So, this manual teaching process is possible because the robot is totally gravity-compensated. The robot controller doesn't feel as if there is any mass of the links, It is as if it is in space. If you take this robot in space and there is no gravity, so there is no influence of gravity on the links and finally, it sees no torque at all. So, if you move the robot, it only requires the torque, which is required to move the robot dynamically. So, all static torques are absent. So, this is how one can teach a modern robot, and you see now it will repeat the sequence, which is torque. So, repeating is something else but teaching. While teaching, I could drag the robot. That is what is an application of gravity compensation. This is a type of collaborative robot. That is KUKA iiwa, you call it. It has a 5 kg payload capacity. Nowadays even greater payload capacity, like 20-25 kg payload capacity. Cobots are also there which can also be dragged similarly.

So, this is one of the applications, and controller implementation with a floating robot model is easier. You see, a robot should behave similarly, irrespective of the direction. So, if there is no gravity from one side of it, all the directions become similar for the robot controller. So,

designing a controller is much easier. If you make your robot gravity compensated, That means your idle torque of the robot is now is Just taking care enough. It provides enough torque, that is, to take care of the torque that is coming due to gravity. So, finally, it is making your system freely floating in the air So uniform behaviour in all the directions can be achieved. So, what are the prerequisites for this type of control? If you have identified your robot? Why do you need identification? Because robot link, although it appears like a solid body from the outside, it is very complex. Inside, there is are transmission system, inside there is the motor, there is a reducer, there are electronics, there are wires which are hanging all around. So, inside, it is very, very complex. So, even with almost complete information available through the CAD model of it, it still requires to be identified because inside it is very complex, which cannot be estimated before assembling this link. So, once it is assembled, it needs to be identified. You do some kind of experiments to find out the centre of gravity location and the total mass of that body. That is the link. So, accurate CG location with respect to the link is to be known, and its mass is to be found out by experiments. So, finally, that is the prerequisite before we proceed further with gravity compensation or the analysis of gravity compensation. So, that is obtained through precise CAD models and identification experiments, as I have said, with isolated links and or after assembling the link to the robot. So, even after assembling it into the robot, there are experiments to find it out because these robots normally have torque sensors inbuilt into the robot joints. So, that is what it helps: to find out this kind of experiment.

Static Joint Forces/Torques due to Gravity



The forward kinematic transformation matrix for the i^{th} frame with respect to robot's base is:

$$[T_i]_1 \equiv \begin{bmatrix} [Q_i]_1 & \mathbf{o}_{i+1} \\ \mathbf{0} & 1 \end{bmatrix}, \quad \text{where } i = 1, 2, \dots, n$$

The position vector of the center of gravity (CG) $\mathbf{c}_i \equiv [c_{ix} \ c_{iy} \ c_{iz}]^T$ with respect to the base frame O_1

$$\mathbf{c}_i = \mathbf{o}_i + [Q_i]_1 \mathbf{d}_i$$

where $\mathbf{d}_i \equiv [d_{ix} \ d_{iy} \ d_{iz}]^T$ is position of the CG of the i^{th} link with respect to the link $i - 1$

The intermediate vectors \mathbf{d}_{ik} that joins the origin of the i^{th} link to the CG of the k^{th} link are:

$$\mathbf{d}_{ik} = \mathbf{c}_k - \mathbf{o}_i \quad \text{where, } k = [i, i + 1, \dots, n]$$

So let us just quickly get into gravity compensation. So, static joint forces and torques are due to gravity. Before we begin, we will just look at the forward kinematic transformation matrix and what it says. So, a forward kinematic transformation matrix is formed like this. So, you see, you have T_i in frame 1. So, the forward kinematic transformation matrix is always with respect to the ground frame, which is attached to the base of the robot, that is, frame 1, which is here. So, this

is the transformation matrix. What does it contain? Basically, let us say you are talking about this i th link. So, Q_i will denote the orientation of the frame, which is attached here. So, it is basically A_1, A_2 . That is the link transformation matrices. So, it is a product of all of them: A_2, A_3 and so on, and so forth till A_n . So, it gives you the orientation of the link frame, which is attached here. So, that is what Q_i is, that is a 3×3 matrix and also the position of this. So, link orientation is given by the matrix, which is here and the final position it gives it is this one. So, O_{i+1} . It is the tip of the link. That is where your link ends. So, this is the frame which is attached at the end of your link. If you can recall the DH parameters, how we placed all the link frames. So, you remember, this is the frame which we place at the end of the link. That is that gives you the position of O_{i+1} . That is the origin. O_{i+1} . So, that is the point about which link $i+1$ rotates or translates, depending on what type of joint it is. So, that is your O_{i+1} . So, this is what is your transformation matrix.

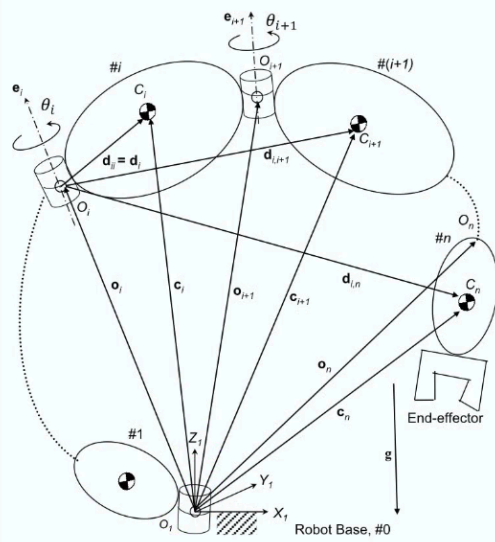
$$[T_i]_1 \equiv \begin{bmatrix} [Q_i]_1 & o_{i+1} \\ \mathbf{0} & 1 \end{bmatrix}, \quad \text{where } i = 1, 2, \dots, n$$

Moving ahead, the position vector of the centre of gravity. That is this location. That is equal to c_{ix}, c_{iy}, c_{iz} , i represent the link number, so it is transposed because I have written it in a horizontal fashion, row-wise, so it is transported, so it is effectively c_{ix}, c_{iy}, c_{iz} , so in column format with respect to the base frame O_1 , which is here. So, it may be given as C_i is equal to O_i , O_i is this plus d_i . d_i is the location of the centre of gravity with respect to its own frame.

$$c_i = o_i + [Q_i]_1 d_i$$

So, normally, this is known if you have a fixed-shape link. So, normally, the centre of gravity is well known with respect to the frame, any of the frame which is attached to it. So, you can calculate the centre of gravity location with respect to the joint also. That is quite easy. So, that is what is d_i , so, but this is in frame O_i . So, in order to know and create this vector, R triangle. So, all the triangle vectors should be in the same frame. So, I prefer to put it in frame 1. So, I have multiplied d_i to Q_i . that is in frame 1. So, Q_i is nothing but, as I said, it is the link, transformation matrices, the product of all of them till here. So, that is what Q_i is. So, it gives you the orientation of the O_i frame with respect to the one frame. So, d_i is now transferred to frame 1. So, now this d_i is in frame 1, O_i is this one, and C_i is the vector which is here. So, this is this vector. Got it. So, that is giving you the location of the centre of gravity of i^{th} link with the first frame, which is here. So, now any intermediate vectors, d_{ik} , what are they? That is $d_{ii}, d_{i,i+1}, \dots, d_{in}$. So, any i k^{th} link will come here. So, it will be d_{ik} . So, k varies from i to $i+1, i+2$, and finally to n . So, d_i 's are all the vectors that originate from i^{th} joint. So, it is all of them, got it. So, why am I considering this? Because I know. So, whatever are the links that come after the i th joint? So, after this joint, if I am calculating torque at this joint. So, all the links that come after this joint, this one, this one, all the links that will come after this joint will have its influence on the torque, final torque or the moment that is received here. Got it? That is the reason I am calculating all the vector distances so that I can take the product cross-product with the gravity forces for all the links. Got it? So that is what I am going to do. So, this is the reason I am calculating all these vectors. Got it moving ahead.

Static Joint Force/Torque due to Gravity



Moment due to gravity at the i^{th} joint \mathbf{n}_i will be the result of moments due to all the link after O_i :

$$\mathbf{n}_i = \sum_{k=i}^n m_k \mathbf{d}_{ik} \times \mathbf{g}$$

m_i : mass of the i^{th} link, \mathbf{g} : acceleration due to gravity

Net torque due to gravity at any joint i is the moment \mathbf{n}_i projected along the joint axis Z_i , i.e., along \mathbf{e}_i .

$$\tau_i = \left[\sum_{k=i}^n m_k \mathbf{d}_{ik} \times \mathbf{g} \right] \cdot \mathbf{e}_i$$

Note: $\mathbf{e}_1 \equiv [0 \ 0 \ 1]^T$ of the first frame lies at the fixed base along the Z -Axis of joint 1.

The net gravity compensation torque vector is:

$$\boldsymbol{\tau}_g = [\tau_1 \ \tau_2 \ \dots \ \tau_n]^T$$



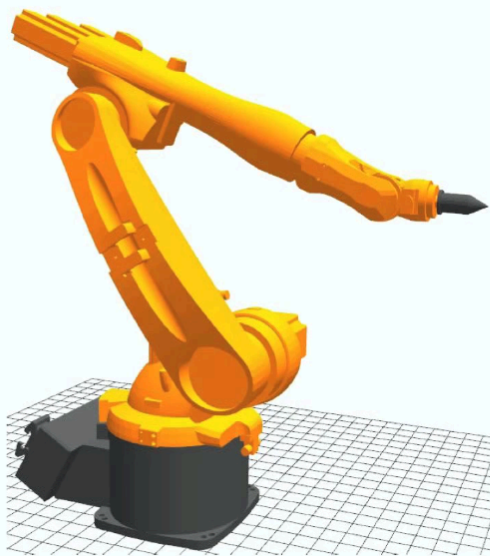
So, a moment due to gravity with a joint, i^{th} joint, will be the result of moments due to all the links that come after O_i , all the links after O_i . So, now I will be using d_{ii} , d_{ii+1} , d_i , and so it is. It is some products of all the \mathbf{d}_{ik} cross $m_i \mathbf{g}$, the sum of all those.

$$\mathbf{d}_{ik} \times m_i \mathbf{g}, \quad k = i, i+1, \dots, n$$

So, your k varies from i to i plus one up to n . So, you got it. So, it is taking the sum of all those. So, that is what this is. So, \mathbf{n}_i is nothing but the moment which is there. So, \mathbf{n}_i is the moment which is realised at this joint got it. So, this is how it is calculated. So, I now know the total moment that is coming over here. So, net torque due to gravity at joint i . If you know the moment, you have to project this moment along the axis which is here. So, you know already that this is nothing but your Z_i axis, and that can be extracted out of the forward kinematic transformation matrix that is still here. So, it is a product of all the matrices till this location. So, that is what will quickly give you the orientation Q_i as well as the vector O_i . So, both of these are combined, and that is there in your transformation matrix, which is here. And you know very well that this gives you the position. These three cross-3 give you the orientation. This is 0, 0, 0, 1. So, the last column of this orientation matrix, three cross 1, is your \mathbf{e}_i vector. So, that is this vector and that is what is aligned along Z_i . So, if you take a dot product, that is, if you want to project your moment along that direction, you have to take a dot product with that unit vector. So, \mathbf{e}_i is a unit vector along the Z_i axis. So, taking the dot product gives me the torque at joint i ; that is what is required. So, you now know, i^{th} joint it is this torque which is going to come. So, all the links are connected with joints. All the joints will have their own torque, and they will have a similar summation of all the cross-products which are there and you have to take the dot product to align it along the joint axis to see all the torques. So, the first link, you know it, is always aligned like this: it is 0, 0, 1. It is aligned with the first perpendicular to the surface plane, that is, \mathbf{e}_1 . So, the first frame lies at the fixed base and Z_1 is along this. So, it is given by 0, 0, 1.

Others can be calculated using transformation matrices, got it? So, this is your torque, so. So, finally, the net gravity compensation torque vector. It is aligned, it is all the torques are here. So, you can just put it in array fashion or column vector format. So, it is τ_1 , τ_2 , τ_3 , τ_4 , τ_5 and till τ_6 if it is the 6 degrees of freedom robot. So, this is what is the set of torque that is to be fed to the robot all the time to maintain its position even if it is not moving. So, it has to balance against gravity and no brakes are applied, got it? So, this makes your robot freely floating. Now you will feel as if there is no gravity which is acting so all the time. But this is dynamic, you know, because if you change the orientation or position of your robot, so joint angle changes, which finally changes all these vectors, so dynamic changes do happen to these, and finally, your torque will also change. So, every position will have a different set of torques that is to be fed to all the joints. So, that is to be updated every time. So, you need to continuously take the feedback of joint angles, update your matrices and vectors and update the new joint torques, which are to be fed to the robot. So, that is how it is made dynamically stable. So, any position, if you take it, it is stable. It will not move due to gravity. Got it so? That is what gravity compensation is.

Resolving External Forces/Torques to Joint Forces/Torques



Video demonstration

- ▶ What are we going to do?
- ▶ **Applications:**
 - ✓ Collaborative manipulation (Robot-Robot), ✓
 - ✓ Human Robot Interaction. ✓
 - Handling tasks in contact (Continuous Control):
 - Surface finishing tasks, Robot assisted assembly
 - Pick-and-Place (Discrete Static loads)
 - Controller design.



So now let us move to the second topic of today. So, this is resolving external force and torque to joint force and torque. Why this is important, I will tell you again. Your robot has to encounter and do some sort of task. So, there are many tasks which require the end-effector of the robot to be in contact with the environment. So, if it makes contact over here at the tip of your robot, this reaction force that is here will create some torque at each of the joints. So, there is a torque which is realised here over here. So, all the joints will see some sort of torque that comes due to the force that acts at the end-effector. So, this is the minimum amount of torque which is now required to take care of the forces. So, in order to maintain your position, even with that force,

you need this additional amount of torque. So, apart from the gravity torque, you now need this additional torque at the joint to face that force or to generate that kind of force. So, that is what is the task. So, what are the applications? Let's say collaborative manipulation. When robot-one of the robots is holding something, another robot is trying to manoeuvre it so that it may be a master-slave kind of robot. If that kind of collaborative manipulation is there, then the robot, which is the master, will move the robot in position control mode, whereas the other one which is holding it. It has to feel the forces. It has to feel the forces that come at its end, and as soon as it feels the force, it has to generate an equivalent amount of torque so that it can maintain the constant force and it can, if required, it can move along with the master robot so that kind of robot-robot collaboration can be achieved. Human-robot interaction: let's say you want to again move your robot based on the external force.



I will show you one small demonstration over here, so you see. What I am trying to do here is I am trying to move this robot using the forces that I am applying from the tip of the robot. That is, I am applying at the end effectors. So, I am applying a kind of push, pull, different sort of direction, force direction. So, there is a force sensor that is feeling the force, and it is converting those forces to the equivalent amount of joint torques by kind of limiting over here; because this is an industrial robot, it doesn't allow you to run the robot in torque mode. So, here I am just limiting the forces which are there at the joints, force and the torque which are there at the joint. See it once again. So, yes, this is what I meant. So, it has to take care of the external force.

So, this is an example of human-robot interaction, handling tasks in contact. It is a continuous control. Let's say you want to do a grinding task-grinding, polishing, buffing-so all those tasks require your robot to maintain a constant force against the surface which it is polishing, grinding, buffing. So, whatever task it is doing, so constant force requirement is there because you want to grind it uniformly all over the surface. So, wherever your robot goes on the surface, you have to maintain the same constant forces against the surface. So, in order to do that, you have to have constantly changing torques at the joint. So, those torques will now produce all the joint torque and will now produce the end effector force which it is trying to maintain. So, this is an

additional torque. So, there are many contributions of torque. The first contribution is due to the gravity torque. You have to compensate against the gravity so that torque is fed to the robot so that it can maintain against the gravity. The second one is to do some kind of job, that is, to take care of the external forces. So, this is the one which is an external force. There are many other torques which are there. One of them may be to move the robot in space with a certain velocity and acceleration. So, those are the different kinds of torque which contribute to the total torque which our actuator generates to do a whole amount of tasks which a robot normally does. So, handling tasks in contact-that is a continuous control- is done. So, surface finishing task, robot-assisted assembly is again a continuous task that a robot can do. So, all the time it may be driven by some sort of external constraints also. You may be required to generate some force in one direction and some other force in another direction, so that kind of controller is also there. So, that also requires external forces and torque. That has to be resolved to the joint torques. So, pick and place, that is a discrete static load that comes on the robot and your robot has to take care of that. So, design any sort of controller for all these applications. So, it requires explicitly stating your joint forces and torques for any external forces and torque that come at this location. That is the tip of your robot's end effector. So, this is a set of applications. This is not an exhaustive list. There are many others also. So, let us quickly analyse these kinds of forces and torques.

Resolving External Forces/Torques to Joint Forces/Torques

Using principle of Virtual Work:

$$\mathbf{w}_e^T \delta \mathbf{x} = \boldsymbol{\tau}^T \delta \boldsymbol{\theta}$$


$\mathbf{w}_e^T = [\mathbf{n}_e^T, \mathbf{f}_e^T]^T \rightarrow$ vector of moments and forces (*wrench*).
 $\boldsymbol{\tau} \rightarrow$ Joint torques.
 $\delta \mathbf{x} \rightarrow$ End-effector angular and linear displacements.
 $\delta \boldsymbol{\theta} \rightarrow$ Joint displacement.

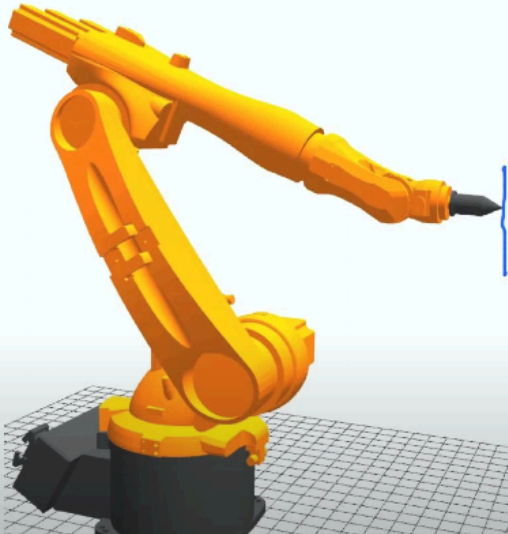
As $\delta \mathbf{x} = \mathbf{J} \delta \boldsymbol{\theta}$,


$$\mathbf{w}_e^T \mathbf{J} \delta \boldsymbol{\theta} = \boldsymbol{\tau}^T \delta \boldsymbol{\theta}$$

$$\Rightarrow \mathbf{w}_e^T \mathbf{J} = \boldsymbol{\tau}^T$$

Or, $\boldsymbol{\tau} = \mathbf{J}^T \mathbf{w}_e$







So yes, using the principle of virtual work is quite old stuff in engineering mechanics that you probably have done quite a number of times. Now, we are applying the same thing to the robot also, so let me just introduce you to the terms which are here. So, if this tip of the robot moves by a small linear and angular displacement given by delta x, which is zero. So, if this moves by that small amount of delta x at the tip, so, this must be doing some kind of work. So, this delta x into the wrench, what is this wrench? A wrench is a vector that contains moment and force. So, that is

your wrench. Wrench is given by three cross one moment, three cross one force, so it is a total six cross one vector. So, that is your wrench vector. So, the wrench is multiplied by the angular displacement and linear displacement.

$$\mathbf{w}_e^T \delta \mathbf{x} = \boldsymbol{\tau}^T \delta \boldsymbol{\theta}$$

So, effectively, the moment is multiplied with the angular displacement and forces are multiplied with the linear displacement. If you do this, so the input is this one. So, that is what goes at the end effector level. So, that is the output work which is done by the robot, so it is wrenched into the displacement vector. So, that is there. The second part is the input work done by the joints. So, it is nothing. But this is your array of torque which is there at each joint. So, that is tau one, tau two, tau three, tau four, tau five and tau six. So, it is all the six joints torques which is here for a six degree of freedom robot. Definitely, it will be different for the seven degrees of freedom robot. So, torque transpose, because it has to be oriented like this. So, you have tau one, you have tau two, you have tau like this arranged over here, and you have joint angles which are here, so torque into joint angles which each of the joints is moving in order to give you the displacement delta x. So, that is your input work done. So, input work done is equal to output work done. So, it is a small incremental displacement, which is written here if you remember your virtual work principle. So, it is to be taken like this. So, I have just balanced both of them. So, now this is your final equation which says: your output work done- and this is your input work done- torque into displacement, angular displacement of the joint. It can even be a linear displacement if it is a prismatic joint, whereas this one is your output work done. So, wrench into the angular and linear displacement array. That is over here. That is delta n, what it. So, now I am replacing delta x with Jacobian of delta theta. So, if you remember, your Jacobian relates the joint rates to the end effector rates using this relation. It is a small differential that is here. So, I am quickly substituting this to this equation. So, what I get here is wrench transpose instead of delta x. i will put j delta theta over here, and the right side remains as it is. So, you can finally write it like this.

$$\mathbf{w}_e^T \mathbf{J} \delta \boldsymbol{\theta} = \boldsymbol{\tau}^T \delta \boldsymbol{\theta}$$

$$\mathbf{w}_e^T \mathbf{J} = \boldsymbol{\tau}^T$$

So, wrench transpose Jacobian is equal to torque transpose. This is torque at the joints transpose, taking transpose both of both sides. I can write it like this, so it becomes: torque is equal to the Jacobian transpose of external forces.

$$\boldsymbol{\tau} = \mathbf{J}^T \mathbf{w}_e$$

So that is the wrench that comes over here. So, whatever is the external force and torque that you want to maintain over here, in order to maintain that, you have to provide this much torque at the joints, so it can be otherwise also if you are providing this much torque. So, you will see this amount of force at the tip of your end effector. So, this is how it is related, and this is very, very important in order to design any controller, for any impedance controller, stiffness controller, admittance controller or any kind of controller to do any specific kind of task.

So that's all for today. I think that should be enough for robot statics. There are many other parameters also that can be studied, but we'll limit them over here because in an industrial robot,

you don't have much freedom to handle much of things. So, this should be quite good enough to understand the kind of application normally which industrial are made for. So, we'll end it here. So, in the next class, I'll do Kinetostatic Measures for Robot Design, in which we'll be doing velocity and force ellipsoids. So, that's all. Thanks a lot.