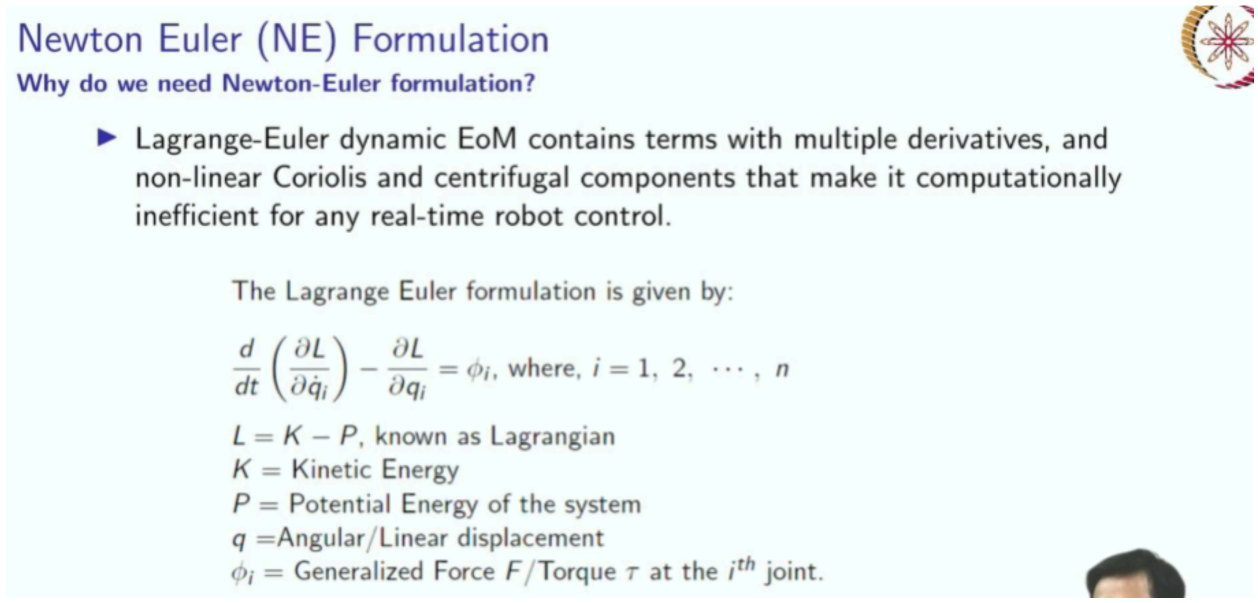
# NPTEL Online Certification Courses
# Industrial Robotics: Theories for Implementation

**Dr Arun Dayal Udai**
**Department of Mechanical Engineering**
**Indian Institute of Technology (ISM) Dhanbad**

**Week: 09**
**Lecture 38**

## Newton-Euler (NE) Approach

Hello everyone, welcome back to the module Robot Dynamics. So, in the previous class, we learnt how to derive dynamic equations of motion using Lagrange Euler formulation. We understood the physical significance of equation of motion terms in context to inertial forces and torques, centripetal and Coriolis forces and torques, and gravitational component of the equation of motion. We learnt the limitations of industrial robots and why it does not allow many of the equations of dynamics to be implemented while we actually program them. So, however, we will continue today with one more formulation, that is, Newton Euler (NE) formulation, to obtain the joint torque and forces given the motions of the links and the joints. We will try to get the equation of motion, the one which we obtained, just like the Lagrange-Euler formulation. So, we will try to get it. So, let us start.



### Newton Euler (NE) Formulation

**Why do we need Newton-Euler formulation?**

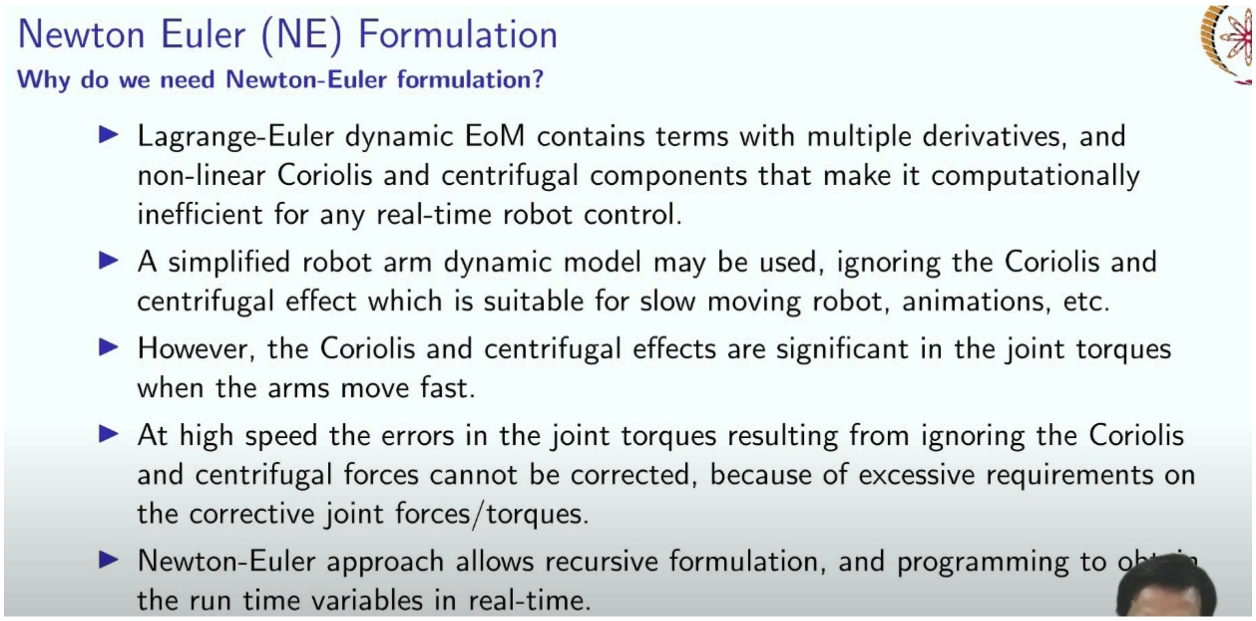
- Lagrange-Euler dynamic EoM contains terms with multiple derivatives, and non-linear Coriolis and centrifugal components that make it computationally inefficient for any real-time robot control.

The Lagrange Euler formulation is given by:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = \phi_i, \text{ where, } i = 1,\ 2,\ \cdots,\ n$$

$L = K - P$, known as Lagrangian
$K$ = Kinetic Energy
$P$ = Potential Energy of the system
$q$ =Angular/Linear displacement
$\phi_i$ = Generalized Force $F$/Torque $\tau$ at the $i^{th}$ joint.

So, before we begin, let me just see why we need Newton Euler (NE) Formulation. What was the problem with our earlier formulation? So, Lagrange Euler's formulation, equation of motion contains the terms with multiple derivatives. You saw it used partial derivatives with respect to velocity, joint velocity, as well as with respect to time. We also used partial derivatives with respect to joint angles. So, you know now it includes quite a lot of derivatives, and there are terms called non-linear terms, which are Coriolis terms, centrifugal components that make it computationally inefficient for any real-time robot control because you need to take the derivatives, you know, and taking derivatives digitally using programming, that is not that computationally easy. So, that is what limits a lot of Lagrange Euler formulations to be used directly in any control system or directly using it, apart from doing some sort of offline simulation and all.



## Newton Euler (NE) Formulation

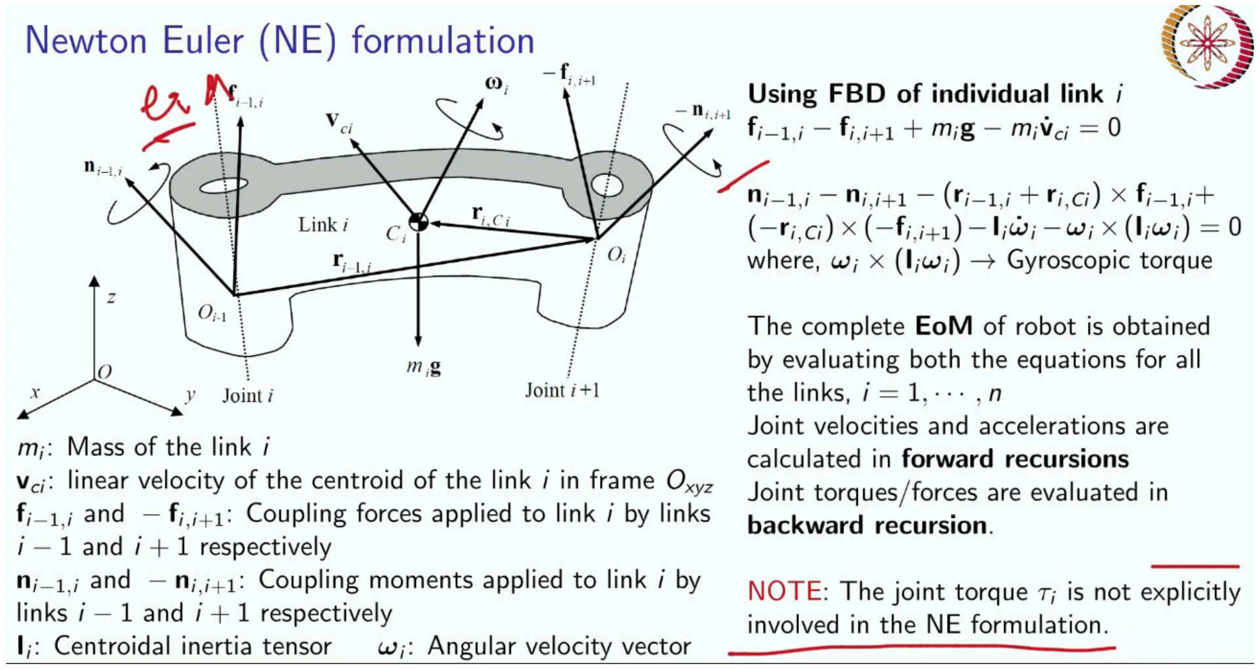
**Why do we need Newton-Euler formulation?**

- Lagrange-Euler dynamic EoM contains terms with multiple derivatives, and non-linear Coriolis and centrifugal components that make it computationally inefficient for any real-time robot control.
- A simplified robot arm dynamic model may be used, ignoring the Coriolis and centrifugal effect which is suitable for slow moving robot, animations, etc.
- However, the Coriolis and centrifugal effects are significant in the joint torques when the arms move fast.
- At high speed the errors in the joint torques resulting from ignoring the Coriolis and centrifugal forces cannot be corrected, because of excessive requirements on the corrective joint forces/torques.
- Newton-Euler approach allows recursive formulation, and programming to obt
the run time variables in real-time.

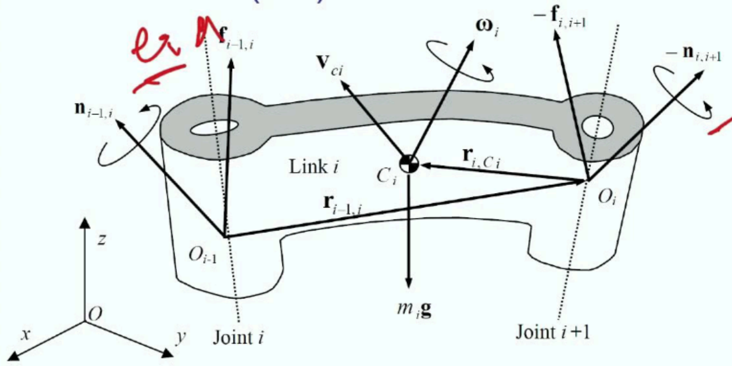
So, a simplified robot arm dynamics model may be used. We saw dynamic torque. Torque is equal to I theta, double dot plus C theta, theta, dot plus gamma theta, you saw. So, you can ignore the term which includes Coriolis and centrifugal component. So that will make it a little simpler. If we assume that the robot is not moving quite fast, we can do that. But yes, that also is possible only for slow-moving robots, for animation and all those stuff which doesn't require much accuracy in calculation.

Coriolis and centrifugal components are significant in joint torques when your arm moves really very fast. Normally, industrial robots run at 2 meters per second. At the tip velocity, it goes like that, so it is very, very fast, you see. So, at high speed the errors in the joint torques resulting from ignoring the Coriolis and centrifugal forces cannot be corrected because of excessive requirements on the corrective joint forces and torques. So, let us say you are using a model-based control. You are using your dynamic equation of motion without Coriolis and centripetal terms so that it is fed to the robot. We are checking the errors. We are further correcting that using our controllers so that correction goes really very high, and that is sometimes not enough to take care of complete, precise control of the end effectors. So, that is what we will talk about these controls and corrections in our control module also.

So, Newton Euler's approach allows recursive formulation. We will see what a recursive formulation is and programming. It also allows easy programming to obtain run-time variables in real-time. There are limitations to this also. We will discuss this immediately after this.



## Newton Euler (NE) formulation



$m_i$: Mass of the link $i$
$\mathbf{v}_{ci}$: linear velocity of the centroid of the link $i$ in frame $O_{xyz}$
$\mathbf{f}_{i-1,i}$ and $-\mathbf{f}_{i,i+1}$: Coupling forces applied to link $i$ by links $i-1$ and $i+1$ respectively
$\mathbf{n}_{i-1,i}$ and $-\mathbf{n}_{i,i+1}$: Coupling moments applied to link $i$ by links $i-1$ and $i+1$ respectively
$\mathbf{I}_i$: Centroidal inertia tensor $\quad$ $\boldsymbol{\omega}_i$: Angular velocity vector

**Using FBD of individual link $i$**

$$\mathbf{f}_{i-1,i} - \mathbf{f}_{i,i+1} + m_i\mathbf{g} - m_i\dot{\mathbf{v}}_{ci} = 0$$

$$\mathbf{n}_{i-1,i} - \mathbf{n}_{i,i+1} - (\mathbf{r}_{i-1,i} + \mathbf{r}_{i,Ci}) \times \mathbf{f}_{i-1,i} + (-\mathbf{r}_{i,Ci}) \times (-\mathbf{f}_{i,i+1}) - \mathbf{I}_i\dot{\boldsymbol{\omega}}_i - \boldsymbol{\omega}_i \times (\mathbf{I}_i\boldsymbol{\omega}_i) = 0$$

where, $\boldsymbol{\omega}_i \times (\mathbf{I}_i\boldsymbol{\omega}_i) \rightarrow$ Gyroscopic torque

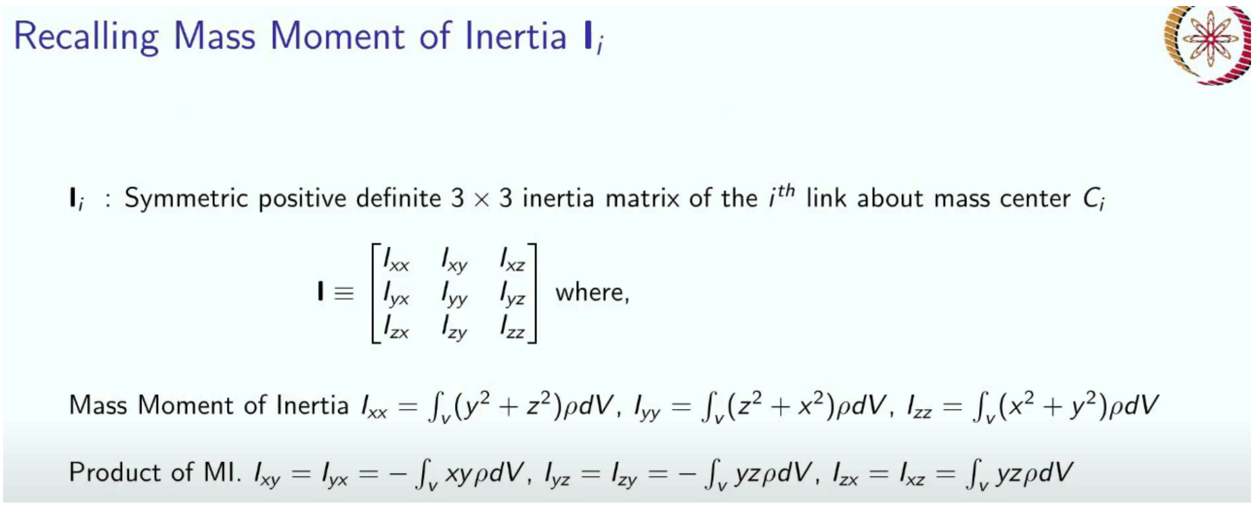
The complete **EoM** of robot is obtained by evaluating both the equations for all the links, $i = 1, \cdots, n$
Joint velocities and accelerations are calculated in **forward recursions**
Joint torques/forces are evaluated in **backward recursion**.

NOTE: The joint torque $\tau_i$ is not explicitly involved in the NE formulation.

So, yes, traditionally, Newton Euler's formulation was derived. It used something like this: a link, you know link-is mounted on a frame which is here. It ends with a frame which goes here. You have seen a similar situation when you were placing frames in your DH parameters. So, you had a frame before to link about which your link is making its motion, and there is a frame which is attached at the end of the link that is actually moving along with the link. So, let us say it is acted upon by forces from the link, which is before this. This is the moment. These are the forces, and similarly, this link applies forces and moments to the link next to it. So, in reaction, this also will see this force and the moment. Let us say it has its own angular velocity, omega i, and it has the centre of mass, which is located here, and it has a velocity given by Vci. mig is the mass which is acting due to the centre of gravity. This force is coming, that is, towards the direction of g. Joint i, joint i plus one. So, there are additional parameters which are put here. So, with respect to Oi, that is the link attached to the frame. So, there are two vectors which are to be noted. One of them is the mass centre location with respect to links, own frame that is given by ri, Ci. So, that is this. And again, you have ri minus one i ($r_{i-1,\,i}$). So, what is that? It is a vector that connects this to this, got it? So, with this, all parameters and symbols that I have used. So, $m_i$ is the mass of the link i, Vci is the linear velocity of the centroid of the link i in frame $O_{xyz}$ that is, with respect to O, and you have forces given by this. Force over here, force over here. and you have moment coupling moments that come on the that is applied on the link by the links i minus one and i plus one, respectively. Those are here, and you have the moment of inertia tensor. You know, a moment of inertia tensor. It looks like this: you have principal moment of inertia that comes here: Ixx, Iyy, Izz, and then you have product moment of inertia: Ixy, Ixz, similarly, Iyx, Iyz, Izx, Izy. So, this is your inertia moment of inertia tensor. It looks like this: these are the product moment of inertia, whereas the terms which come here in the diagonal are the principal moment of inertia. Omega i is an angular velocity vector. That is here. So, using the free-body diagram of the link i, you can write this equation. We wrote it earlier while we were doing statics. We use till here, only this time this additional term will also come. This is the force acting due to the acceleration of the centre of mass. That is this. So, if this is the velocity. Vci dot is the acceleration of the centre of mass, so that is also added here. Apart from this gravitational force, this is the gravitational force, which causes the force due to the acceleration, and these are the forces that are acting on the body. So, these are the two forces. So, the vector sum of all of them should be equal to zero because, assume, if it is taken care of by accelerating forces over here, the sum should be equal to zero, or this is the set of forces which is causing this acceleration.

$$f_{i-1,i} - f_{i,i+1} + m_i g - m_i v_{ci} = 0$$

Either way, it is true. Similarly, when we consider moments from both directions, both sides, one is this, another one is this, so those are the pure moments that are coming on the link, and you also see moment. That is because of the forces acting on this frame over here and over here. So, you know the distance is already, so quickly you can call it fi minus 1 i, which is acting over here, and you have distance which is given by two vector sum: this and this. That is, effectively I am trying to find out this vector. So, this vector in this force, that is the moment taken about Ci, and you also see a moment due to this part of it, and that is given by minus rci. That is this vector and this force that is here, and you have angular acceleration, which is here, and moment of inertia is here. This is the inertial moment that will come. So, it is given by i omega dot, and you also have a gyroscopic moment, which is coming here. So, that goes here. So, that is given by omega i cross i omega i, so that is another term. So, you have what are moments that you see it is the pure moments that are acting on the link and moments due to the forces, moments due to the inertia of the link itself, inertial moments that are due to the omega dot, and you have a gyroscopic moment. So, that comes here. So, the complete equation of motion is obtained by evaluating both the equation and this equation, Newton and Euler equations and combining them for all the links, and for i equal to one to n. So, for all the links, you write these two vector equations, and you can do them in a forward and backward manner. So, what you do in the forward recursion you calculate velocities and acceleration. In forward recursion that means, if you are seeing, this serial chain looks like this: you have links which are connected like this. So, any velocity over here is going to affect the tip velocity. Velocity over here will also affect this. So, this is actually affected by any of the velocities which come prior to this. So, you have to move forward. You have to move forward from here to here. So, all the velocity contributions due to the links which come prior to this are added on, and it goes here. So, we do velocity and acceleration at any joint or at the link mass centre. So, that is to be considered by forward recursion. That is, all the velocities which come before this are to be considered, and they are contributing to the velocity that comes here. And similarly the joint forces are evaluated in backward recursion. So, we have similarly done this when we were doing statics. So, again, if your robot looks like this, if you want to calculate joint forces over here, all the links that come after this have a contribution over here, but any link which comes before this does not have any contribution to the torque that will come here. So, that is the reason it is calculated in a backward manner. So, you know your constraints, which are acting at the tip. Then you know this, you know this, and you move like this. So, that is the reason you are moving backwards. So, using these two pairs of vector equations for all the links, you calculate the joint velocities and acceleration, as we have discussed in an earlier module, also while discussing differential motions. So, use that, use these equations, and you can do backward recursion to calculate the moments. So, moments are projected further along the joints to obtain the torques at the joints. So, joint torque is not explicitly involved in Newton Euler's formulation. So, that was there in Lagrange Euler's formulation. You see each joint, and you directly obtain the joint torques. Here, you don't obtain the joint torque; you obtain the moment, and you have to project that along the axis that is ei, ei minus 1, ei minus 1, like that. You have to project onto this axis to obtain the joint torques. So, this is a traditional approach of Newton Euler. Again, you see, you have to write a huge number of equations, you have to substitute a lot, and you have to do. You have to obtain the joint torques. Let me just move a bit so that you can see the slide clearly. This is what was there.



## Recalling Mass Moment of Inertia $\mathbf{I}_i$

$\mathbf{I}_i$ : Symmetric positive definite $3 \times 3$ inertia matrix of the $i^{th}$ link about mass center $C_i$

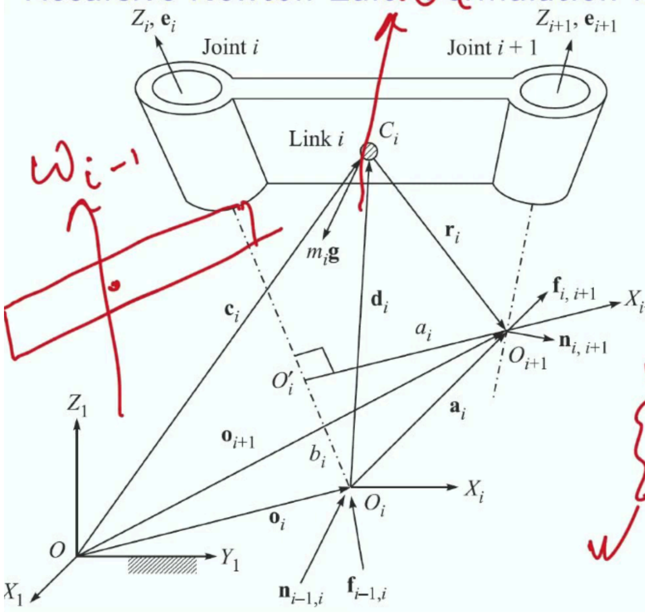
$$\mathbf{I} \equiv \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix} \text{ where,}$$

Mass Moment of Inertia $I_{xx} = \int_v (y^2 + z^2)\rho dV$, $I_{yy} = \int_v (z^2 + x^2)\rho dV$, $I_{zz} = \int_v (x^2 + y^2)\rho dV$

Product of MI. $I_{xy} = I_{yx} = -\int_v xy\rho dV$, $I_{yz} = I_{zy} = -\int_v yz\rho dV$, $I_{zx} = I_{xz} = \int_v yz\rho dV$

So, now let us discuss the standard moment of inertia. What does it look like? As I have told you, so it should look like this. This is simple: a recapitulation. You just should remember how mass moment of inertia is calculated: along xx, along yy, along zz. So, these are the diagonal terms. You know that is directly giving you the principle. Moment of inertia and product moment of inertia is defined like this. So, yes, this is how you calculate the link moment of inertia. This is to be calculated about the centre of mass of the link. So, anything. If it is in any other frame, you have to bring it to this frame. So, bringing it to any frame, you know how to do it using the orientation matrix of that particular link. So, we will do it. We will do it and demonstrate it.

## Recursive Newton Euler Formulation for Dynamics



**Forward Computations**

**Step 1**: *Angular velocity propagation*

Angular velocity of $i^{th}$ link is

$$\boldsymbol{\omega}_i = \begin{cases} \boldsymbol{\omega}_{i-1} + \dot{\theta}_i \mathbf{e}_i & \text{for revolute joint} \\ \boldsymbol{\omega}_{i-1} & \text{for prismatic joint} \end{cases}$$

Expressing in $i^{th}$ link frame

$$[\boldsymbol{\omega}_i]_i = \begin{cases} \mathbf{Q}_{i-1}^T[\boldsymbol{\omega}_{i-1}]_{i-1} + \dot{\theta}_i[\mathbf{e}_i]_i & \text{for revolute} \\ \mathbf{Q}_{i-1}^T[\boldsymbol{\omega}_{i-1}]_{i-1} & \text{for prismatic} \end{cases}$$

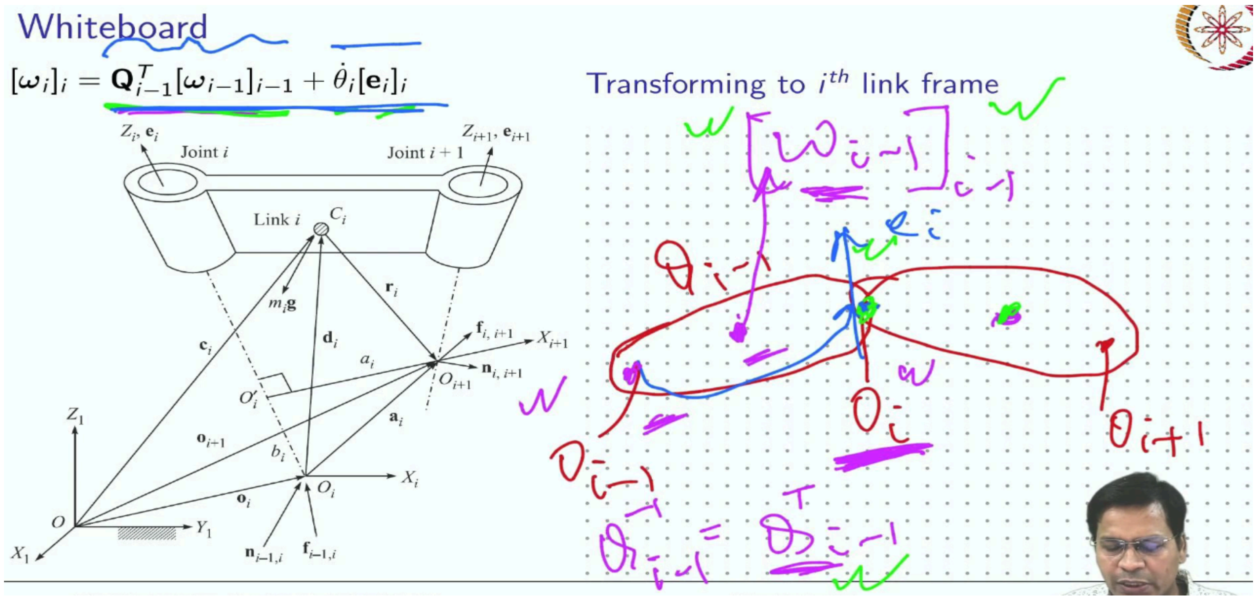
where $\mathbf{Q}_{i-1}^T$ is $3 \times 3$ rotation matrix representing orientation of frame $i$ attached to link $i-1$ with respect to the frame $i+1$ attached to link $i$: ✓

$$\mathbf{Q}_{i-1}^T = \begin{bmatrix} c\theta_{i-1} & s\theta_{i-1} & 0 \\ -s\theta_{i-1}c\alpha_{i-1} & c\theta_{i-1}c\alpha_{i} & {}_{-1} \\ s\theta_{i-1}s\alpha_{i-1} & -c\theta_{i-1}s\alpha_{i} & {}_{-1} \end{bmatrix}$$

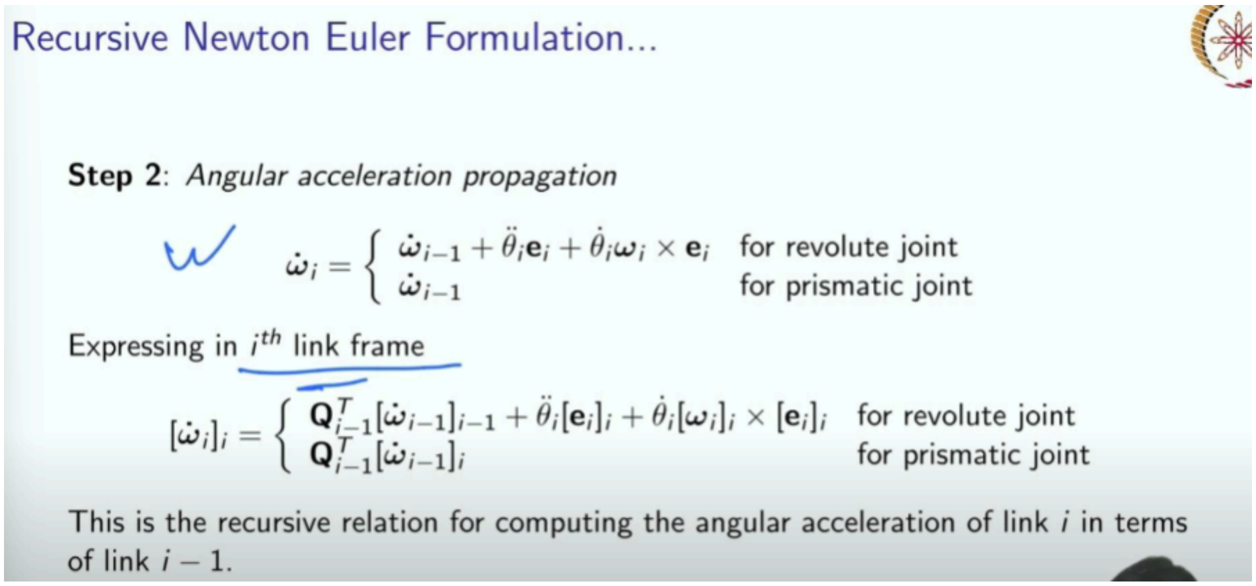
So, now there is yet another extension of Newton Euler's formulation, which is actually allowing you to calculate this programmatically recursion through recursion. If you know joint parameters, link and joint parameters for one of the links which come next, you can calculate this one. So, using this, you can calculate this. Using this, you can calculate the previous one. Using this, you calculate the previous one. So, iteratively, you can come backwards, or, similarly, you can go forward while calculating the velocity. So, that is how, using recursion, it becomes very, very simple to do programming and formulating these algorithms. So, yes, forward computation. You know this equation very well already. We have discussed this in our earlier module in kinematics earlier. In differential motion also, we have discussed this. We have taken the derivative, and we found out the velocity. We also found out the acceleration. Again, I am just coming back to that. So, yes, angular velocity, omega i, for any link is given by omega i minus 1.

$$\square_i = \square_{i-1}$$

So, this is your link, this is your link, that is ith link. So, the angular velocity of this link, omega i, let me just leave it up. The angular velocity of this should be angular velocity, which is imparted by the previous link because this link is mounted on top of the previous link. So, whatever angular velocity comes due to this angular velocity, omega i minus 1. You also have joint angular velocity. You have theta dot that is here and that is along ei. So, if you multiply this scalar theta dot with ei, this effectively gives you omega i. So, that is over here, that is landing over here in its frame. So, total omega i is the net angular velocity due to this contribution, and the joint Angular velocity sum together will give you the angular velocity that this Ci, or the whole of this link, Ci's. So, the whole of the link will have angular velocity due to omega i minus 1. That is the one on which it landed. So, there is a link which is mounted on this. If this previous link rotates, this also rotates with the same angular velocity. Apart from that, there is relative angular velocity that is to be added. So, that is what is there. In the case of the prismatic joint, there is no theta dot, and there is no angular velocity. So, the whole of the second link is mounted on the previous link. Suppose this rotates with angular velocity, omega i minus 1. So, the next link will also have the same angular velocity because there is no relative angular velocity. There is only linear displacement, which is there. Got it? So, that is in the case of prismatic joints. Got it? Now, you know that all the vector sums can be done only if they are represented in the same frame. So, philosophically, it is exactly this. But now, in order to do it actually practically, you should understand these representations are in different frames. We know the values in its frame. We have to convert it to the frame in which we are calculating. So, in order to do that, you now have to do this. So, what is this actually? This is Qi. You know this very well. Qi minus 1 is the rotation matrix subset. That is a rotation matrix subset of the transformation matrix, you know that. It was 3 cross-3 rotation matrix and translations over here. This was 1, this was 0 0 0. So this was your Q, and that can be obtained from the transformation matrix, you know. So, this is your Qi that is the link transformation matrix that takes you from this to this, and this gives you the position of this with respect to this over here and orientation of i plus 1 with respect to i. If it is Ti i minus 1. So, you got it. So, this is what your transformation matrix gave you. So, a part of this in the case of. So, let me just wipe it off. so yes, this is your same matrix. Now, omega i minus 1 is a link which was connected prior to this. This had an angular velocity of omega i minus 1, and you have angular velocity over here. That is what you want to obtain. That is omega i. So, Qi minus 1 is a 3 cross 3 rotation matrices. You just note it down. That represents the orientation of the frame i attached to the link i minus 1. With respect to the frame i plus 1 attached to the link i. So, how does this transformation happen? You basically extracted this from the transformation matrix i minus 1 link transformation matrix. So, you know all these parameters using the DH parameter. So, you got this orientation directly. So, let me just elaborate upon this over here: how this equation came, actually.



So, yes, let me just draw it once again over here. You have a link, which is here. You have yet another link that comes here, and this is your frame O i minus 1($O_{i-1}$). This is your frame Oi and then you have a frame which is here that is Oi plus 1 ($O_{i+1}$). So, this Qi minus 1 basically gives you a transformation that takes you from here to here. Orientation transformation from here to here. Now, you have angular velocity about this link, that is, omega i minus 1, about its frame. You know it is omega i, so this is given by omega i minus 1, so that is here. Now, you have to represent all these 2, the frame which is here about because you want to do everything at this, over here. So, now you have to transform this to this. You know this angular velocity is over here, and it is represented over here. But you know the transformation. This is the fixed frame to this link, and you know the location of this orientation with respect to this. That was given by Qi minus 1. When you take the transpose of this, that is equivalent to taking invert to this. So, the inverse of this is equal to the transpose of this. So, taking the inverse actually represents this with respect to this. That makes this frame stationary, this frame stationary. Now, you are looking at everything in this frame. So, when you multiply Qi minus 1, transpose to omega i, this one. You are multiplying this to this over here. So, what you do, basically you represent this omega in this frame. So, now you know omega i minus 1 in this frame. So, this is now well known, and you already know this link is rotating by theta i dot about an axis in its frame given by ei. So, that again is in this frame, ith frame. So, now both of these can be added together. So, what I actually did here was I multiplied Qi minus 1 and transposed it to omega i minus 1. That effectively transformed omega i minus 1 to ith frame. Again, this is the relative angular velocity. So, the total angular velocity in ith frame is omega i. that is seen by this. That becomes omega i in its frame. So, this is how it works. So, now let us go back to our earlier slide. So, that is what was given by this. In the case of prismatic joint, only this term will be absent. There is no relative angular velocity, so only this part remains. So, now you got what is omega i in its frame.


## Recursive Newton Euler Formulation...

**Step 2**: *Angular acceleration propagation*

$\omega$

$$\dot{\boldsymbol{\omega}}_i = \begin{cases} \dot{\boldsymbol{\omega}}_{i-1} + \ddot{\theta}_i \mathbf{e}_i + \dot{\theta}_i \boldsymbol{\omega}_i \times \mathbf{e}_i & \text{for revolute joint} \\ \dot{\boldsymbol{\omega}}_{i-1} & \text{for prismatic joint} \end{cases}$$

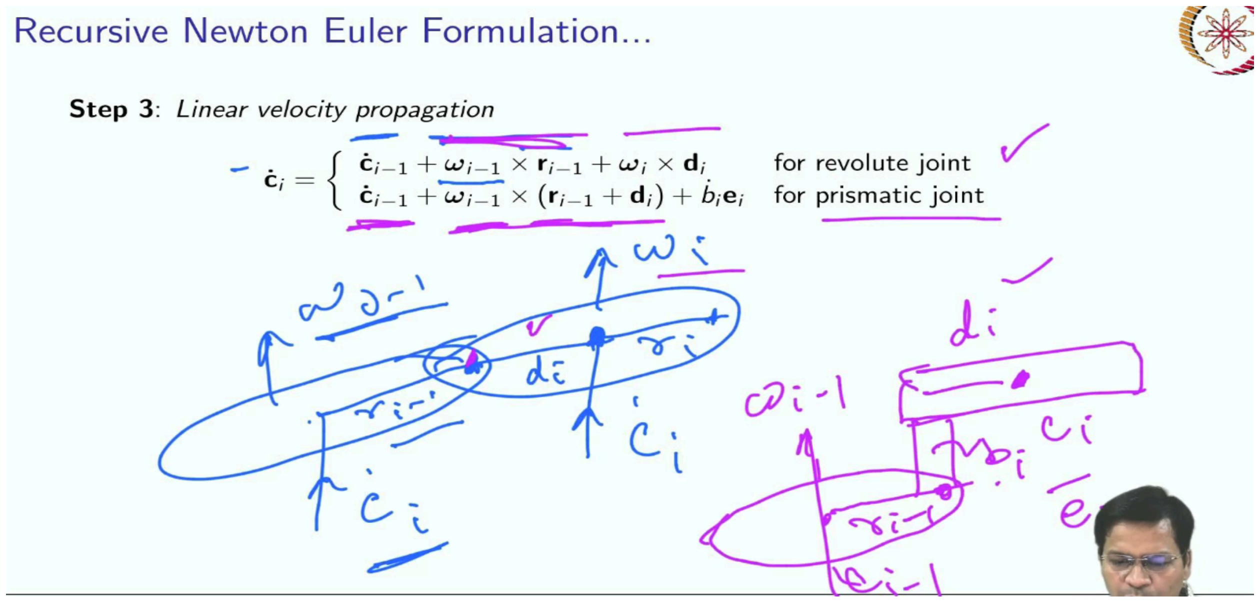
Expressing in $\underline{i^{th} \text{ link frame}}$

$$[\dot{\boldsymbol{\omega}}_i]_i = \begin{cases} \mathbf{Q}_{i-1}^T [\dot{\boldsymbol{\omega}}_{i-1}]_{i-1} + \ddot{\theta}_i [\mathbf{e}_i]_i + \dot{\theta}_i [\boldsymbol{\omega}_i]_i \times [\mathbf{e}_i]_i & \text{for revolute joint} \\ \mathbf{Q}_{i-1}^T [\dot{\boldsymbol{\omega}}_{i-1}]_i & \text{for prismatic joint} \end{cases}$$

This is the recursive relation for computing the angular acceleration of link $i$ in terms of link $i-1$.

Now, let us move ahead. So, this is what you have seen. So, now angular acceleration. How does that propagate? So, that was one. This is a recursive relation. You see if you know omega i minus 1 because you know velocities are done by forward computations, acceleration and velocity, so you have expressed the next one, this one with respect to this, so this comes with later, this comes prior, got it. So, you have to calculate this from this. You have to go from the bottom-most first link to the last link. So, that is how we move.

Now, moving at. in the case of angular acceleration, I did nothing. I simply took the derivative of omega i, which we have obtained earlier, and you can directly get to this. If you want you can just get it to this omega i was equal to omega i minus 1 plus theta i dot into ei. This is a vector, vector, vector. This omega i dot is a scalar. If you take the derivative of this scalar, if you take the derivative of this, you get the omega i dot should be equal to omega i minus 1 dot plus. These two are products. So, you do theta i double dot ei. This is again along the same direction, plus. You have theta i dot. Now, ei is a vector, so it is theta I dot is not changing in magnitude, so it is changing in direction. And how it is changing is given by omega i cross ei. So, these are the two vectors. So, you see, you have to take the cross-product over here, and this is there. So, this is perpendicular to the direction of ei. So, this is along a tangential direction, and this is along the same direction. So, yes, these are the two accelerations, these are the two accelerations that are added to this. So, finally, just taking a derivative of this, you can obtain the acceleration. Got it? And in the case of the prismatic joint, all of these will be absent, got it? So, this is quite trivial now at least. So, again, in with link frame, the way we did it for angular velocity, we can do it with angular acceleration also. Again, we have to bring everything to the ith frame. We again multiplied with the same transformation matrix, now that is the orientation matrix, the inverse of that so that it comes to the with frame. So, this equation in with frame can be written as this. So, this, again, is a recursive relation for computing the angular acceleration of the link I in terms of I minus 1. So, you are going forward again. So, angular acceleration and velocities are done.

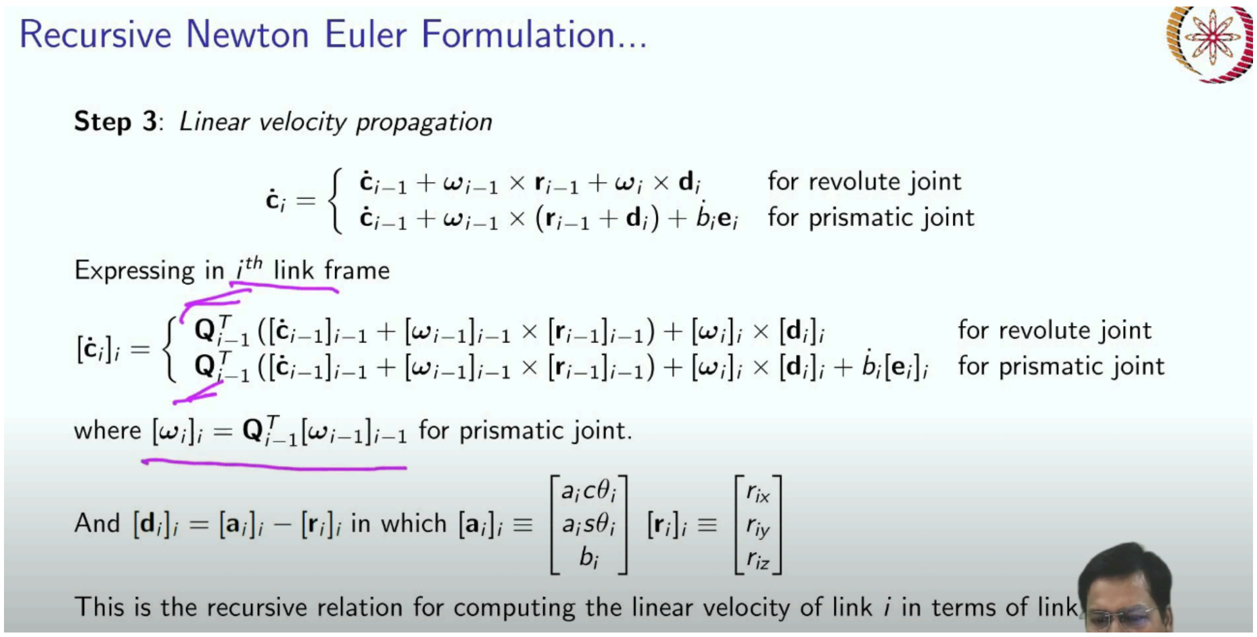


## Recursive Newton Euler Formulation...

**Step 3**: *Linear velocity propagation*

$$\dot{\mathbf{c}}_i = \begin{cases} \dot{\mathbf{c}}_{i-1} + \boldsymbol{\omega}_{i-1} \times \mathbf{r}_{i-1} + \boldsymbol{\omega}_i \times \mathbf{d}_i & \text{for revolute joint} \\ \dot{\mathbf{c}}_{i-1} + \boldsymbol{\omega}_{i-1} \times (\mathbf{r}_{i-1} + \mathbf{d}_i) + \dot{b}_i \mathbf{e}_i & \text{for prismatic joint} \end{cases}$$

Now, the linear velocity and acceleration. So, yes, this time again, this is your Ci, that is, the centre of mass velocity, I want to find out. So, again, this link is riding upon the previous link, so it sees a velocity which is contributed by the. So, this is a link. You have a previous link which is here. So, how will you do it? You have a velocity over here that is given by the Ci location; the Ci dot is its velocity, and you have angular velocity omega i minus 1. So, angular velocity contributes to its total velocity omega i minus 1 has its contribution over here that is Ci location, Ci dot we want to find out. So, over here, there is a contribution due to omega i minus 1. So, that is given by this distance. So, omega i minus 1 cross ri minus 1, and again, omega i itself will l have its contribution over here. That is given by this. Let me just put down all the variables here only. So, this is your ri minus 1. And you have another vector, which is di. All these are in three dimensions, and I am just drawing them in a planar fashion. This is your ri that will take you to the next one, the next joint. So, this one has its contribution here. So, this is your omega i, and this has omega i minus 1. So, yes, if you have to transfer omega i minus 1 to this joint, so you have to multiply with the distance, which is here. So, ri minus 1 cross omega i minus 1 gives you velocity. That comes here. That is the velocity contribution due to omega i minus 1. This is pure velocity. This is velocity due to angular velocity. This is pure velocity, and then you have velocity contribution due to di distance. So, this is your di distance, and this is your omega i. So, that also has got contribution over here. So, overall, the total velocity is seen by this. Now again, in the case of the prismatic joint, things are a little different, so in this case, you have a link, but this time, this is your omega i minus 1. This, again, is your ri minus 1. This is your joint. But this joint, this time, has got a prismatic extension and then comes your link that comes next. It may be rotating again, and it may be another prismatic joint; forget about it. But yes, this is your centre of mass location of link i. This is Ci minus 1, and you have a few things which are here. This is your joint offset, and this is di, let us say. It is along i joint vector, axis vector. So, this is your di. di is the joint offset. That is what is changing in the case of prismatic joint. You know that it is not constant; that is extending, or it can retract. So, bi is a joint variable. There is no theta i here, and then you have di that comes here. So, now you can understand. So, when you want to come to this joint. So, what velocity contribution will come here? It is one due to the pure velocity of the previous link, because this link itself is riding upon the previous link. So, it is given by Ci minus 1 dot and now due to the angular velocity contribution. So, total angular velocity, omega i minus 1 is at a vector distance of ri minus 1 and di from here. So, that is directly added over here, and the cross product of that will give you linear velocity due to omega I minus 1, and pure velocity component due to this, di dot and ei will also give you some velocity. So, that is coming here. So, total velocity is this. So, in the case of rotary and prismatic joints, it comes like this. You have to draw your figures, better figures. Maybe you can draw in three dimensions somewhere in CAD, join the vectors and see what it looks like. Put all the variables clearly, and you can see it.



## Recursive Newton Euler Formulation...

**Step 3**: *Linear velocity propagation*

$$\dot{\mathbf{c}}_i = \begin{cases} \dot{\mathbf{c}}_{i-1} + \boldsymbol{\omega}_{i-1} \times \mathbf{r}_{i-1} + \boldsymbol{\omega}_i \times \mathbf{d}_i & \text{for revolute joint} \\ \dot{\mathbf{c}}_{i-1} + \boldsymbol{\omega}_{i-1} \times (\mathbf{r}_{i-1} + \mathbf{d}_i) + \dot{b}_i \mathbf{e}_i & \text{for prismatic joint} \end{cases}$$

Expressing in $\underline{i^{th}}$ link frame

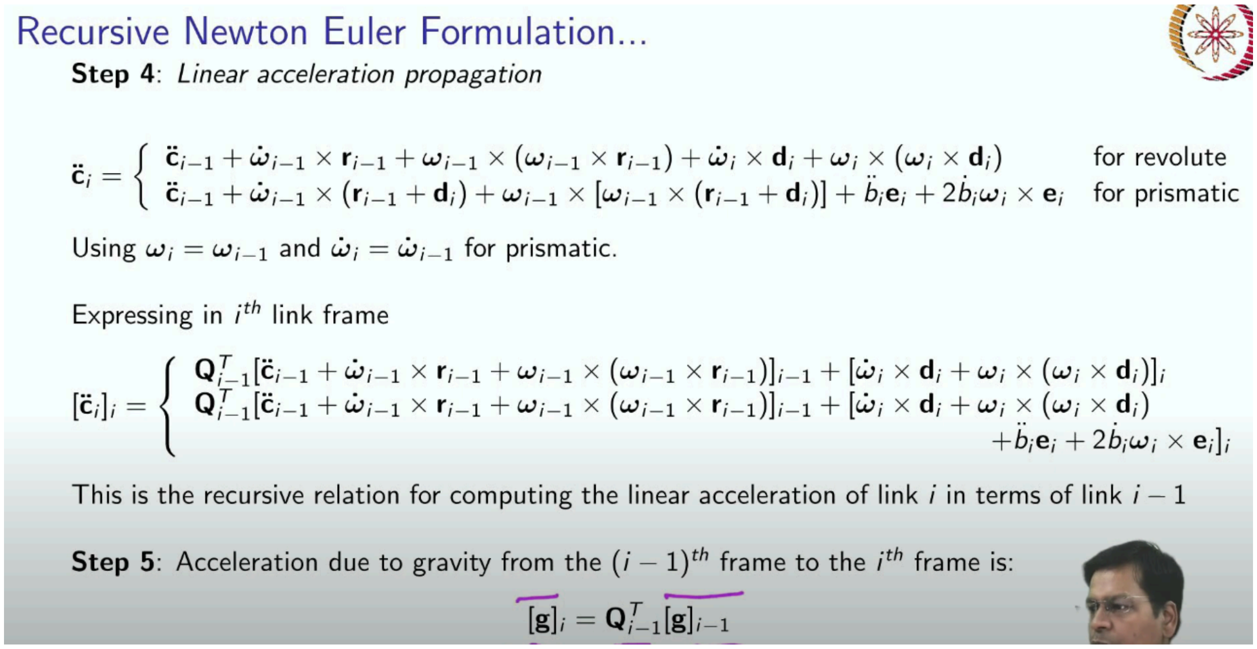
$$[\dot{\mathbf{c}}_i]_i = \begin{cases} \mathbf{Q}_{i-1}^T ([\dot{\mathbf{c}}_{i-1}]_{i-1} + [\boldsymbol{\omega}_{i-1}]_{i-1} \times [\mathbf{r}_{i-1}]_{i-1}) + [\boldsymbol{\omega}_i]_i \times [\mathbf{d}_i]_i & \text{for revolute joint} \\ \mathbf{Q}_{i-1}^T ([\dot{\mathbf{c}}_{i-1}]_{i-1} + [\boldsymbol{\omega}_{i-1}]_{i-1} \times [\mathbf{r}_{i-1}]_{i-1}) + [\boldsymbol{\omega}_i]_i \times [\mathbf{d}_i]_i + \dot{b}_i [\mathbf{e}_i]_i & \text{for prismatic joint} \end{cases}$$

where $\underline{[\boldsymbol{\omega}_i]_i = \mathbf{Q}_{i-1}^T [\boldsymbol{\omega}_{i-1}]_{i-1}}$ for prismatic joint.

And $[\mathbf{d}_i]_i = [\mathbf{a}_i]_i - [\mathbf{r}_i]_i$ in which $[\mathbf{a}_i]_i \equiv \begin{bmatrix} a_i c\theta_i \\ a_i s\theta_i \\ b_i \end{bmatrix}$ $[\mathbf{r}_i]_i \equiv \begin{bmatrix} r_{ix} \\ r_{iy} \\ r_{iz} \end{bmatrix}$

This is the recursive relation for computing the linear velocity of link $i$ in terms of link

So, yes, hope it is quite clear. So far, it is not more than engineering mechanics, once again vectoring and yes, I am using matrices over here again to convert everything to link frame, with link frame. Again, you have to multiply with Qi minus 1, transport. It is inverse, actually. So, where, in the case of the prismatic joint, it was like this: again, I have taken a derivative of the same. So, omega i in its frame is this in the case of the prismatic joint. So, in the case of prismatic joint, it is very, very trivial. You know that we have just carried it out from the previous slide. Now, what are these parameters? So what is this? This is basically [ai]i. That can be written as we have written it earlier also. So, ai cos theta, ai sin theta, and theta are actually the joint variables. In the case of prismatic joint, bi is also a variable, but it may be a constant in the case of rotary joint. So, this is the ai vector, that is, the vector which connects oi to oi plus 1 at 3, and this is the frame in case of a link. oi minus 1 to oi likewise. This is the ri vector. So, ri has got 3 components. Those are put as rx, ry and rz. So, this is again a recursive relation for computing the linear velocity of link i in terms of link i minus 1. This is link i minus 1. So, this is done.



## Recursive Newton Euler Formulation...

**Step 4**: *Linear acceleration propagation*

$$\ddot{\mathbf{c}}_i = \begin{cases} \ddot{\mathbf{c}}_{i-1} + \dot{\boldsymbol{\omega}}_{i-1} \times \mathbf{r}_{i-1} + \boldsymbol{\omega}_{i-1} \times (\boldsymbol{\omega}_{i-1} \times \mathbf{r}_{i-1}) + \dot{\boldsymbol{\omega}}_i \times \mathbf{d}_i + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{d}_i) & \text{for revolute} \\ \ddot{\mathbf{c}}_{i-1} + \dot{\boldsymbol{\omega}}_{i-1} \times (\mathbf{r}_{i-1} + \mathbf{d}_i) + \boldsymbol{\omega}_{i-1} \times [\boldsymbol{\omega}_{i-1} \times (\mathbf{r}_{i-1} + \mathbf{d}_i)] + \ddot{b}_i \mathbf{e}_i + 2\dot{b}_i \boldsymbol{\omega}_i \times \mathbf{e}_i & \text{for prismatic} \end{cases}$$

Using $\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1}$ and $\dot{\boldsymbol{\omega}}_i = \dot{\boldsymbol{\omega}}_{i-1}$ for prismatic.

Expressing in $i^{th}$ link frame

$$[\ddot{\mathbf{c}}_i]_i = \begin{cases} \mathbf{Q}_{i-1}^T [\ddot{\mathbf{c}}_{i-1} + \dot{\boldsymbol{\omega}}_{i-1} \times \mathbf{r}_{i-1} + \boldsymbol{\omega}_{i-1} \times (\boldsymbol{\omega}_{i-1} \times \mathbf{r}_{i-1})]_{i-1} + [\dot{\boldsymbol{\omega}}_i \times \mathbf{d}_i + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{d}_i)]_i \\ \mathbf{Q}_{i-1}^T [\ddot{\mathbf{c}}_{i-1} + \dot{\boldsymbol{\omega}}_{i-1} \times \mathbf{r}_{i-1} + \boldsymbol{\omega}_{i-1} \times (\boldsymbol{\omega}_{i-1} \times \mathbf{r}_{i-1})]_{i-1} + [\dot{\boldsymbol{\omega}}_i \times \mathbf{d}_i + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \mathbf{d}_i) \\ \qquad + \ddot{b}_i \mathbf{e}_i + 2\dot{b}_i \boldsymbol{\omega}_i \times \mathbf{e}_i]_i \end{cases}$$

This is the recursive relation for computing the linear acceleration of link $i$ in terms of link $i-1$

**Step 5**: Acceleration due to gravity from the $(i-1)^{th}$ frame to the $i^{th}$ frame is:

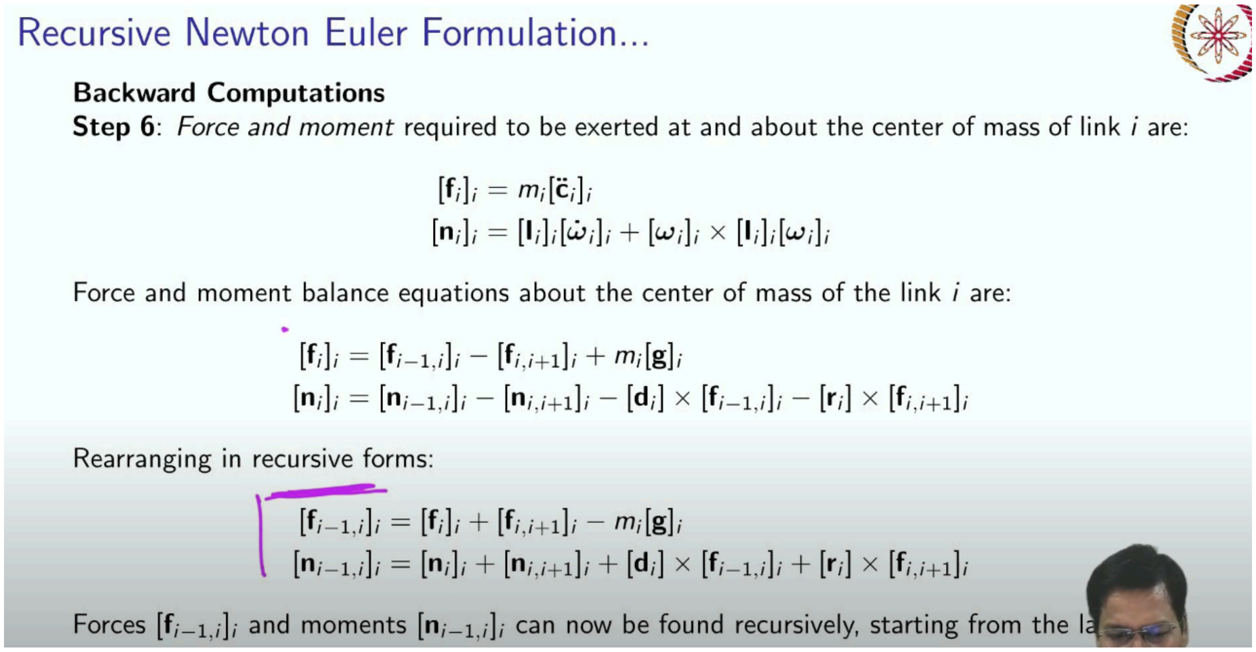
$$[\mathbf{g}]_i = \mathbf{Q}_{i-1}^T [\mathbf{g}]_{i-1}$$

So, this is linear velocity contribution and then finally, you have linear acceleration propagation. I did nothing; I simply took the derivative of the earlier equation, and I got to this. This includes everything. If it is a vector, it has to be a vector if it is not changing. Vector can change in two number of ways. First, due to the change in its magnitude and change in its direction and direction, if it is changing, you have to take the cross product with the direction. So, that is what is done over here because the RI, if you take the cross product of omega and r, either omega is changing or r is changing. Again, if r is changing, r can change in 2 different ways. So, r can be just changed like this, or it can take this form, so that is what is to be understood first. So, r can change either directly by changing the magnitude of r or by changing the direction of r. These 2 are independently given like this. So, you have to take a derivative accordingly. So, all the parameters which were seen over here are to be considered that way. So, this is the term which I am talking about. So, you have to take the derivative of r in 2 times. First, the r dot will come. Next time, you have another omega cross r, so that is what is here. So, you have omega dot r, and when you have to take this, you have to take omega cross r. that comes once again. So, r dot, because you know r is a constant term, so r dot does not exist. So, you only see omega cross r, same over here. In the case of omega and d, you again did you know d is constant, so it is this, so this is the second term, so that is all it is done. And in the case of the prismatic joint, you have b double dot. That will come, and this is 2 times because one of them has arisen due to the derivative over here. So, overall, this is the linear acceleration propagation. Again, this is to be put in terms of ith frame. Everything is to be put in ith frame because we have to conclude with ith frame first.

Then, we have to do a recursive calculation. So, every time we move forward, calculate the velocity: i plus 1, i plus 2 till n. these are all the recursive relations. So, if you know omega i minus 1, you calculate omega i. If you know Ci minus 1 double dot so that you can calculate Ci double dot, got it. So, you have to move forward that way. So, you are moving from i minus 1. i finally up till n. This is how you are moving recursively in the forward direction. So, acceleration velocity is to be done like that. So, this is how linear acceleration is also done. So, we are done with all those. So, now acceleration due to gravity is also being converted to the frame.

$$[g]_i = Q^T_{i-1}[g]_i\text{-}1$$

So, you again have to use the same transformation matrix, that is, the orientation matrix change. So, that is how it is done. G is consecutively changed from the previous frame to this frame. Again, this becomes the previous one. The next one will come that is how you keep moving from 0 to. This starts from 0, that is, the ground frame, and now it gradually moves till the end effective frame. That is the last frame. Link name. So, that is how g is also converted to the with a frame every time it is considered for that particular name. So, that is how, in 5 steps, you did all with velocities and acceleration and g.



## Recursive Newton Euler Formulation...

**Backward Computations**

**Step 6**: *Force and moment* required to be exerted at and about the center of mass of link $i$ are:

$$[\mathbf{f}_i]_i = m_i[\ddot{\mathbf{c}}_i]_i$$
$$[\mathbf{n}_i]_i = [\mathbf{I}_i]_i[\dot{\boldsymbol{\omega}}_i]_i + [\boldsymbol{\omega}_i]_i \times [\mathbf{I}_i]_i[\boldsymbol{\omega}_i]_i$$

Force and moment balance equations about the center of mass of the link $i$ are:

$$[\mathbf{f}_i]_i = [\mathbf{f}_{i-1,i}]_i - [\mathbf{f}_{i,i+1}]_i + m_i[\mathbf{g}]_i$$
$$[\mathbf{n}_i]_i = [\mathbf{n}_{i-1,i}]_i - [\mathbf{n}_{i,i+1}]_i - [\mathbf{d}_i] \times [\mathbf{f}_{i-1,i}]_i - [\mathbf{r}_i] \times [\mathbf{f}_{i,i+1}]_i$$

Rearranging in recursive forms:

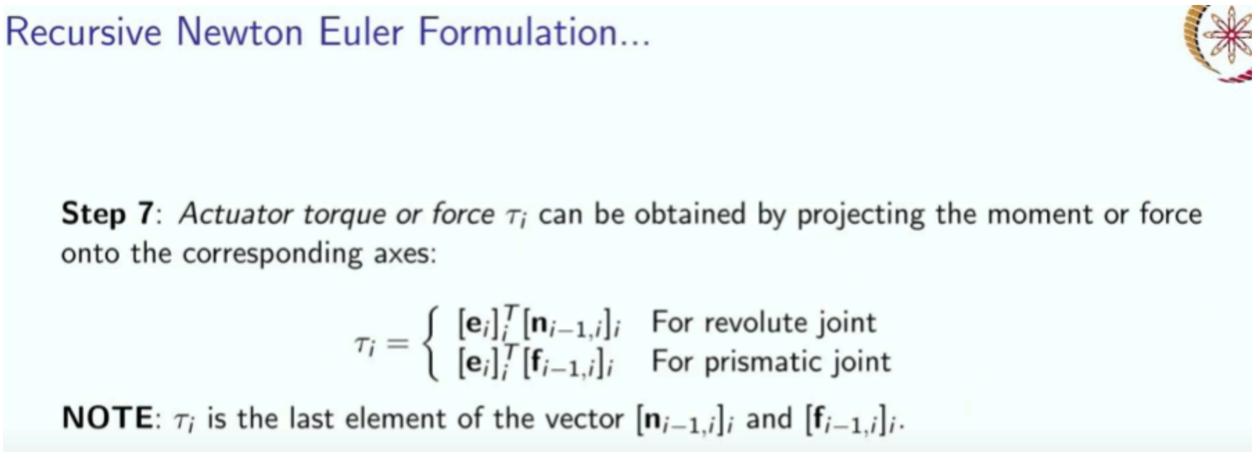
$$[\mathbf{f}_{i-1,i}]_i = [\mathbf{f}_i]_i + [\mathbf{f}_{i,i+1}]_i - m_i[\mathbf{g}]_i$$
$$[\mathbf{n}_{i-1,i}]_i = [\mathbf{n}_i]_i + [\mathbf{n}_{i,i+1}]_i + [\mathbf{d}_i] \times [\mathbf{f}_{i-1,i}]_i + [\mathbf{r}_i] \times [\mathbf{f}_{i,i+1}]_i$$

Forces $[\mathbf{f}_{i-1,i}]_i$ and moments $[\mathbf{n}_{i-1,i}]_i$ can now be found recursively, starting from the la

Now, we will use backward computation. Now that I know all the angular velocity, I know all the linear velocities and acceleration, you know omega, you know the omega dot, you know the c dot, and you c double dot also. You have all the velocities and acceleration in hand while doing forward computation for all the links. Now, you can use this equation to calculate the forces and moments. So, force is nothing but mi Ci, double dot, mass into the acceleration of that. Similarly, the moment will be this term. This is an additional term. You know it is the gyroscopic moment that is going to come, so that is also considered here. So, in terms of recursive relation, these two fundamental equations that we have done earlier also, so that is to be used. This was a similar equation that we have used over here. I will just show you here.

So, this is a similar one when you have converted the equations: previous force, this force, mass and velocity, and acceleration forces. So, all the forces.Are here, all the moments are here. It is exactly a similar equation, but yes, this is a recursive thing that we want to discuss. So, you see,

terms are very, very similar. This is, again, 2 forces that are acting on the link. This is the link: own force due to the gravitational force that makes the total force.
Similarly, the moment is because of the moments that come here. That comes here. That is corner leading. The final moment is over there. These are the forces that are causing the moment on this link. So, total moment and total force are something like this. So, it is again a recursive relation. You see, it moves in a backward manner. This time it is in a backward manner. This time again, I will tell you. So, if you have arranged it in a recursive fashion, what effectively you did, you have to arrange it in a backward manner. So, this time, because you know this can be computed, because you already know the terms which are before this, all, all the terms are rearranged in a recursive manner. You know, you know, already this so you can calculate the previous one. So, this time, this is the link, this is the link, this is the link. So, effectively, you start from here, where you know the forces that are acting here, and then you calculate at this step, adding this gravity component. You use this, use this and calculate this, use this, use this and calculate this, use this and this and calculate this. Not just the gravity component there are other components also that will come. So, that is what is contributing to this. So, finally, you are calculating in a backward manner because you know everything at the end frame. So, that is why you are moving backwards. So, I have rearranged the same force and moment balance equation in this manner. Now, you know most of the terms. So, backward way.



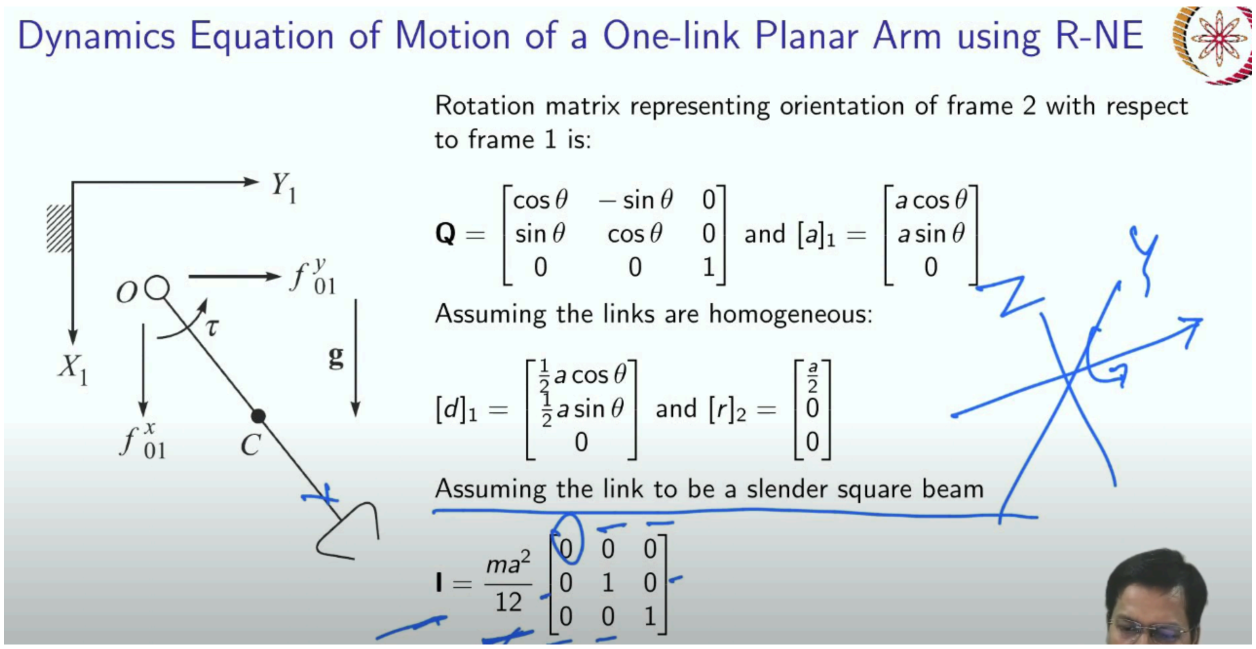
## Recursive Newton Euler Formulation...

**Step 7**: *Actuator torque or force* $\tau_i$ can be obtained by projecting the moment or force onto the corresponding axes:

$$\tau_i = \begin{cases} [\mathbf{e}_i]_i^T[\mathbf{n}_{i-1,i}]_i & \text{For revolute joint} \\ [\mathbf{e}_i]_i^T[\mathbf{f}_{i-1,i}]_i & \text{For prismatic joint} \end{cases}$$

**NOTE**: $\tau_i$ is the last element of the vector $[\mathbf{n}_{i-1,i}]_i$ and $[\mathbf{f}_{i-1,i}]_i$.

And finally, when you calculate all the moments, you project them using an ei vector. ei vector can be extracted using link transformation matrices, that is, using forward kinematics, and you can project all the moments to the link joint axis vector, and you get the torque along that axis. So, tau i is the last element of the vector. This tau i is again: in the case of the revolute joint, you have to use this and project. In the case of the prismatic joint, this is converted to a force that is acting on the prismatic joint. This is how, in 7 steps, you do all the recursive calculations.



## Dynamics Equation of Motion of a One-link Planar Arm using R-NE

Rotation matrix representing orientation of frame 2 with respect to frame 1 is:

$$\mathbf{Q} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ and } [a]_1 = \begin{bmatrix} a\cos\theta \\ a\sin\theta \\ 0 \end{bmatrix}$$

Assuming the links are homogeneous:

$$[d]_1 = \begin{bmatrix} \frac{1}{2}a\cos\theta \\ \frac{1}{2}a\sin\theta \\ 0 \end{bmatrix} \text{ and } [r]_2 = \begin{bmatrix} \frac{a}{2} \\ 0 \\ 0 \end{bmatrix}$$

Assuming the link to be a slender square beam

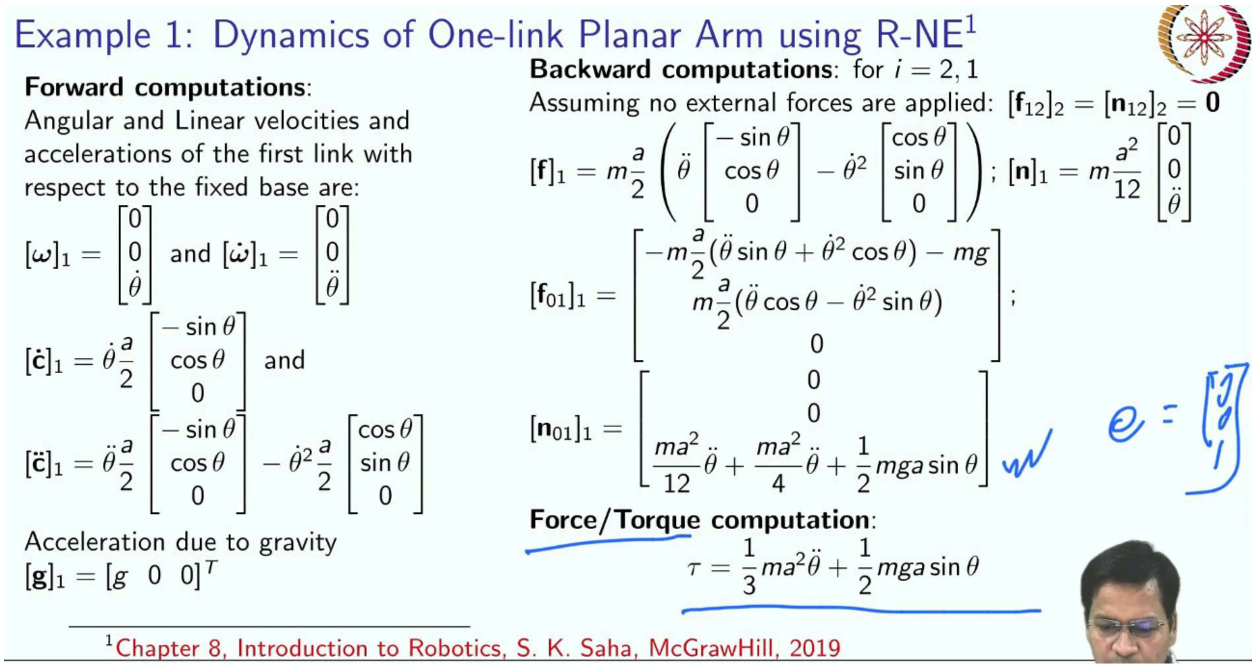
$$\mathbf{I} = \frac{ma^2}{12} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Let us quickly see one example here. So, this is just a single link manipulator, recursive Newton Euler formulation to do this. So, this is the joint. I have assumed X directed downwards, that is, along g, which is shown here. X1, Y1, 2 is the frame which is at over here. So, there are 2 frames only. So, about the first one, the link is moving, and a link is moving with the second joint, which is placed on top of it. There is no second link. So, this is the tau i want to obtain. These are the forces, reaction forces that act over here. So, I have just isolated them. I will just check what are the forces that are going to come. So, this is the transformation matrix. This is nothing. But expressed as frame 2 with respect to frame 1. This is nothing but rotation about the Z-axis which is perpendicular to X1, Y1. So, that is, rotation about the Z axis is given by cos theta minus sine, sine, and cos 1 goes here. This is the orientation matrix of this link that will represent frame 2 with respect to frame 1. This is ai in the first frame when it is represented in the first frame. So, if this is link length a, link length a, link a will can be represented, as in the first frame, as a cos theta, a sine theta and 0. These are nothing but components along X component along Y. So, cosine goes here, sine a goes here, and 0 because it is frame 2, and it is 0 because it is a planar system. So, these are some variables that I quickly obtain.

Again, assuming the links to be homogenous, I am assuming this link has uniform mass, it is slender, and mass is uniformly distributed all over. So, the centre of gravity is exactly in the midway, somewhere over here. This is your C1 location. So, yes, d1 and r1 can be given as this. So, you know what they are. So, this is your d1, this is your r1. So, again, they have to be represented in different frames, you know. So, this is nothing, but a1 by 2 should give you d1, so it is d1.

Similarly, r2 is in its frame, that is, in the link frame itself. So, it is a by 2. There is no cosine and theta because it is a constant, so it is a by 2, 0, 0; that is, there is only one variable, that is a by 2 along that direction. So, that is the only thing that it can see. This is d and r. So, yes, what is the assuming this link to be a slender, square beam? Exactly a thin beam, very, very thin beam.

Product moment of inertia will be absent. You will only see the moment of inertia of this link along its Y-axis and Z-axis, about the X link. The X-axis lies along this because it is slender. There is no moment of inertia along there, got it? So it will only have a moment of inertia, which is orthogonal to this along these two directions, got it? This is Y, and this is Z. So, that is given by mlma$^2$ by 12, you know already. The moment of inertia of a thin cylinder bar about its centre of gravity is given by this. So, that is I. So, all product moment of inertia is 0, moment of inertia about this axis X is 0. So, this is the principal moment of inertia that remains. So, this is what your inertia tensor can be written as. So, you have obtained this many variables.



## Example 1: Dynamics of One-link Planar Arm using R-NE[1]

**Forward computations**:

Angular and Linear velocities and accelerations of the first link with respect to the fixed base are:

$$[\boldsymbol{\omega}]_1 = \begin{bmatrix}0\\0\\\dot\theta\end{bmatrix} \text{ and } [\dot{\boldsymbol{\omega}}]_1 = \begin{bmatrix}0\\0\\\ddot\theta\end{bmatrix}$$

$$[\dot{\mathbf{c}}]_1 = \dot\theta\frac{a}{2}\begin{bmatrix}-\sin\theta\\\cos\theta\\0\end{bmatrix} \text{ and}$$

$$[\ddot{\mathbf{c}}]_1 = \ddot\theta\frac{a}{2}\begin{bmatrix}-\sin\theta\\\cos\theta\\0\end{bmatrix} - \dot\theta^2\frac{a}{2}\begin{bmatrix}\cos\theta\\\sin\theta\\0\end{bmatrix}$$

Acceleration due to gravity

$$[\mathbf{g}]_1 = [g \;\; 0 \;\; 0]^T$$

**Backward computations**: for $i = 2, 1$

Assuming no external forces are applied: $[\mathbf{f}_{12}]_2 = [\mathbf{n}_{12}]_2 = \mathbf{0}$

$$[\mathbf{f}]_1 = m\frac{a}{2}\left(\ddot\theta\begin{bmatrix}-\sin\theta\\\cos\theta\\0\end{bmatrix} - \dot\theta^2\begin{bmatrix}\cos\theta\\\sin\theta\\0\end{bmatrix}\right); \; [\mathbf{n}]_1 = m\frac{a^2}{12}\begin{bmatrix}0\\0\\\ddot\theta\end{bmatrix}$$

$$[\mathbf{f}_{01}]_1 = \begin{bmatrix}-m\frac{a}{2}(\ddot\theta\sin\theta + \dot\theta^2\cos\theta) - mg\\ m\frac{a}{2}(\ddot\theta\cos\theta - \dot\theta^2\sin\theta)\\0\end{bmatrix};$$

$$[\mathbf{n}_{01}]_1 = \begin{bmatrix}0\\0\\\frac{ma^2}{12}\ddot\theta + \frac{ma^2}{4}\ddot\theta + \frac{1}{2}mga\sin\theta\end{bmatrix}$$

$$e = \begin{bmatrix}0\\0\\1\end{bmatrix}$$

**Force/Torque computation**:

$$\tau = \frac{1}{3}ma^2\ddot\theta + \frac{1}{2}mga\sin\theta$$

[1]Chapter 8, Introduction to Robotics, S. K. Saha, McGrawHill, 2019

Now, omega: omega in its frame can be written as OK. Omega 1 is nothing but 0, 0. There is no rotation about X and Y, and there is only rotation about the Z axis. So, it is theta dot, and you know it is theta dot. Rate of change of joint angle. Again, the omega dot, that is, angular acceleration, is the double dot of that. That is it. Now, the velocity: so, you already know what your position is, so you already know the location. So, you can directly take a derivative of that. You can obtain Ci dot, so I did nothing. You already know that location. It was a by 2 cos theta, a by 2 sin thetas and 0. So, this was your C location in the first frame. So, this was there. You just take a derivative of this, and you can obtain this. Got it, and again, acceleration is nothing but a derivative of this. I could obtain this. So, this is your acceleration of the centre of mass, Ci. In this case, it is C1, acceleration due to gravity.

In the first frame, again, it is along the X-axis only. You know, X was like this, Y was like this so that it can be written like this. So, actually, I did not have the space can be written as g 0, 0, like this. I can write transpose like this, so this is how G is there.

Now, using backward computation. You have done with, you have done with all the forward computation. Now, I have to find out the forces and moments. So, because this link is terminal, there is no link that comes after this. So, this makes these 2 forces and moments equal to 0. That comes from the link which comes after this. So, that is equated to 0.

$$[f_{12}]_2 = [n_{12}]_2 = 0$$

Now, the force is simply mass into the acceleration, which is there. So, the acceleration, you know it, is exactly this one. So, this goes inside. That comes here. So, that is the force and the moment, and the moment is again i theta, double dot, so I have simply used this to obtain this. Again, if you write it in vector format, you can write it like this, so vector f in the first frame can be written as so you just take m a by 2, that comes here with this. This only goes here, and along X, you also see there is a force which is added here. That is, that was along X. So, the net force which comes here is due to the acceleration which is here that is given by this acceleration top row and due to the gravitational force. So, total force. Similarly, along the Y direction. You see, it is only due to the acceleration in that direction that comes here. There is no acceleration along that. So, this is your force. Again, the moment you know that acceleration, the moment you know there are no moments along X and Y. It is only due to this. You have converted that because the moment of inertia was about its centre of gravity. Now, it is about the tip. So, you have to do that transformation, and can obtain this. So, that transformation is done using a parallel axis. You can do this, or you can simply use the matrix-to-matrix transformation to do this, obtain this moment of inertia can be transformed. Qi Q transpose, you can do, and you can do it. So, that way, you can do this is equivalent to doing parallel axis theorem, applying parallel axis theorem. And this is the added torque due to the gravitational force. So, this is the net torque that acts at a joint point. So, that is the force, and this is the moment. Now, the torque computation, because this is a revolute joint, it is only the torque that matters. So, you just project this to vector. A vector which is here, in this case, ei is equal to 0, 0, 1. Only that is only first joint you have. So, you just project this using this take dot product and you can just get this. So, by adding these two, you can obtain ma square by 3, and this remains as it is. So, this is the scalar torque that is acting at the joint one. So, this is how it is applied.

So, with that, we will end today, and in the next class, I will just discuss the dynamic equation of motion of a two-link Manipulator using Newton Euler's approach. Again, vector learning we will see. So, that is all for today. Thanks,