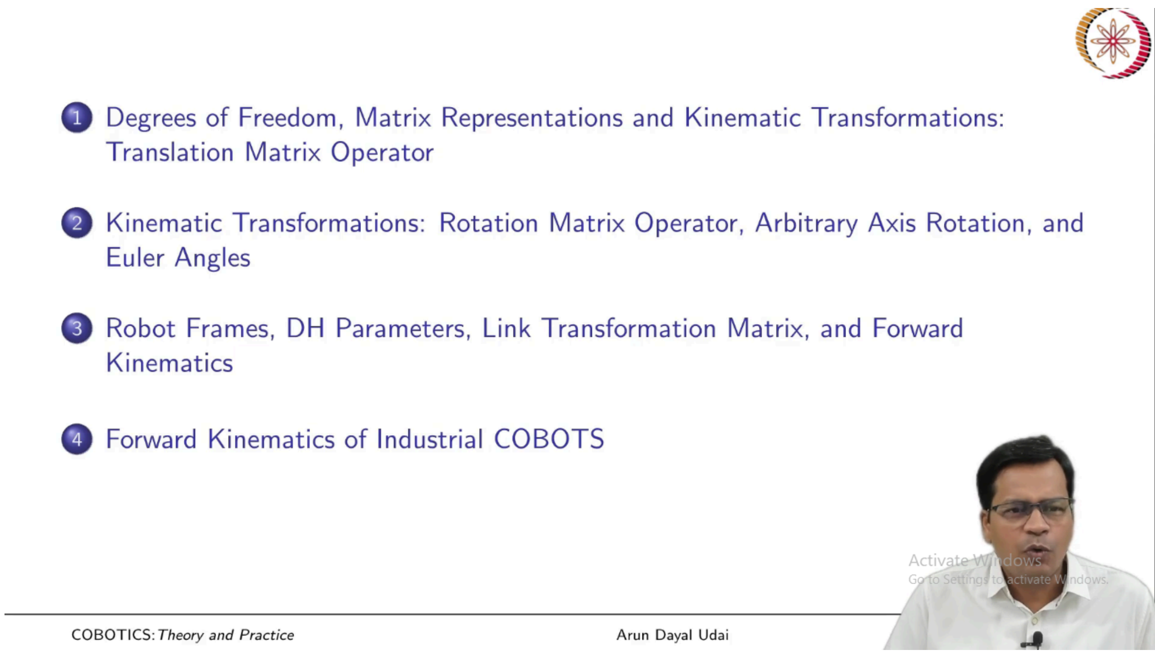
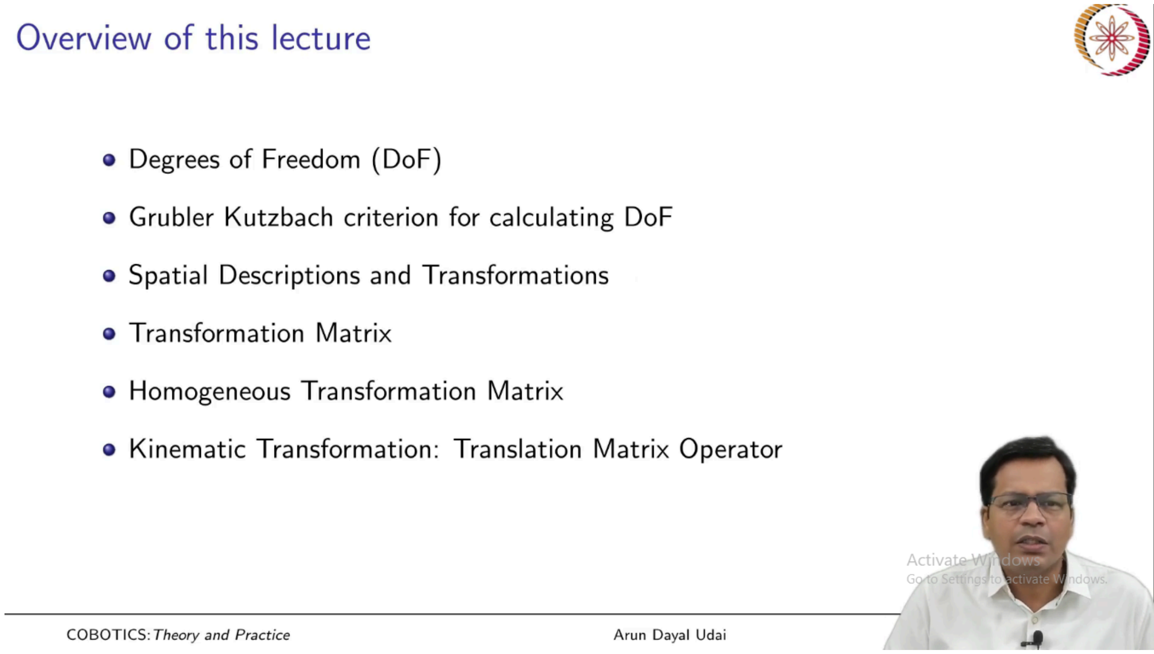
NPTEL Online Certification Courses
COLLABORATIVE ROBOTS (COBOTS): THEORY AND PRACTICE
Dr Arun Dayal Udai
Department of Mechanical Engineering
Indian Institute of Technology (ISM) Dhanbad
Week: 03
Lecture: 11

Degrees of Freedom, Matrix Representations, and Kinematic Transformations

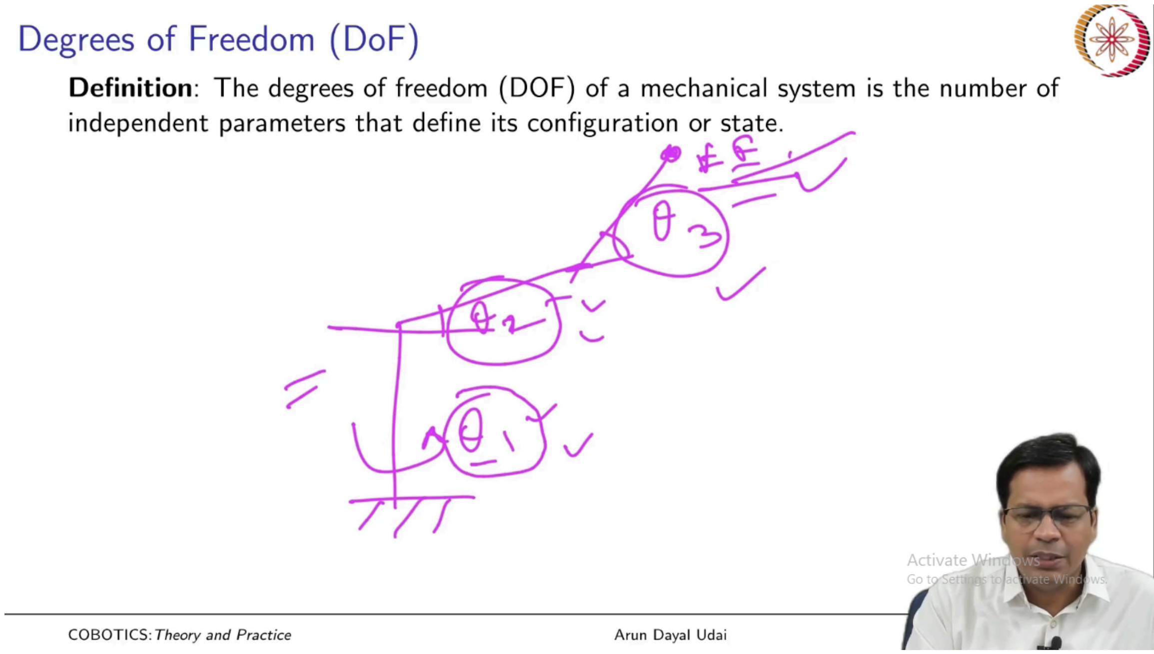


Welcome back to the third week of the course Cobotics: Theory and Practice. In this week, we will discuss the Transformation Matrix and Robot Kinematics. It will have four lectures. The first will be on Degrees of Freedom, Matrix Representation, and Kinematic Transformations. We will begin with the Translation Matrix. The second lecture will continue with Kinematic Transformation, including Rotation Transformation, Arbitrary Axis Rotation, and Euler Angles. Lecture 3 will comprise Robot Frames, Dh Parameters, Link Transformation Matrix, and Forward Kinematics. Moving ahead to the fourth lecture of this week, we will discuss the Forward Kinematics of Standard Industrial COBOTS.



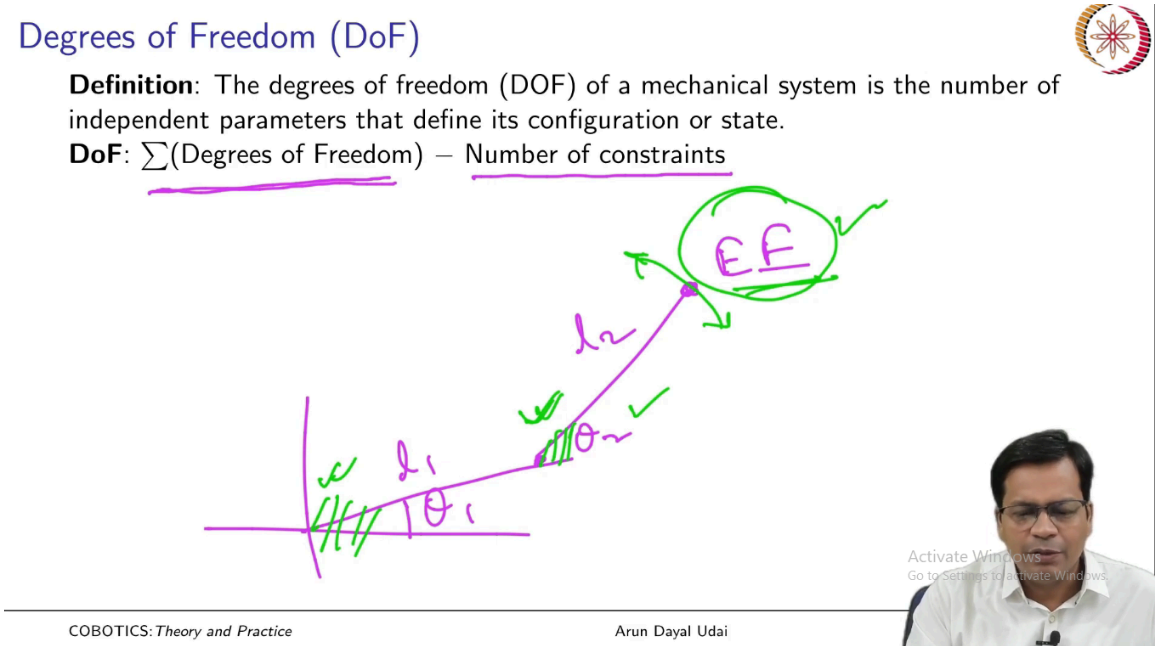
So, let us begin with the first lecture, which is on Degrees of Freedom, Matrix Representation, and Kinematic Transformation. We will start with the Translation Matrix Operator Kinematics. The overview of this lecture is as follows. We will begin with Degrees of Freedom. We will discuss Grubler Kutzbach's criterion for calculating Degrees of Freedom.

We will discuss Spatial Description And Transformation, Transformation Matrix, Homogeneous Transformation Matrix, and we will finally end with Kinematic Transformation, discussing the Translation Matrix Operators.



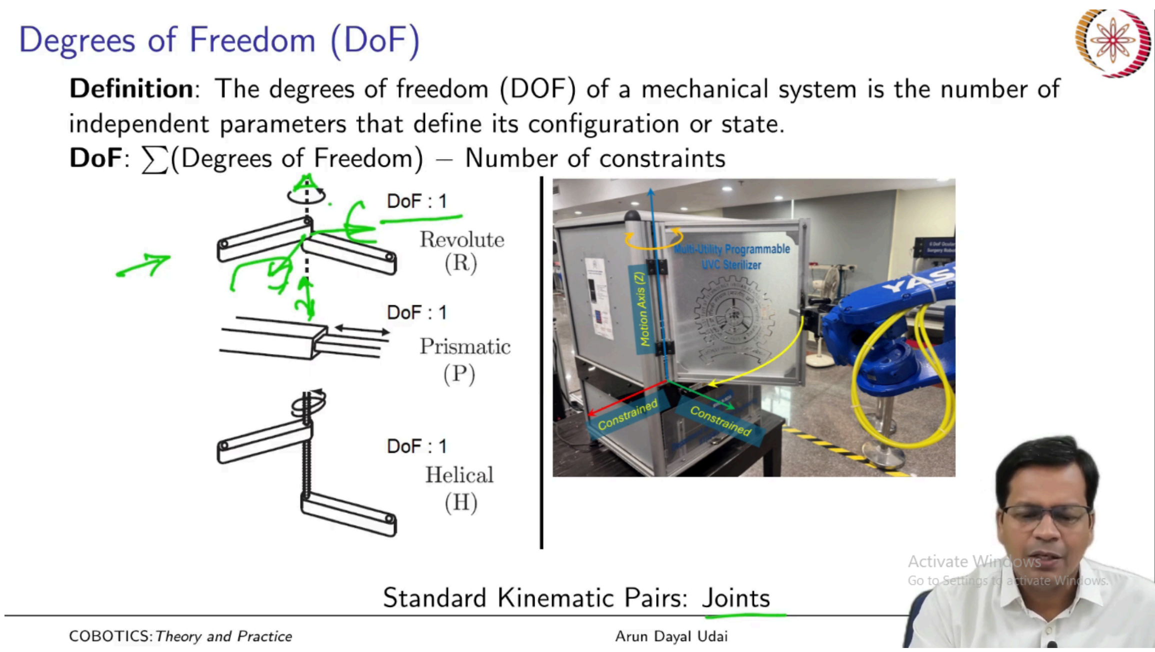
So, let us begin with Degrees of Freedom. So, yes, the Degrees of Freedom of a mechanical system is the number of independent parameters that define its configuration or state.

Let us say if it is a robot; it has multiple Degrees of Freedom. It has the first revolute joint here, the second one here, and the third one, let us say, here. We define each one of them, that is, theta 1, theta 2, and theta 3, which are the revolute joint angles, to finally define the end effector coordinates, that is, the pose of the robot. So, it needs to be defined. All three angles, theta 1, theta 2, and theta 3, are needed to fully define the state of the robot, which is defined by the end effector coordinates, that is, where it is. It is at x, y, z, or it is at x, y, z roll, pitch, yaw. So, complete information about its state is given by the three input states of its joint angle. So, this robot will now have three Degrees of Freedom. So, this is just an example, giving you using a robot for any mechanism, any parallel, serial, or any kind of mechanism or a machine, this is the same.

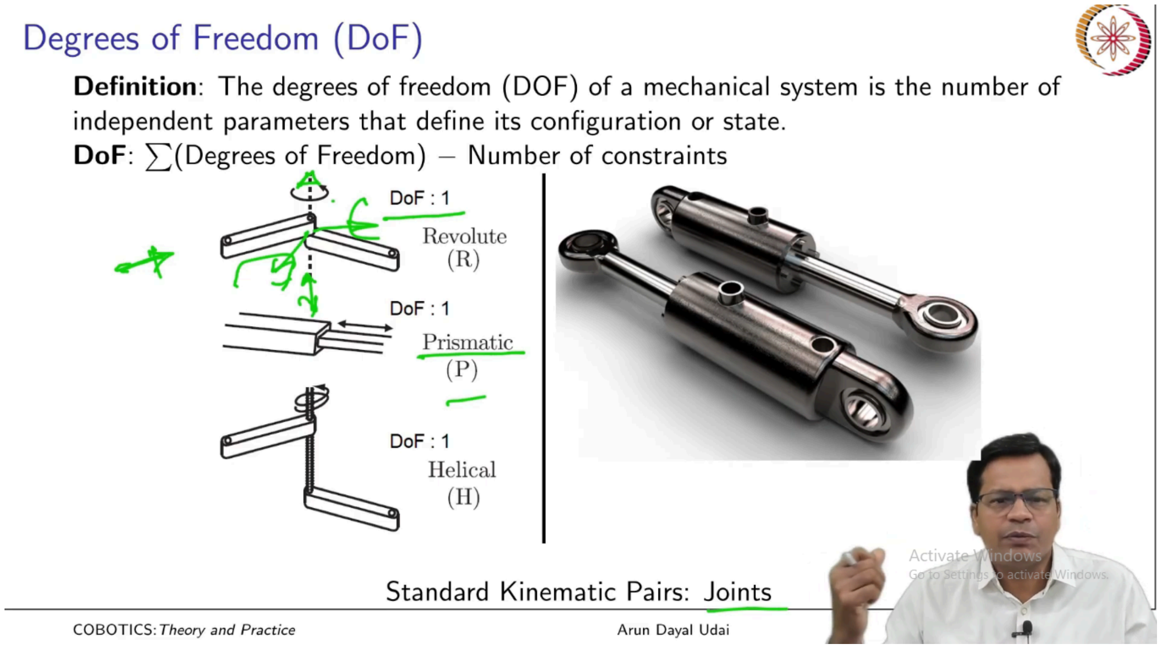


So, effectively, total Degrees of Freedom is the summation of all the Degrees of Freedom minus the number of constraints. So, the number of independent constraints that can be put on a robot so as to fully constrain a robot in terms of robots. Let's say, again, a similar robot, let's say it has two Degrees of Freedom and it has one link, link one, another link, link two, with joint angles theta one and theta two, finally defining the end effector coordinate. If I constrain this joint, the end effector can still change its coordinate by just changing the second joint angle.

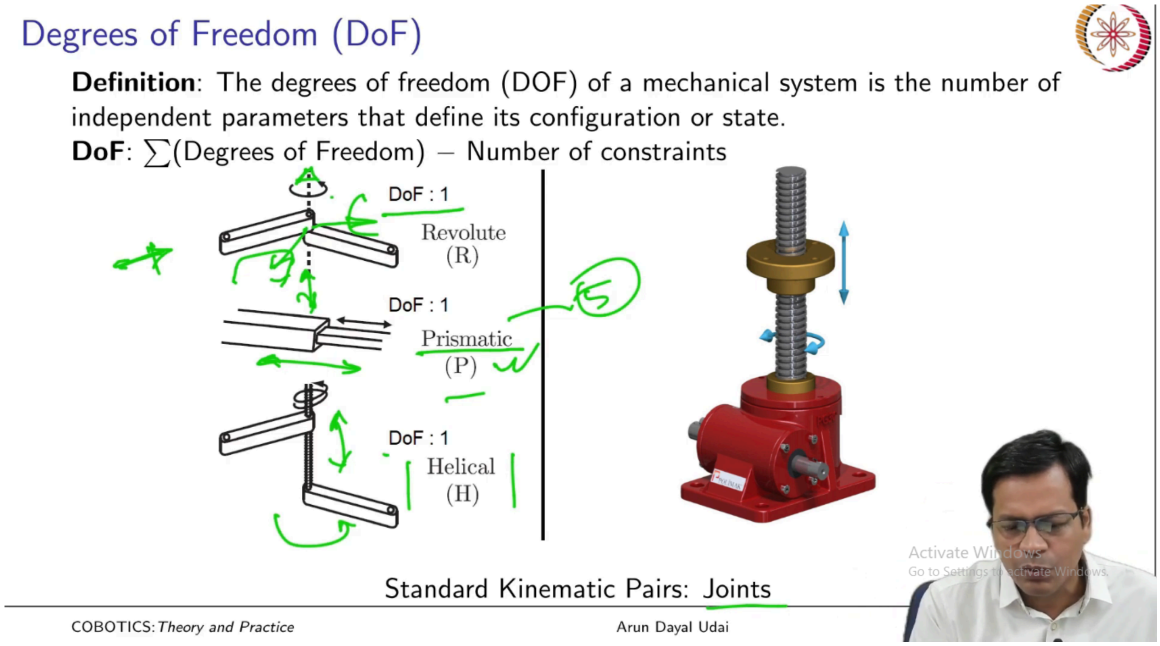
If I again weld it up fully, the second joint also, then the end effector is fully constrained. So, that means to constrain the end effector, you need to constrain all the joints. So, the number of independent constraints that you need to put so as to fully constrain the end effector or fully define the pose of the robot. So, this is what becomes the Degrees of Freedom of a particular machine, a robot, or a mechanism.



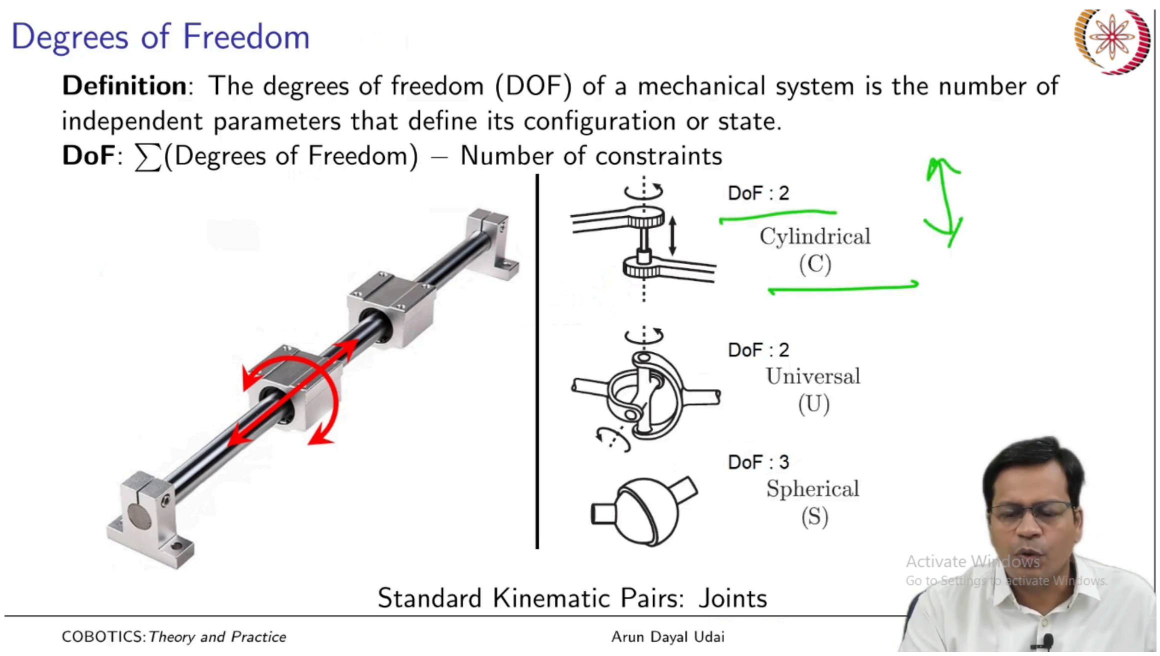
So, now let us look at various Degrees of Freedom that could be there in any joint. Standard kinematic pairs are known as joints in the case of robots. So, if it is a revolute joint, which is normally present in any of the robot joints in the case of revolute jointed robots, so if it is a robot like this, it has a joint here, and it has a joint here, okay? It has multiple joints, which finally make it up. So, these joints are normally revolute joints. So, in this case, anybody with respect to a fixed frame can have a maximum of 6 Degrees of Freedom. So, in this case, if it has 1 degree of freedom, that is just a revolute joint about one of the axes, which means the remaining five are fully constrained. So, all five constraints would include translation along this direction and this direction, and rotation along this rotation along this, and it also constrains the translation along the vertical axis. So, altogether, five Degrees of Freedom are constrained, and one of them is left free, about which this joint is defined. So, this is a revolute degree of freedom.



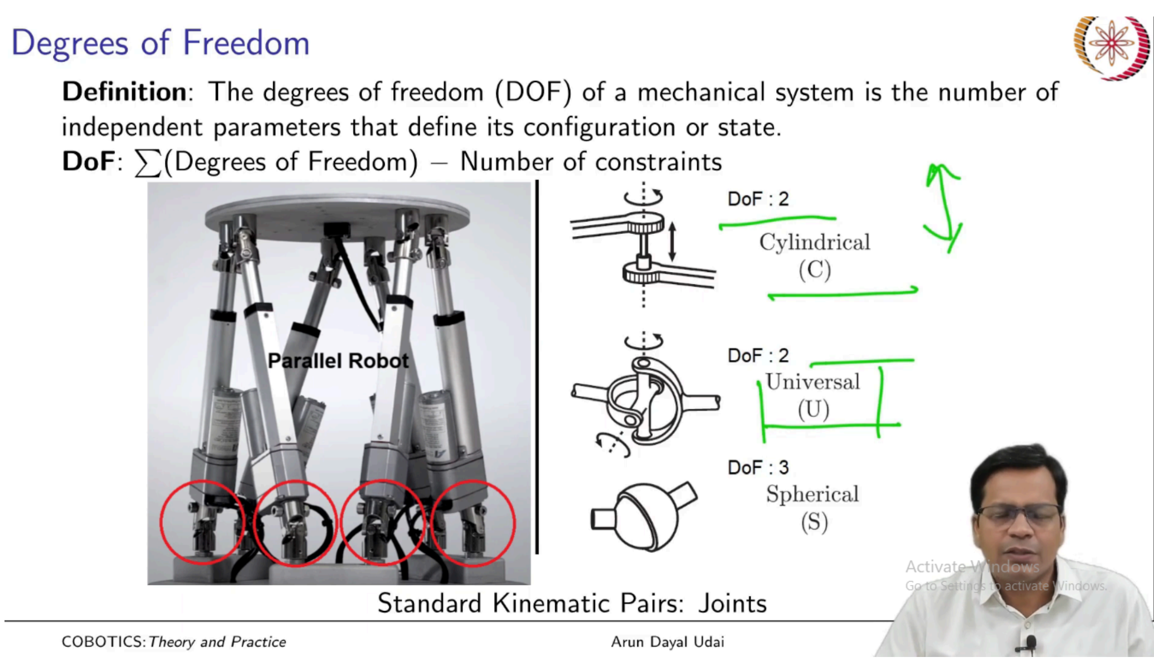
Next is a Prismatic Joint, which is quite common in the case of robots again, or it is a plunger and barrel kind of system or a hydraulic plunger if you have seen it, or a typical syringe, they have such kind of joint. So, it is a prismatic joint. So, in this case, the plunger can only go back and forth into the cylinder or out of the cylinder, right? So, it cannot rotate about any of its axes if it is shaped like this. So, it is again constrained by 5 Degrees of Freedom, and 1 degree of freedom is left free. So, this is a prismatic joint.



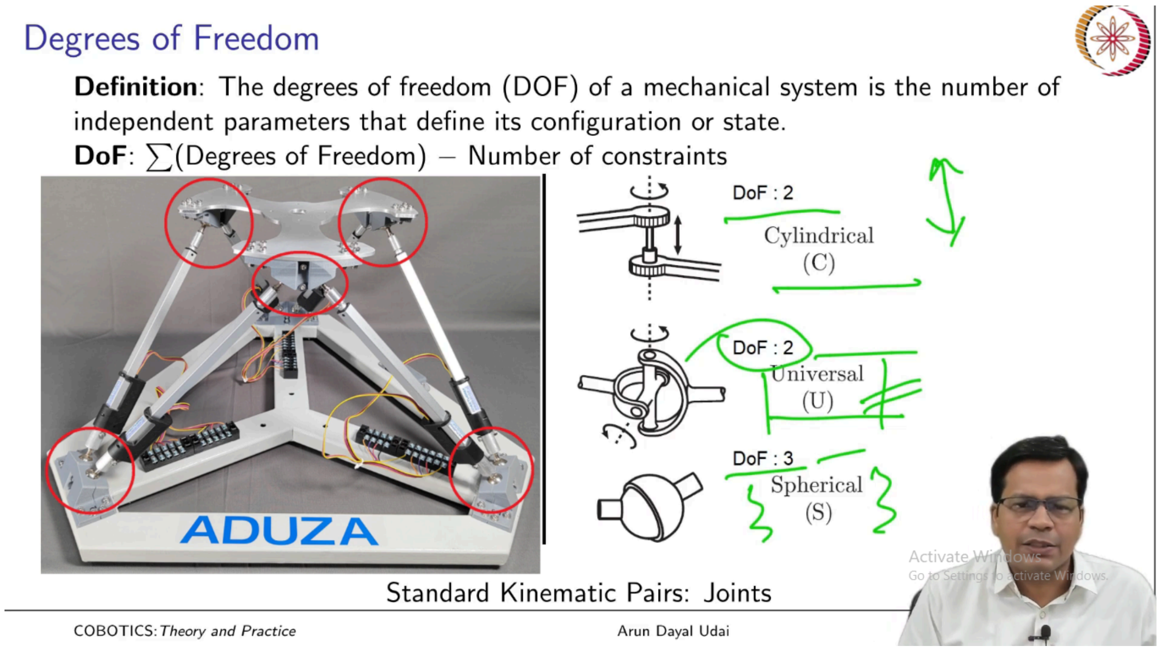
The third one here is a Helical joint. So, this is rarely used in robots. So, if you rotate, you also move in one of the directions. You also translate in this direction. So, when I rotate, I also translate. That means they are not independent of each other. Okay. So, that means if you rotate, you also translate. So even though it looks like it has one revolute and one free translation direction, it is not so. It effectively leads you to one degree of freedom only because two of them are linked, which appears like free, and others are fully constrained. So, the remaining four are fully constrained; two of them are linked, finally ending you with one degree of freedom only.



Next is a Cylindrical Joint. This is very much similar to a revolute joint only, but in this case, you can move along this vertical direction also. So, two are free, and four are constrained.

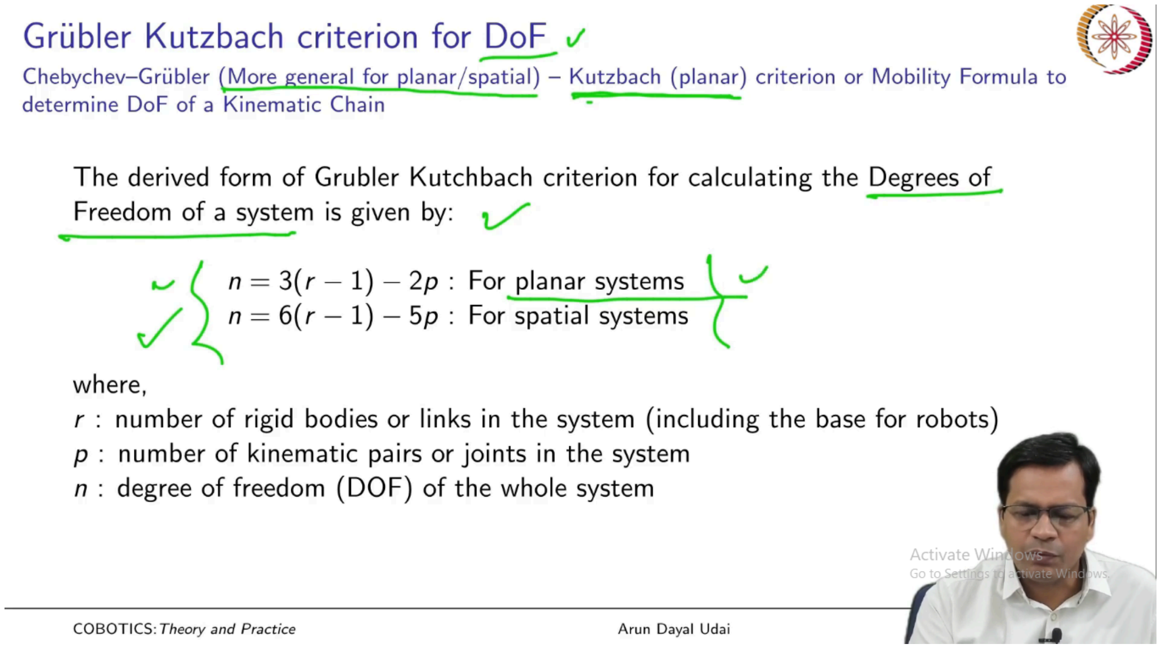


This is again, This is quite a common universal joint. It is used quite commonly in robot passive joints, normally in the case of parallel robots also. So, this is having two Degrees of Freedom, the rotational Degrees of Freedom orthogonal to each other that are free, and the remaining Degrees of Freedom, four Degrees of Freedom, are constrained. So, it is left with two DoFs. This is known as a universal joint.



The sixth one, which is the final one, has three Degrees of Freedom. It is known as a Spherical Joint. So, it can have rotation along all three orthogonal axes, and it cannot have translation in any of the directions. That means three translations are constrained, and three are horizontal. Three rotations are free, so altogether, it has three Degrees of Freedom.

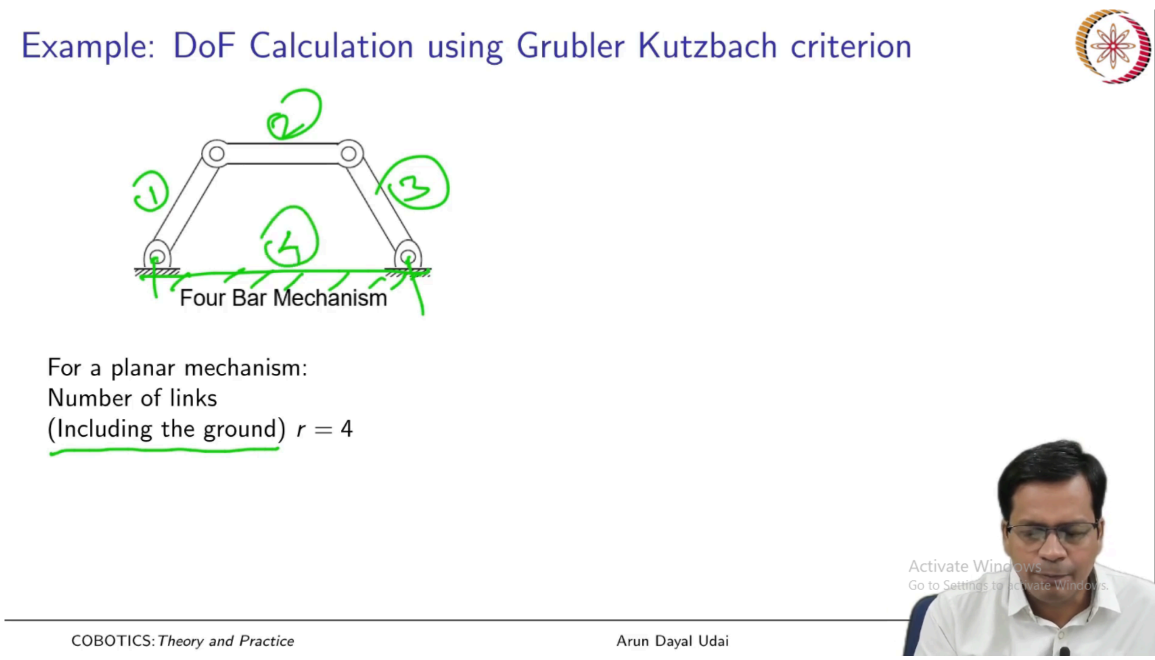
These are standard kinematic pairs, which are commonly known as joints. Two of them, the universal joint, prismatic joint, and revolute joint, are very common in robotics.



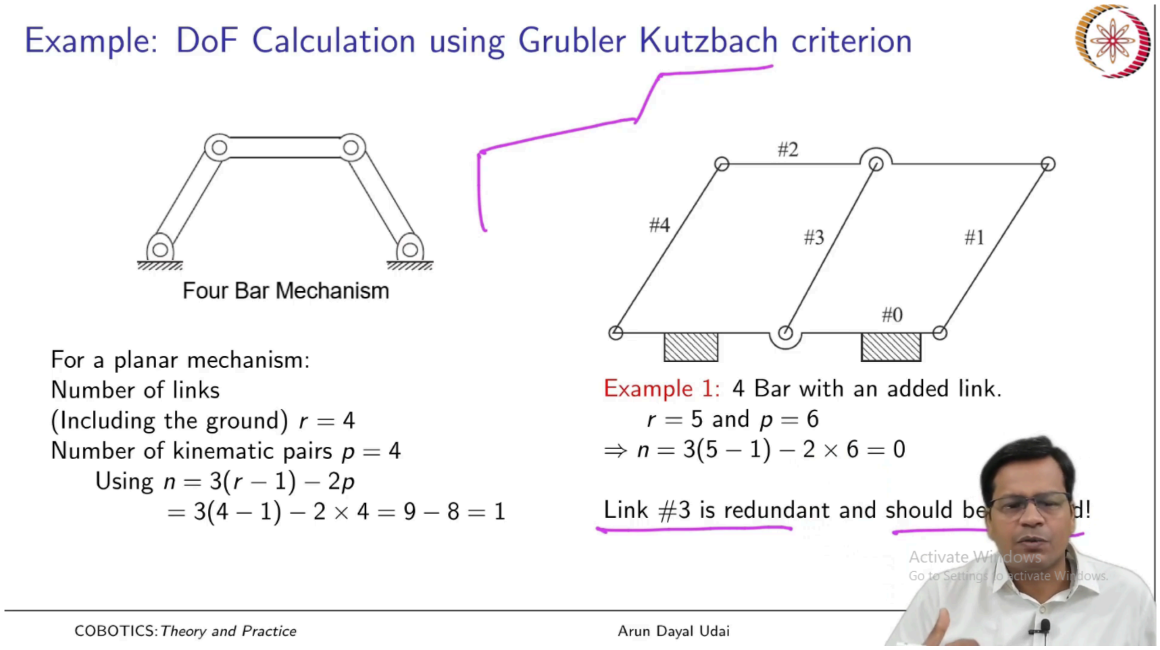
The next is the Grubler-Kutzbach criterion for Degrees of Freedom calculation. The Chebychev-Grubler-Kutzbach criterion is a more common name for it. Kutzbach gave a planar equation for that to find out the Degrees of Freedom based on conditions like the number of links it has, the number of kinematic pairs it has, and it can calculate. So, these are the two formulas.

$$n = 3(r - 1) - 2p : \text{For planar systems}$$
$$n = 6(r - 1) - 5p : \text{For spatial systems}$$

So, the derived form of the Grubler Kutzbach criterion for calculating the Degrees of Freedom of any system is given by this and this. This one was given by Kutzbach long ago, which is only for planar systems, whereas the second one was an extension to this, given by Grubler later on, which is also applicable for spatial systems. So, that is for three-dimensional systems, and this is for 2D planar systems. So, it says n is equal to 3 times r minus 1 minus 2p, and the second one is n is equal to 6 times r minus 1 minus 5p, where r is the number of rigid bodies or links in the system, including the base of the robot, and p is the number of kinematic pairs or joints in the system, and n, that is the output, is the degree of freedom of the whole system.



Let us look through one of the examples. This is a four-bar mechanism here, okay? So, the number of links, including the ground, is four: one, two, three, and the one which is connecting these two is a static link, which is not a moving link, but it is making these two points at a fixed distance. It is not allowing this distance to change, so this is also a virtual link that is placed on the ground. So, effectively, it is the fourth link. So, the total links are four. The number of kinematic pairs here is four revolute pairs. You can see one, two, three, and four, okay? So, it is a planar system, so I'll use the first formula, okay? So, it says n is equal to three times r minus one minus two p. If I substitute r and p here, what I get is So, one degree of freedom. So, this mechanism is a single-degree-of-freedom system. So, over here, you see, can you apply the degree of freedom definition here? So, the number of independent constraints need to be defined in order to fully define the system, or the number of independent constraints that I can put so as to fully constrain the system.



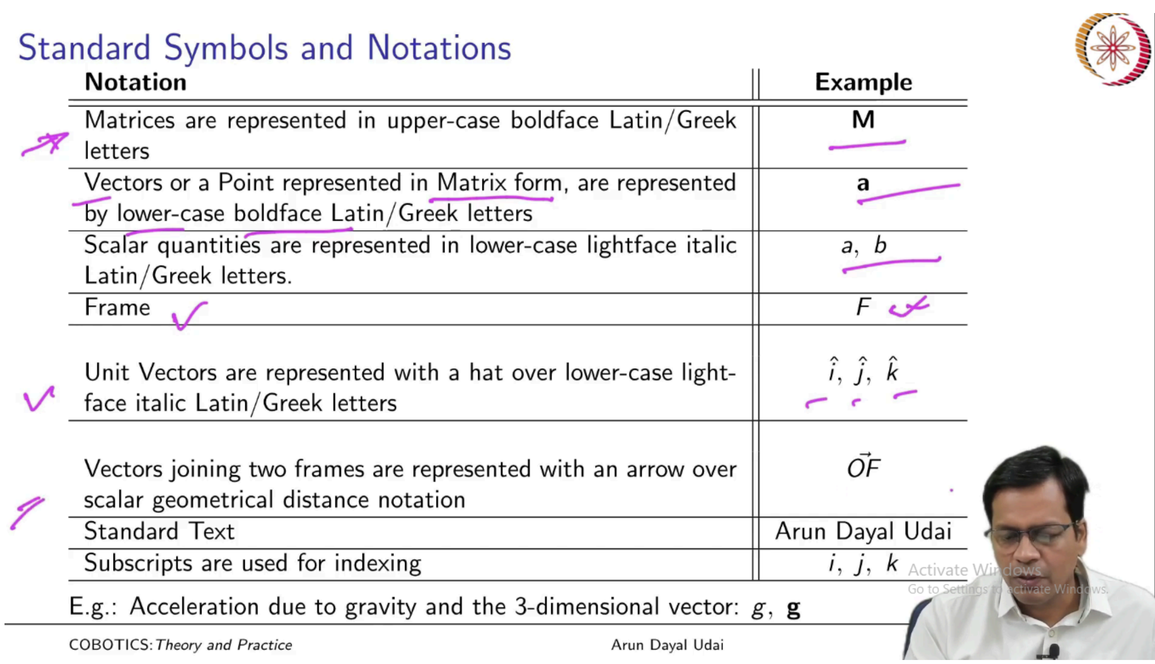
So, let us just weld one of the joints. So, if I weld this one, I have put a constraint here, which means now, if I freeze this up, what will happen? This point will be frozen. So, if this point is frozen, that means the overall system is now a triangle. So, this cannot move, this cannot move, okay. That means the remaining two are making it a triangle, which is a stiff structure and it cannot deform. None of the joints can further move. So, that means the whole of this structure is now frozen, fully constrained. This mechanism can no longer move. That means just by fixing one of the joints, the whole of the mechanism gets fixed. So, that means it has one degree of freedom.

Let us extend this example with this one. Over here, I have just added one additional link with one revolute joint and two revolute joints, and this is the additional link here. So, in this case, r, that is, the number of links, becomes equal to 5: 1, 2, 3, 4, and 5. The number of joints here is 6; that is, the kinematic pair is 6. If I put them in the planar equation, it gives me 0. Is it really zero here? If this mechanism, you all know, can at least move like this. All the links will move parallelly, and they will move like this, okay? So yes, it has some degree of freedom, which is quite obvious. But yes, using this equation, it comes to 0 here. How is it possible? Actually, you see, link three over here is a redundant link. It has no role to play, and these two joints also.

This is just supporting it. Let us say there is a huge load from here. Yes, it can help you to take up that load in order to prevent it from sagging, something like this. But definitely, it does not have any role in creating the motion or maybe creating a new type of motion anywhere in the system.

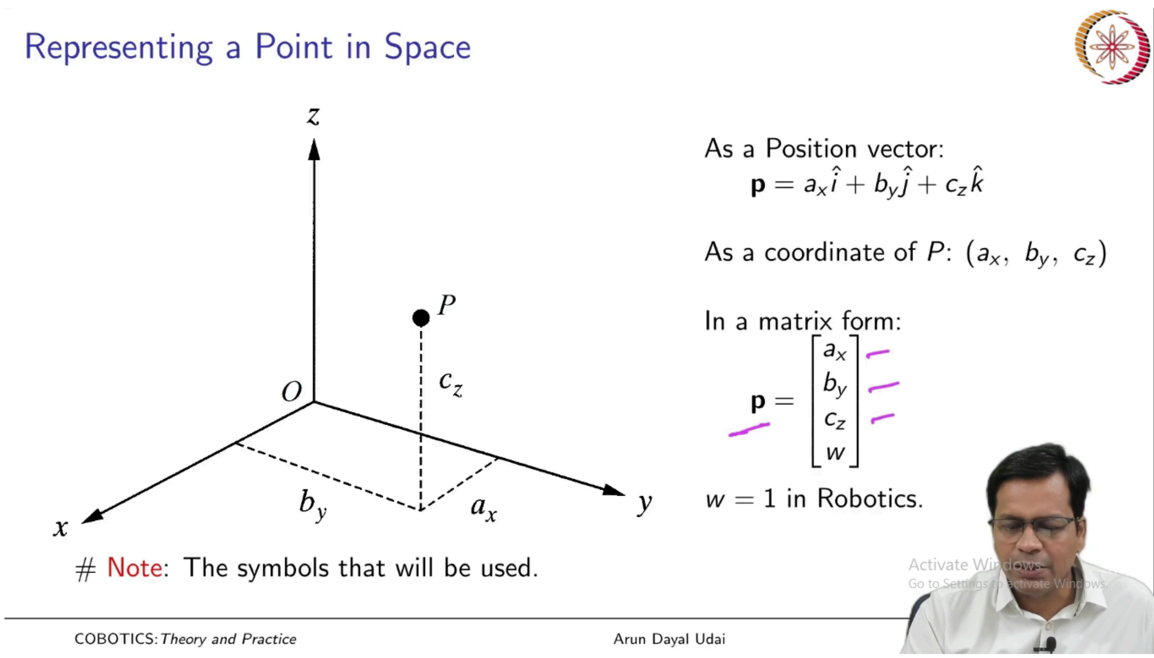
So, that means this is a redundant link that is there only to support the system. This should be removed before you calculate the degree of freedom of the system. Otherwise, this type of result is quite possible. So, link 3 is a redundant link, and it should be removed before calculating the degree of freedom.

So, yes, in the case of a serial system, as in the case of robots, industrial robots, they are just like an arm. In that case, you will find it gives you a degree of freedom exactly equal to the number of joints. That is in the case of any serial chain system like this. That is for any robot.



So, yes, let us move ahead with Spatial Description and Transformation. What is that? Before we actually move in, let us formalise the standard symbols and notations that I will be using throughout my lecture from now on. So, matrices will be represented using bold capital vectors M or any point represented in matrix form are represented using lowercase boldface Latin or Greek letters like this, scalars like this, any frame or any position in space will be denoted using capital letter italics. Unit vectors will be represented using a cap on the top and small letter italics. The vector joining the two frames, like O and F over here is given as a vector with an arrow mark on the top. Standard text will be like this, and any subscripts that are used for indexing will be like this. An example is the acceleration due to gravity, and the three-dimensional vector is this that is g and **g**. Both are the same, but one of them is a vector, and the other one is the magnitude of g, which is a scalar quantity.

So, mind it, this is very, very important because you should be conversant with these symbols and notations in order to follow my lectures.



Let us start by representing a Point in Space. This is a point, Space here, I would mean a Cartesian space where you have x, y, and z, three orthogonal axes. They are 90 degrees from each other.

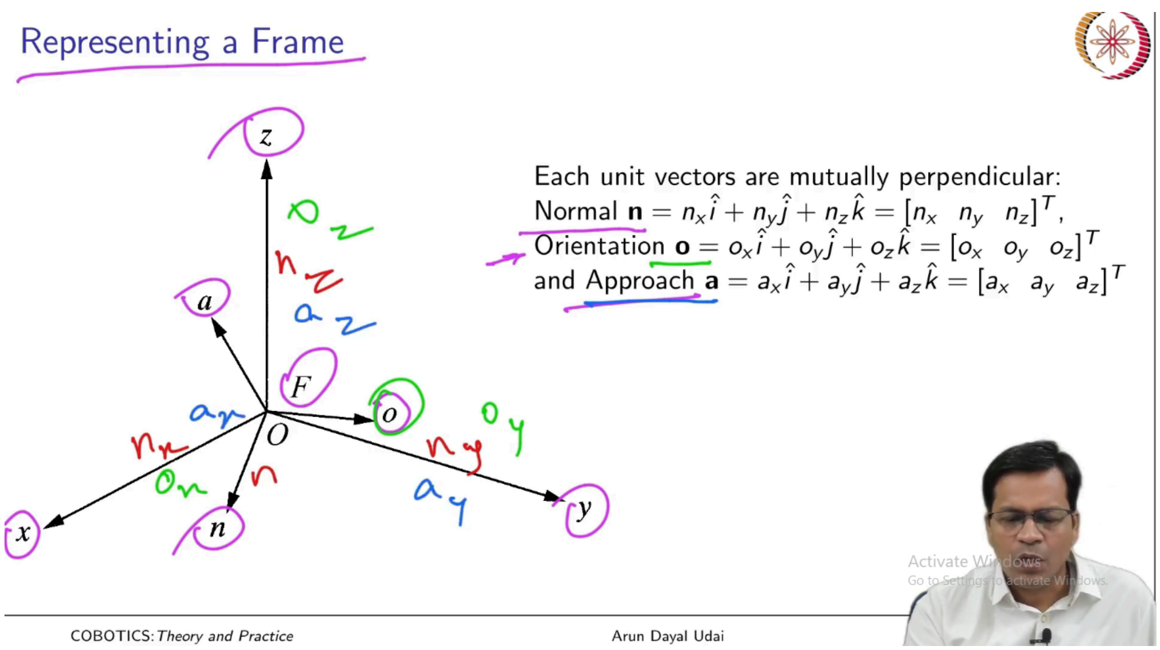
Before we begin, we are formalising this. It is not an oblique axis. It is not a cylindrical coordinate system or a spherical coordinate system. It is a Cartesian coordinate system. And any point here is given by three coordinates ax, by and cz, so these can be written as a position vector. So, any point P is the position vector, so this is your p that comes here. It may be written as ax i, by j, and cz k. So, they are i, j, and k are three unit vectors along these orthogonal axes. So, ax, by, and cz are basically the projections of this position vector along X, Y, and Z. So, this is a well-known fact. By now, you already know this.

Next, it can be represented just as a coordinate as in brackets, you can write ax, by, cz. This is another very popular format. But in the case of robots and robotics, we will be using something like this.

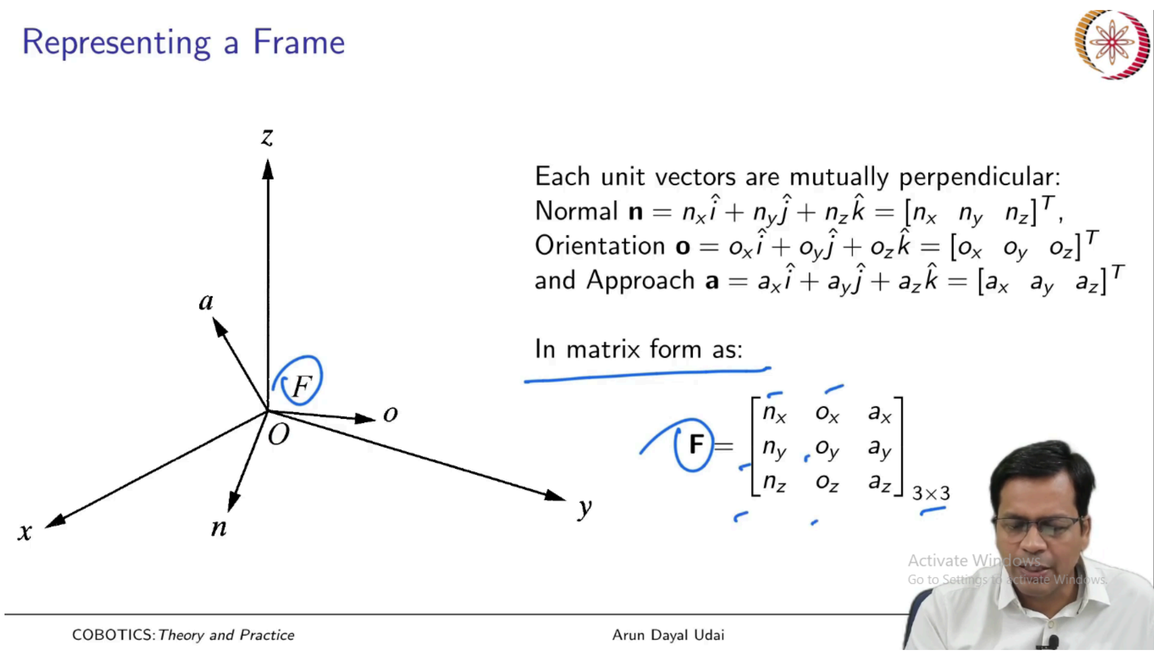
In a matrix form:

$$\mathbf{p} = \begin{bmatrix} a_x \\ b_y \\ c_z \\ w \end{bmatrix}$$

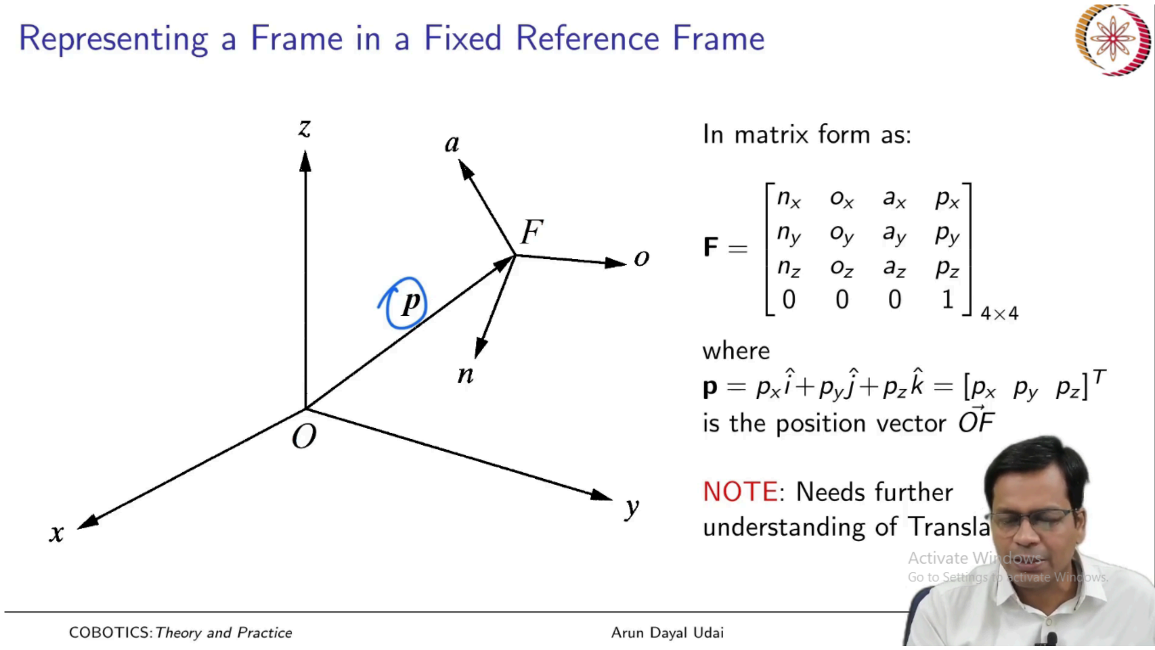
It will be denoted using a vector P, given as ax, by, cz, w. w is 1 in the case of robotics. So, effectively it will be ax, by, cz, 1. So, this is a 4x1 matrix with which a point in space will be denoted.



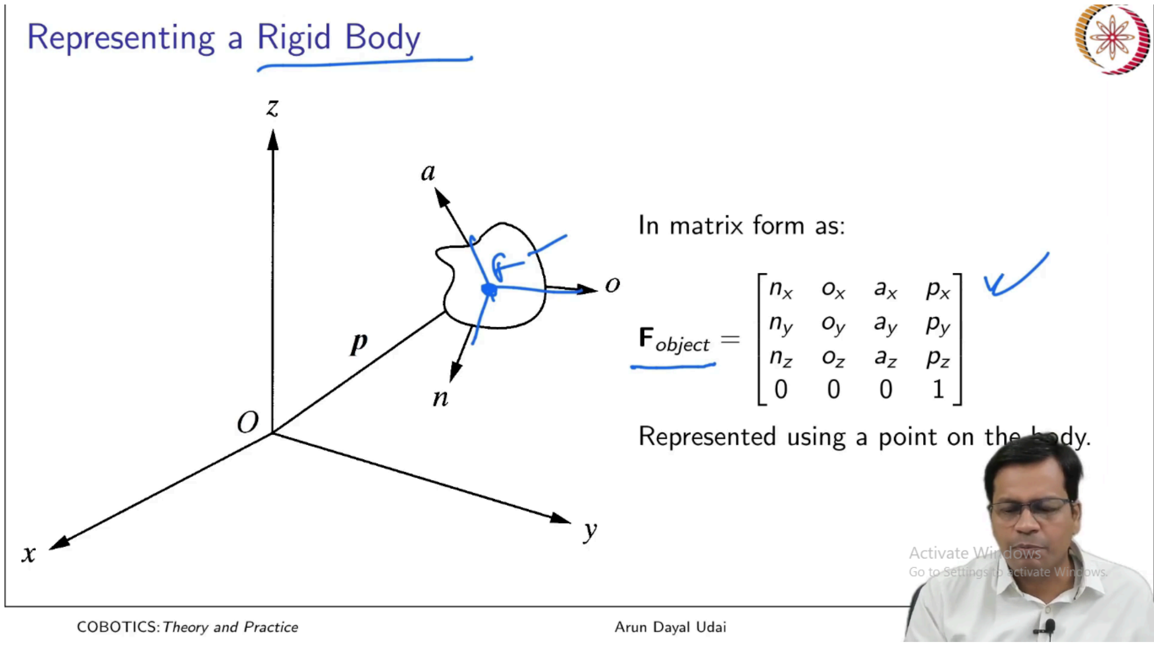
Now, representing a Frame. This frame coincides with the origin frame, but yes, you see, it is a little bit rotated. So, it is again given by three orthogonal vectors, which are mutually perpendicular to each other, normal and orthogonal. This is n, this is o, orientation and approach vector. Three orthogonal vectors are these: nx, ny, nz, ox, oy, oz, ax, ay, az. Orthogonal projections along x, y, and z. So, nx means the projection of n along x, n along y, and n along z. Similarly, for o, it is o along x, o along y, o along z. For approach a, a along x, a along y, a along z. So, these are the projections of n, o, a along three orthogonal directions. So, that finally gives me nine elements to define this fully.



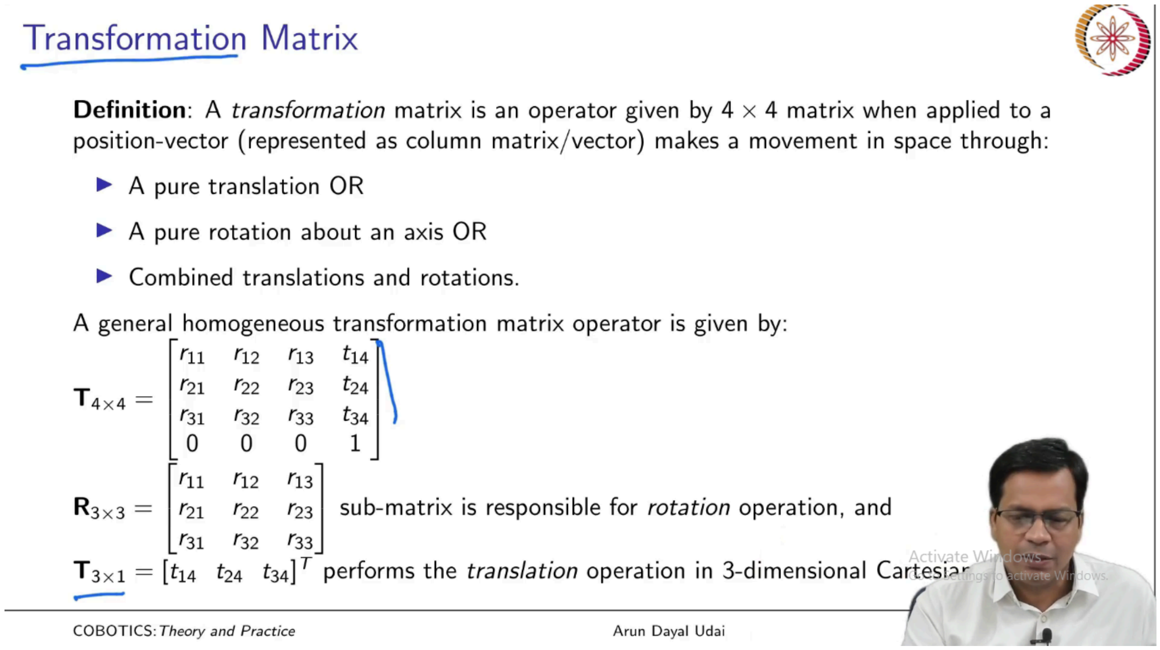
So, effectively, in matrix form, it can be written as a frame over here because it is a 3x3 matrix, it is in bold capital F.So, you see it is nx, ny, nz, ox, oy, oz, ax, ay, az. So, these nine elements finally form the matrix and, ultimately, the frame.



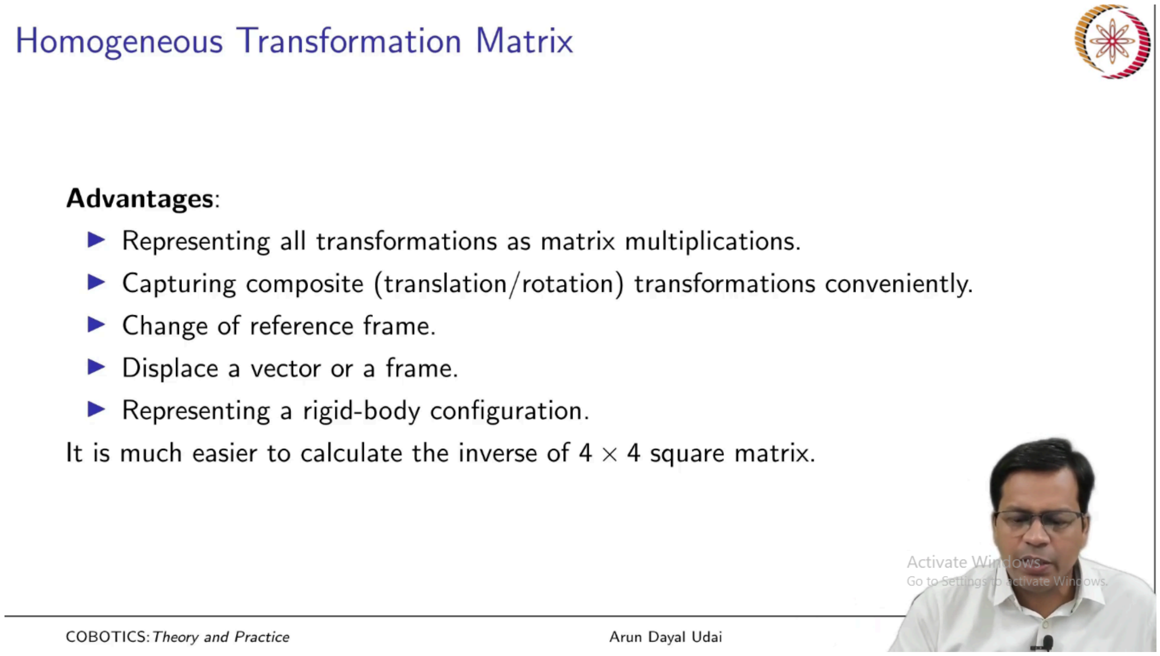
So now, the frame in a Fixed Frame means the location of this frame in the fixed frame would also include the orientation. Basically, n, o, a will finally give you the orientation of frame F with respect to frame O because it has components along x, y, and z for all three unit vectors assigned to the frame, which are n, o, and a. Here, you see there is an additional column, which is px, py, pz, and 1. That is in the standard robotics format. So, it is px, py, pz. What is that? p is the position vector from O to F. It gives you the location of F. So, P along X, P along Y, and P along Z are the projections of P along the X, Y, and Z axes. So, that finally gives you the position of the frame with respect to the origin, origin O. This is the complete representation of the frame in space that tells you the orientation using these three columns and the position using the last column, which is here. So, this finally tells you the position and orientation of the frame in space. Okay. Over here, you see it is Px, Py, Pz, which is the translation along the x, y, and z directions. We will discuss more about this later on in this lecture.



Representing a Rigid Body would now require the frame to be placed on a rigid body. A frame, once attached to the body, is enough to define the position and orientation of the entire rigid body. That means if you define the location and orientation of the frame, you have fully defined the rigid body as well because this body is not deformable. So, you see, the frame (F_{object}) of the object, which is the object frame, now becomes the same as that of the frame's location. This frame is fixed to this rigid body somewhere over here. So, whatever the position and orientation of this frame are, they become the position and orientation of this entire rigid body in space. So, this is how we will define the entire rigid body using frame F. This is represented using a point on the body. This frame is placed anywhere on the body, as a point on that rigid body. It can be any point on the body. The position will differ, but definitely the orientation of the entire frame will not change. If the location of this point changes, the position will change, but definitely the orientation will remain the same.

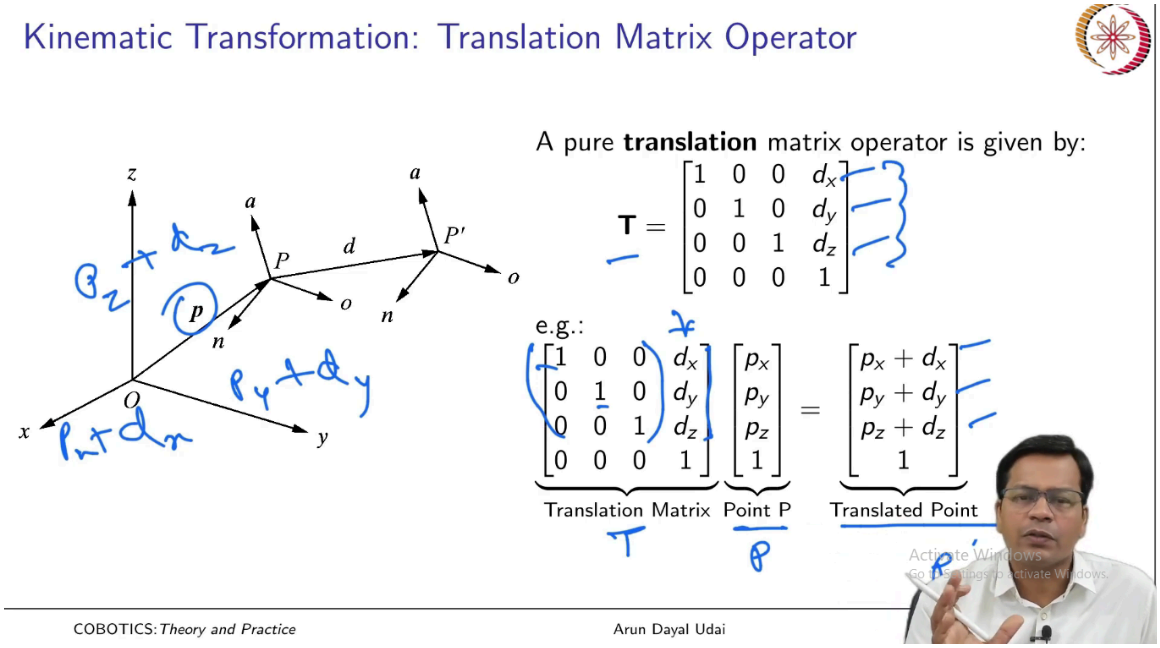


Now, let us come to the Homogeneous Transformation Matrix, Translation Matrix, and Rotation Matrix Operators. So, to begin with, we will start with the Transformation Matrix. A Transformation Matrix is essentially an operator that is given by a 4x4 matrix, which, when applied to a position vector represented as a column vector. You have seen a point, and a position is given by Px, Py, Pz, 1. It makes some movement in space. The movement can be translation, or it can be rotation. In the case of robotics, these two movements are possible. So, it can be a pure translation, a pure rotation about an axis, or it can be a combined translation and rotations, which can also be multiple rotations. So, a general homogeneous transformation matrix operator will be given as a 4x4 matrix which includes the rotation transformation matrix over here and the translation transformation matrix over here. So, we will understand these elements in detail one by one. So, what this does is basically R3x3 is a submatrix that is responsible for the rotation operation, and t3x1, which is the last column's three elements, performs the translation operation in three-dimensional Cartesian space. So, overall, this transformation matrix is to be applied to a point to transform that point, which is why it is known as a Transformation Matrix,



So, this homogeneous transformation matrix can now represent all the transformations as a matrix multiplication. So, you have to apply this transformation to a point to get to a new position or suppose it changes both the position as well as the orientation. So, it is a multiplication over here. It captures the composite that is the translation as well as rotation transformation conveniently. A single matrix can have both. It can do transformation that includes rotation as well as translation operations. It can change the reference frame, not just a position vector. It can also change the reference frame of the object. It can displace a vector or a frame. A vector can also be displaced. It can represent the rigid body configuration. So, you have seen a homogeneous transformation matrix could define the orientation and the position using this.

You have seen it earlier here. So, it is much easier to calculate the inverse of a square matrix. Now that it is a 4x4 square matrix, it becomes quite convenient to take the inverse transformation also. We will discuss all these in very much detail.



So, now let us talk about the translation matrix operator. How it happens? You see, P is the original location of this point in frame O. This is the position vector, which tells you the position of point P in the frame. O, okay, and n o a gives you the orientation. We are not talking about that. So, I just want to translate this point from location P to P dash. The pure translation matrix operator is given as this, where you saw the last column 3 element is the translation. That is what we are desiring here. This operator, when applied to the point P, you get the translated point P dash. When you multiply this transformation operator by this point P, what you get? There is no rotation. There is a unit matrix in the 3x3 rotation space of this transformation matrix. There are only elements present in the translation part of it, that is, the 3x1 part of it, which is the last column here. So, when this is multiplied with this, you see the location of this point has changed from Projections that essentially initially have px, py, and pz, but later it became px plus dx, py plus dy, and pz plus dz. So, these are the individual translations along each axis that have happened. So, those were put over here as dx, dy, and dz. So, whatever the element that goes here, the value that comes here is the value that is added along each orthogonal axis. So, this is how this is capable of transforming your point from p to p dash. This is known as a translation matrix operator.

So, in the next class, we will follow this up and discuss Kinematic Transformation. We will extend it to the Rotation Matrix Operator as well. We will see what Arbitrary Axis Rotation is and how to do that, and we will also discuss Euler Angle systems.

That is all for this lecture. Thanks a lot.