INTELLIGENT CONTROL OF ROBOTIC SYSTEMS

Prof. M. Felix Orlando

Department of Electrical Engineering

Indian Institute of Technology Roorkee

Lecture 02: Robotics: Primer-Kinematics and Dynamics

Good morning, today we have the lecture. Entitled Robotics Primer, where we focus on kinematics and dynamics of robotic systems. The outline of this lecture will be: first, we will discuss robot kinematics, in that we will be seeing both forward kinematics with examples and inverse kinematics with examples. Then, we will see the dynamics of a robotic system, in that again we see the classification of dynamics, which is forward dynamics and inverse dynamics, along with the model-based robot dynamic control.

Coming to the kinematics, what is kinematics? Kinematics is the study of motion, considering the relationship between the robot joints and the Cartesian positions reached by the robotic system. It is classified into forward kinematics and inverse kinematics. In forward kinematics, we have the inputs being the joint angles and the output being the end effector position.

So, there is only a one-to-one relationship. You give the joint angle, definitely it will go to one position. Whereas, in the case of inverse kinematics, we have the input being the end effector position and the output of the inverse kinematics is the joint angles. So, having seen the description of forward kinematics and inverse kinematics, we can say that forward kinematics is simpler as compared to that of the inverse kinematics. For the control of a robotic system, inverse kinematics can be used.

Now, let us go deep into forward kinematics. As seen from this schematic, you can observe that q, the general coordinate of the robotic system, represents a three-link robotic system. For that, we have the joint variables θ_1, θ_2, and θ_3. So, here, as I mentioned, we provide some joint angles, and it reaches this point. That means it was initially lying here, this robotic system with links L_1, L_2, and L_3, and once we provide θ_1, θ_2, and θ_3, it moves to position a. Now, for forward kinematics, the important analysis we must perform is through DH parameters, Denavit and Hartenberg.

parameters, which are four in number: a_i, d_i, θ_i, and α_i. As the name indicates, a_i and d_i are the link lengths, or precisely the length parameters, while α_i and θ_i are the angular parameters. When we say a_i, let us define what a_i is. a_i is the distance along x_i from o_i to the intersection of x_i and z_{i-1} axis. Similarly, d_i is the distance along z_{i-1} from o_{i-1} to the intersection of x_i and z_{i-1} axis. d_i is the variable if joint i is prismatic. Whereas, in the case of a revolute joint, d_i can be a constant.

Similarly, α_i is the angle between z_{i-1} and z_i, measured around the z_i axis. Finally, θ_i is the angle between x_{i-1} and x_i, measured around z_{i-1}, and θ_i is the variable if joint i is a revolute joint. We can observe through these schematics the representation of alpha and thetai. You can see that between z_{i-1} and z_i, measured around x_i, we can determine alphai. Whereas, in the case of θ_i, thetai is the angle around z_{i-1} between z_{i-1} and x_i.

So now, θ_4 is an angle around z_3. Similarly, θ_9 is an angle around z_8 between axis x_8 and x_9. In forward kinematics through the DH parameters, we must have the homogeneous transformation matrix, which is given by a combination of rotation, transformation, rotation, translation, translation, and rotation, which means it is a combination of rotation and translation. More accurately, it is first having the rotation around the z-axis by an amount θ_i, then translation along the z-axis by an amount d_i, then translation along the x-axis by an amount a_i.

And finally, rotation around the x-axis by an amount α_i. With this combination, we can get the homogeneous transformation 4x4 matrix, OK. So finally, having substituted what is rotation around the z-axis by theta and similarly for translation and rotation around the x-axis, we have this for these 4 matrices. Combined, they form this one, the final matrix. Where the last row 0 0 0 1 is the scale value that will help us in having the analysis based on DH parameters in order to have, for n degrees of freedom robotic system, the position and orientation of the end effector with respect to the base or The position and orientation of the nth frame with respect to the 0th or base frame, we can have the simplification that is aided by this scalar term or the scalar row placed in the homogeneous transformation matrix's 4th row.

Now, we are going to see the algorithm for deriving forward kinematics. First, we need to establish the joint axis of the given manipulator, the physical manipulator. What are the joint axes and term them as z_0, z_1, up to z_{n-1}. And form the base frame such that it obeys the right-hand screw rule. Label the origin of the base frame as O_0.

That means origin O and the zeroth frame. Locate the origin O_i at each joint i. That means joint 1 will have the origin O_1, and joint 3 will have origin O_3. Then establish x_i along the direction normal to the plane formed by the intersecting z-axis, z_{i-1} and z_i, or along the common normal through O_i if the z-axis is parallel to each other. That means if they are parallel, you can have x_i in this way.

If they are like this, z_{i-1} and z_i, you can have x_i parallel to this plane formed by z_{i-1} and z_i. Then form y as per the right-hand screw rule at each joint. Now coming to the example of x, Robot manipulators in forward kinematics: we first take a two-revolute-jointed non-planar robotic system, and from 0 to 1 frame, we can have the first row of the DH table. And from 1 to 2 frames, we can have the second row of the DH table. Finally, we can say that the first row contributes to the homogeneous transformation matrix represented by T^{01}, and the second homogeneous transformation matrix for the second row.

is T^{12}. Finally, T^{02}, that is the position of the tip frame with respect to the base frame 0, is given by the product of these two individual homogeneous transformation matrices, and finally, we get this from which we can say that the three elements of the last column represent the position of the end effector of the robot, and the 3x3 rotational matrix will give the orientation of the end effector, where from the 3x3 matrix, we can find alpha, beta, and gamma of the end effector. Here, we can see the simulation of this non-planar two degrees of freedom robotic system. This is basically what we have simulated in the MATLAB programming platform.

Also, we have done the forward kinematics of a hand exoskeleton developed in our laboratory, which is meant for grasping tasks, and here we can see that in the previous case, we have seen the kinematics of a serial link. Here, it is the kinematics of a kinematic model of a closed-loop link. Or a closed loop, you can say, which is a combination of both physical links or metallic links and the human being link. Because the human link is a part of the hand exoskeleton, and hence we have two closed loops coming into the picture to form the kinematic model of the hand exoskeleton. And we have tested the kinematic validation.

We have first simulated it with the kinematic model that we have made, and then we have validated it. We have moved the real system placed on the human hand and taken the trajectory, so that we could see that the two trajectories almost match perfectly, and hence our kinematic model is validated. Now, coming to the inverse kinematics, here we have the input as the end effector pose, which means the position and orientation of the end effector, and we need to find the joint variables Q, θ, and D as the output. That means, given a position B in the Cartesian space, we need to move this end effector from position A to B. So, at present, we have θ_1, θ_2, and θ_3 as the joint angular set.

We need to find a new joint angular set in order to reach B. So, that is the task of inverse kinematics. There are three ways we can do that. One is through the direct approach, another one is through the geometric approach, and the third one is closed-loop inverse kinematics based, which involves the Jacobian matrix computation. The direct or algebraic approach involves equating the final value of

Of the end-effector pose and orientation pose, and the end-effector pose matrix homogeneous matrix with the expression, and by equating it with the non-linear functions of the joint variables based on homogeneous transformation matrices expressions and the values, we can get the values of theta. Another one is using the geometrical relationship between the joints and the links, which is called the geometric approach. Okay, now coming to the closed-loop inverse kinematics approach. We can see that the block diagram of the closed-loop inverse kinematics approach for inverse kinematics is given here, where given the desired angle or desired position of the Cartesian position of the end-effector, we need to have the joint angle formed by this closed-loop inverse kinematics so that the robot tip can reach that x_d. So, as per the block diagram, the first block is the control gain, and then we have the control law, and that law will give $\dot{q}$, which is the control input to the robotic system and that is fed back to the robotic system. In terms of simulation, it is the forward kinematic model; in terms of the real system, it will be a robotic system. So, the control gain is multiplied by the error and then added to the joint inverse Okay. So, Jacobian inverse.

So, we can say that $\dot{q} = J^{-1} * \dot{x}_d + K_p * e$. That is the control law. And this control law will lead to first-order error dynamics $\dot{e}$ plus $K_p * e = 0$. This differential equation will have the solution e(t), which is going to 0 as time tends to infinity.

So, this approach of inverse kinematics ensures that this is point A, point B. You need to reach from point A to point B through various waypoints. So, this is the trajectory of the actual robotic system. So, this one, this CLIK guarantees that the trajectory between the desired and the actual is different. Almost zero because this control law, or the inverse kinematic CLIK law, takes into account the error between the Cartesian actual position and the Cartesian desired position of the end-effector.

This is how we get the first-order error dynamic equation, okay. Now, coming to the Jacobian matrix, the relationship between the end effector velocities and the joint angular velocities is given by a Jacobian matrix, where the end effector velocity is represented by a linear velocity and angular velocity combination, which is equal to J(q) $\dot{q}$ or joint angle velocity. The Jacobian matrix has two components in that it has two components.

One is the Jacobian matrix responsible for the position, and the Jacobian matrix responsible for the orientation. The Jacobian matrix is of size, in general, 6 by n, where r stands for the row, which is 6 degrees of freedom in the Cartesian space, and n stands for the column, which is the degrees of freedom of the robotic system. So, an n-degrees-of-freedom robotic system will have a 6 by n matrix. Therefore, for a 4-degrees-of-freedom SCARA robot, it will be a 6 by 4 Jacobian matrix. The individual component, each column of a robotic system or a Jacobian matrix, is given by J_p, J_i, J_{pi}, and J_{oi} being z_i and 0 vector for a prismatic joint because the orientation part will be 0 here. Whereas for a revolute joint, it is Z_{i-1} cross P - P_{i-1}, where C_{i-1}, $Z_{i-1.}$

are the third column of a rotational matrix for the i minus 1th frame, and P is the end effector position, and P_{i-1} is the fourth column of the homogeneous transformation matrix for the frame i -1. So, this Jacobian matrix is called the geometric Jacobian matrix, where we represent the end effector velocity as $\dot{x}$ equal to a 6 by 1 vector, which is a linear combination of the linear velocities $\dot{x}$, $\dot{y}$, $\dot{z}$, and instead of angle derivative, we give the angular velocity as the velocity part. Then we can say that the Jacobian matrix is a geometric Jacobian matrix. If we represent the end effector velocity in terms of $\dot{\alpha}$, $\dot{\beta}$, $\dot{\gamma}$, which means the orientation derivative, then we can say that the Jacobian involved is an analytical Jacobian matrix.

Now, the example for inner kinematics is given for a SCARA manipulator, which has four degrees of freedom. So, when projecting the manipulator's posture onto the XY plane of this manipulator, we get from the two revolute jointed robotic system easily. We get a θ_2 joint angle, which is given by this expression, and from this θ_2, we can find θ_1 and then we can go for computing θ_4. Finally, we can find the prismatic joint variable D3 from the above computed joint angles θ_2, θ_1, and θ_4. Now, coming to the statics, before we enter into dynamics, we just see what statics is.

Statics means the study of force, considering no motion of the robotic system, whereas we consider only the forces. An object, a mobile manipulator, is in contact with the wall, so there is no movement, whereas there is force interaction. So, there we will be having statics involved. The goal of statics is to determine the relationship between the generalized motion forces applied to the end effector and the generalized forces applied to the joints.

Forces for a prismatic jointed robotic system and torques for the revolute joints with the manipulator at an equilibrium configuration. Hence, the relationship $\tau = J^{-1}(q)\gamma$, where γ is the tip force, and τ is the joint torques of the robot. Now, coming to the dynamics. What is dynamics? Dynamics is the study of motion, considering the forces associated with the robotic system.

Here in this lecture, we have seen three definitions, including dynamics. The first one is kinematics. What is kinematics? It is the study of motion without considering the forces associated with the robotic system. Then we have the definition of statics, which is the study of forces when there is no movement of the robotic system.

Then we see dynamics. Dynamics is the study of motion considering the forces associated with the robotic system. The general highly coupled non-linear dynamic equation of an n degrees of freedom robotic system is given by

$$\tau = M(\Theta)\ddot{\Theta} + V\left(\Theta,\dot{\Theta}\right)\dot{\Theta} + G(\Theta)$$

where M is the inertia matrix of size n cross n and V It is called the Coriolis and centrifugal matrix, which is also of size n cross n, and G of theta is the gravity vector of size n cross 1. And obviously, $\ddot{\theta}$ is the joint acceleration vector of size n cross 1, and $\dot{\theta}$ is the joint angular velocity of size n cross 1.

So, with this, we have the left-hand side joint torque, which is also of size n cross 1. Now, dynamics has been further classified into forward dynamics and inverse dynamics. Forward dynamics means the input is torque, whereas the output is joint trajectory $\ddot{\theta}$, $\dot{\theta}$, and θ. Whereas in inverse dynamics, the input is the joint trajectories, and the output will be joint torque. Forward dynamics is used for simulation cases.

Whereas inverse dynamics is meant for controlling robotic systems. Now, having seen that the inverse dynamic model is going to be helpful for controlling robotic systems. Let us see the model-based manipulator control system. Which means we have a dynamic model given by this expression. If we assume friction, it becomes one more term, which is friction.

Then, the model-based manipulator control system will have the procedure as such. The first one is we can have the partition control law for this dynamic-based robot. The robot control is given by $\tau = \alpha\tau' + \beta$

Let that be the partition control law for this case. Again, it is given by

$$\tau = \alpha\tau' + \beta$$

where τ is the n cross 1 vector of joint arcs, and we choose α being the mass matrix or inertia matrix m of theta and beta being the rest of the dynamic model, which is given by

$$V(\Theta,\dot{\Theta})\dot{\Theta} + G(\Theta) + F(\Theta,\dot{\Theta})$$

And we consider the servo law, which is given by

$$\tau' = \ddot{\Theta}_d + K_v\dot{E} + K_p E$$

where Kp and Kv are control gain matrices with E being the difference between the desired joint angle and the actual joint angle and $\dot{E}$ being the difference between $\dot{\theta}_d - \dot{\theta}$. So, now we know the value of α, we know what tau dash is, and what beta is. So, substituting back into this dynamic model means the control law is now substituted back into the dynamic model equation 2.

Then we can have this second-order error dynamic equation, which is given by

$$\ddot{E} + K_v\dot{E} + K_p E = 0$$

which means that the error tends to 0 as time tends to infinity, and the matrices K_v and K_P are diagonal matrices. So, if we go by a joint-by-joint basis, we have the error vectors coming out to be $\ddot{e}_i + k_{vi}\dot{e} + k_{pi}e = 0$

When we have simulated for a given θ_d, $\ddot{\theta}_d$, and $\dot{\theta}_d$, we can obtain these values for giving here as input from interpolation. That means, take theta naught equal to, say, 20 degrees and theta final being, say, 45 degrees. So, this is your desired angle, and you have one initial angle for the robotic system, say, 20 degrees.

Between this and this, you can have an interpolated trajectory from which we can get theta of t, theta dot of t, and theta double dot of t, which can be given as input to this case.

Thus, in this lecture, we have seen robot kinematics, statics, and dynamics, with the concluding part as the model-based robot dynamic control. In short, we can say that Kinematics, involving both forward and inverse kinematics, has been seen with examples for both kinematics, forward kinematics, and inverse kinematics. As I want to mention, compared to inverse kinematics, forward kinematics is a simpler one. The problem or the challenges associated with inverse kinematics are, in some situations, we may get For one position of the end effector, we get several or multiple joint angles, and we need to optimize which one is the best joint angle by considering the objective functions to minimize energy or to maximize manipulability measure.

Then, coming to the statics, we have seen the expression of that, and it is advised by me to read the duality property of robot statics to have the relationship between the end effector forces and the joint torque vector. Then we have seen robot dynamics with forward dynamics and inverse dynamics involved in the model-based control. In the model-based control, given the input as joint trajectories theta d, theta d dot, theta d double dot, we are finding the joint torque vector tau by partition control law given by tau equal to alpha tau dash plus beta, but tau dash is a servo law.

Thus, we can involve both forward dynamics and inverse dynamics. Where is forward dynamics involved? Forward dynamics is involved in getting the feedback angles. That means, given the desired joint angle theta d, theta d dot, and theta, d double dot, we can get the actual system's joint variables through forward dynamics because the forward dynamics receives the joint torque, and we get the joint angle by inverting from the dynamic model.

Thus, we have seen three topics here in the robotics primer involving robot dynamics and robot kinematics. We have seen kinematics, statics, and dynamics, along with control, in this lecture. The readers are requested to go through several books and example problems in order to understand these topics. Thank you very much.