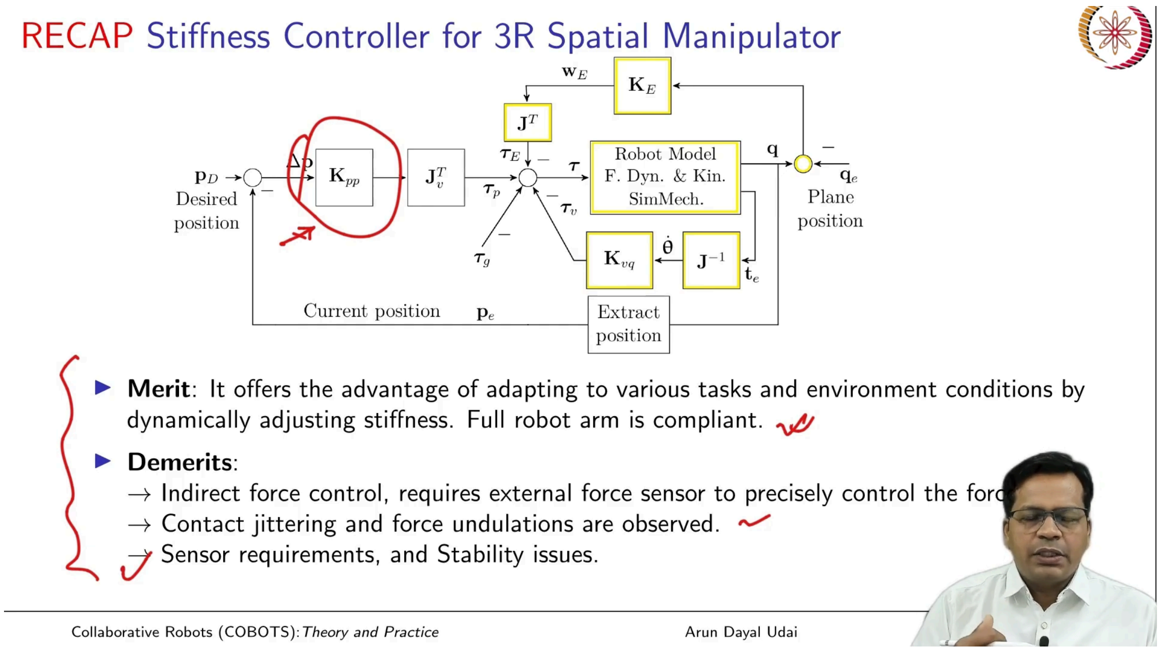
NPTEL Online Certification Courses
COLLABORATIVE ROBOTS (COBOTS): THEORY AND PRACTICE
Dr Arun Dayal Udai
Department of Mechanical Engineering
Indian Institute of Technology (ISM) Dhanbad
Week: 08
Lecture: 36

Impedance Controller and its variants

Extending the discussion on the Stiffness Controller that I covered in my previous lecture, I will move on to the Impedance Controller, which has the potential for a wide range of collaborative robot applications and is most widely used in modern cobots. The impedance controller fills the gap left by the admittance hybrid or stiffness controllers through their adaptations in different forms and their potential to be used in a wide range of applications. So, let us start with the impedance controller in this lecture.



RECAP Stiffness Controller for 3R Spatial Manipulator

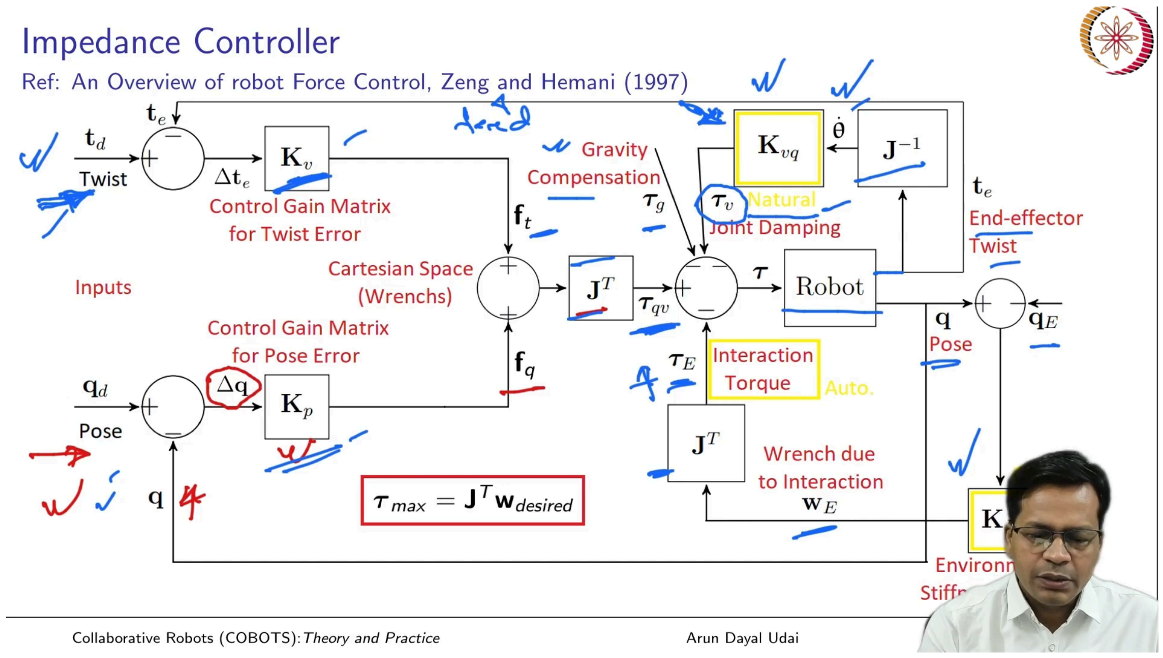
- **Merit**: It offers the advantage of adapting to various tasks and environment conditions by dynamically adjusting stiffness. Full robot arm is compliant.
- **Demerits**:
  $\rightarrow$ Indirect force control, requires external force sensor to precisely control the forc
  $\rightarrow$ Contact jittering and force undulations are observed.
  $\rightarrow$ Sensor requirements, and Stability issues.

Just have a quick recap on the stiffness controller for the 3R Spatial Manipulator that we did. So, you see, it has a few merits. It offers the advantage of adapting to various tasks and environmental conditions by dynamically adjusting the stiffness. You can dynamically adjust this (Kpp) because it is all implemented in software. So, you can dynamically adjust while the robot is already running. So, you can dynamically adjust the stiffness, and the full robot arm is compliant. So, the whole robot is compliant, not just the part that is after the force sensor, which is the case in the hybrid controller or the case of external force control loops. In this case, the whole arm is compliant.

There are a few demerits also. It is an indirect force control, which requires an external force sensor for precise control of force. If you need it for certain applications where precise force control is required, you need to have a wrist force-torque sensor, which can directly check the forces and make an external force control loop. The way we did earlier, also.

So, contact jittering, you have seen through simulation, we have observed. As soon as the robot establishes contact with the environment, it tends to bounce up and down and finally settles down. This force undulation was also observed. Sensor requirements are there. If you want to have precise force control, you should have an external force sensor. Stability issues are there. You need to adjust the control gains quite precisely. Those stability issues are also there. How much stiffness should be kept for a definite surface? So, surface-to-surface bouncing characteristics are different. So, you need to select the gains properly.

So, these are a few limitations of stiffness controllers. This can only be recovered by having a damping kind of thing. You cannot have just stiffness. So, damping is required to dampen out the jittering. So, that is what is taken care of by the impedance controller.

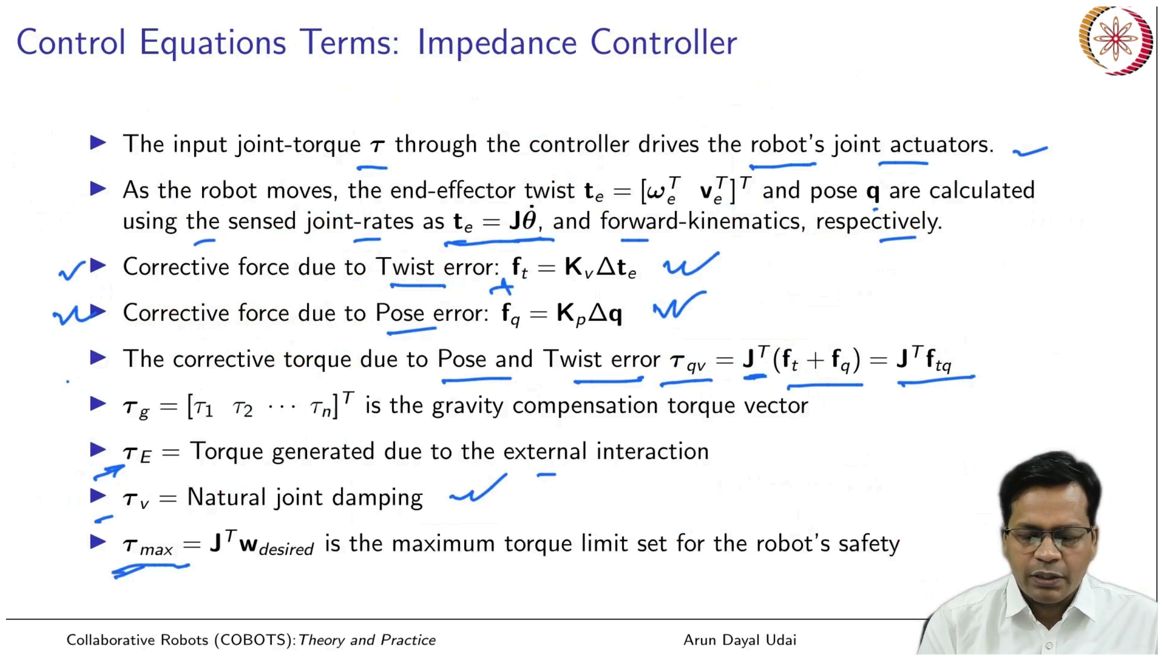


So, let us see that. So, the impedance controller apart from having the pose input (qd) and pose feedback (q). So, it calculates the difference (delta q) in the pose command, and here (Kp) is your stiffness controller, which generates the force multiplied by the Jacobian. It generates the corresponding torque due to the positional error. Not just that, on top of this. It also takes in the twist input. That is the vector of velocities angular and positional velocities. So, those two velocity components are also here. It continuously checks out through its feedback the current velocity. So, those are easily calculated the position as well as the end effector twist can be easily calculated using robot forward kinematics. So, that is what is obtained. So that is sent as feedback here. So, feedback for velocity is different in difference in velocity errors multiplied with velocity gain, or it is also known as a twist gain or impedance gain. So, this is your stiffness (Kp) part, and this is your velocity gain (Kv). So that generates the force due to the twist error (te) or velocity error to combine. It comes here and gets multiplied by the Jacobian transport, and what you get is a combined torque due to the pose as well as the velocity error. This torque finally goes to the robot, and the robot runs.

Apart from that, in the case of simulation, you already know we also had this in our previous controller. This (tau E) is the torque due to the interaction. I am calculating it through equations here because it is a simulation. You may be required to do that as well.

If at all, you are doing a simulation. So, this is the environment stiffness here multiplied by the position of the robot and the environment difference between them. So, how much the robot penetrates the wall multiplied by the environment stiffness gives you the wrench that arise due to the interaction, and the wrench multiplied by the Jacobian transpose gives you the environment torque. The torque is due to the environment, and similarly, this is the gravity compensation torque. This is always required for any controller to make it make the robot float in the air. So, it should not drop down due to its weight, and you have a natural joint damping characteristic. That is found out by checking that the end-effector twist multiplied by the Jacobian inverse gives you the joint velocity. Joint velocity into the joint damping gives you the torque. The torque is due to the joint velocity. So, this is imposed here because this is a simulation. The joint velocity that can also be obtained directly from the joints, also and if it is not a natural joint damping, you want to impose something extra so that you can always have that characteristic, you can put it here using a similar technique. In that case, joint velocity can be obtained directly from the joint encoders and joint angular sensors. So, this generates a tau v, which is the joint damping torque.

So, there are many torques here. Firstly, due to the pose and twist error, which is what we want to implement, that is, kp and kv, they were there, which generates ft and fq. Multiplying with the Jacobian transpose gives you the pose and velocity error torque. Next is interaction torque, next is gravity compensation torque and the damping torque (tau v) combined actually goes to the robot joints and mind it, this robot is a joint torque actuated robot, not a position controlled robot. So, this is what is impedance controller is.



Control Equations Terms: Impedance Controller

- The input joint-torque τ through the controller drives the robot's joint actuators.
- As the robot moves, the end-effector twist $\mathbf{t}_e = [\boldsymbol{\omega}_e^T \;\; \mathbf{v}_e^T]^T$ and pose $\mathbf{q}$ are calculated using the sensed joint-rates as $\mathbf{t}_e = \mathbf{J}\dot{\boldsymbol{\theta}}$, and forward-kinematics, respectively.
- Corrective force due to Twist error: $\mathbf{f}_t = \mathbf{K}_v \Delta \mathbf{t}_e$
- Corrective force due to Pose error: $\mathbf{f}_q = \mathbf{K}_p \Delta \mathbf{q}$
- The corrective torque due to Pose and Twist error $\tau_{qv} = \mathbf{J}^T(\mathbf{f}_t + \mathbf{f}_q) = \mathbf{J}^T \mathbf{f}_{tq}$
- $\tau_g = [\tau_1 \;\; \tau_2 \;\; \cdots \;\; \tau_n]^T$ is the gravity compensation torque vector
- $\tau_E =$ Torque generated due to the external interaction
- $\tau_v =$ Natural joint damping
- $\tau_{max} = \mathbf{J}^T \mathbf{w}_{desired}$ is the maximum torque limit set for the robot's safety

So, the input joint torque tau through the controller drives the robot's joint actuators. As the robot moves, the end effector twist and pose are calculated using the sensed joint-rates as this.

$$t_e = J\dot{\Theta}$$

This is how we calculate and forward kinematics, respectively. The corrective force due to the twist error is calculated like this.

$$f_t = K_v \Delta t_e$$

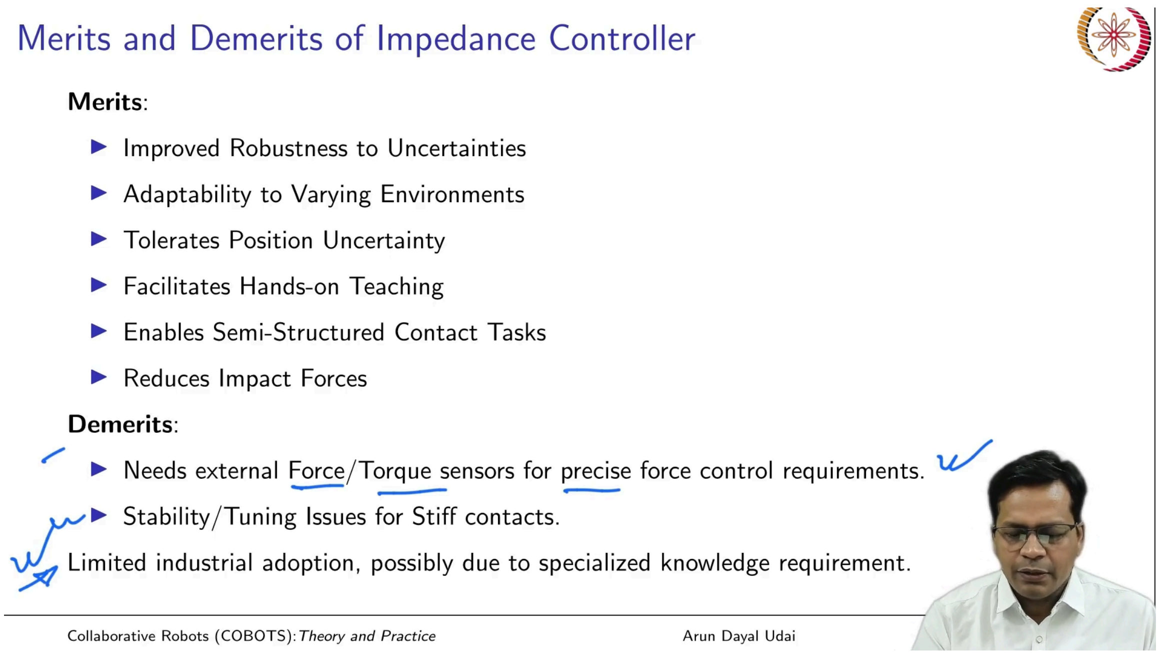
Similarly, the corrective force due to the pose error is calculated here.

$$f_q = K_p \Delta q$$

The corrective torque due to the pose and twist error combined is Jacobian transpose times this plus this. So, this gives you the torque.

$$\tau_{qv} = J^T(f_t + f_q) = J^T f_{tq}$$

Tau g is the gravity compensation torque. Tau E is the torque generated due to external interaction. Tau v is the natural damping, joint damping. It can also be synthetically imposed to make the system stable at times. Tau max is something that needs to be imposed to combine all the torques. This should not generate a torque that is not acceptable to the actuators. So, in that case, this tau max is also imposed. This is nothing but a saturation that you need to put. It is the maximum torque limit set for the robot's safety. So, this is how it works.



Merits and Demerits of Impedance Controller

Merits:

- Improved Robustness to Uncertainties
- Adaptability to Varying Environments
- Tolerates Position Uncertainty
- Facilitates Hands-on Teaching
- Enables Semi-Structured Contact Tasks
- Reduces Impact Forces

Demerits:

- Needs external Force/Torque sensors for precise force control requirements.
- Stability/Tuning Issues for Stiff contacts.

Limited industrial adoption, possibly due to specialized knowledge requirement.

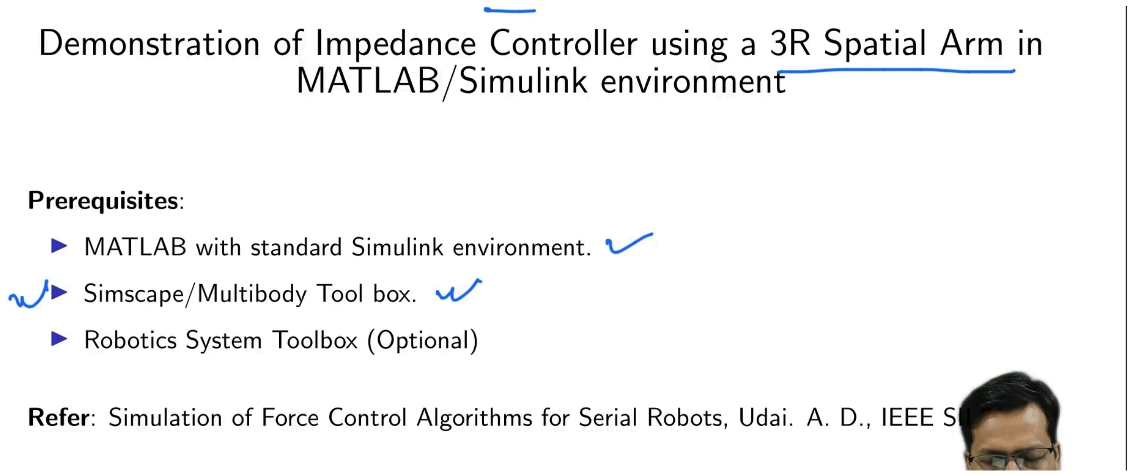
Collaborative Robots (COBOTS): *Theory and Practice* Arun Dayal Udai

So, the merits and demerits of such a controller are: first, let us discuss the merits. So, it has improved robustness to uncertainties. So, in any uncertain environment, it can easily adapt. It's not like an admittance or a hybrid controller. It adapts to varying environments. It tolerates position uncertainty also. If it hits a table, because it is a compliance is set at the controller level. So, in that case, it behaves in a compliant manner. It won't penetrate into the table due to any position uncertainty. Whatever interaction it is doing, it will be compliant. It facilitates hands-on teaching.

If you see here, what I am doing here, I am able to drag this robot to different points. I can teach here. I have taken it to different locations. I will just fast forward it. So, after teaching, you can quickly release and let the robot repeat those points. So, teach a repeat kind of programming. It becomes very handy and you can take the robot to any location and do teach repeat. So, it supports that.

It enables semi-structured contact tasks also. Semi-structured means you do not know which direction you have to be compliant. So, it is not properly structured, as in the case of a hybrid controller, it needs a strictly structured environment where you precisely know which direction you need compliance. It reduces any impact forces. So, these are the merits of this.

But there are a few demerits also. Apart from it needs little more complex controller. It also needs more computational power at the controller end. These are a few demerits here. It needs an external force sensor for precise force control requirements. If at all, there is a requirement, which is very rare in the industry, but in order to do so, you will require a precise force sensor. Stability and tuning issues are there, especially for any stiff contacts, in which it can also tend to bounce back. So, you have to balance between the positional error gain and the velocity error gain, that is, the Kp and Kv factors that I have shown you. That actually creates the impedance. It is a combination of impedance as well as the positional error gain, which is stiffness. Very limited industrial adaptation exists because of specialised knowledge requirements. Not because of its inherent demerits, but yes, in industry, the people who program them need to be trained, and Cobots still need to be adapted by the industry. So, that is still lacking. But it will definitely have potential, and it will come up and be widely adopted in the very near future.



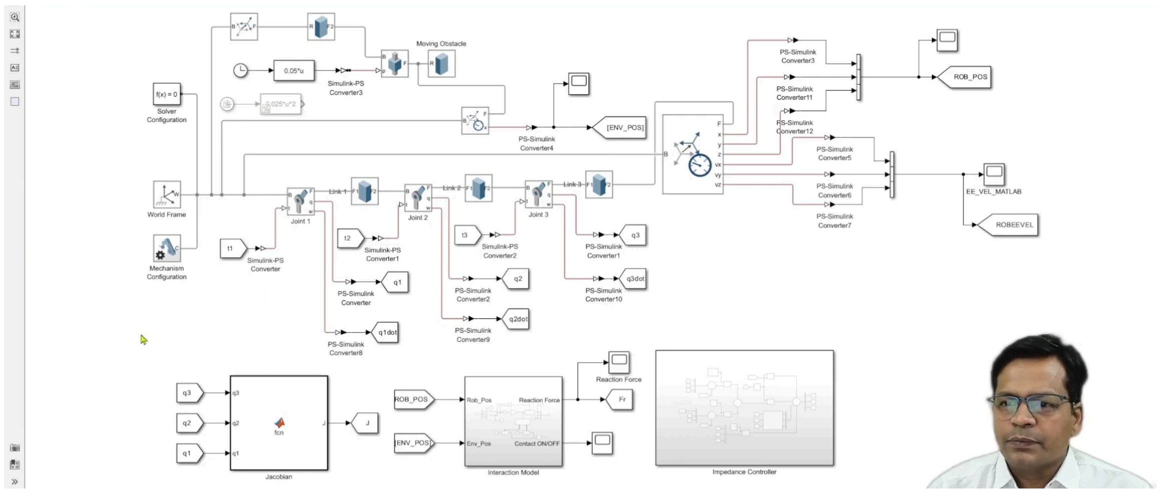
Demonstration of Impedance Controller using a 3R Spatial Arm in MATLAB/Simulink environment

Prerequisites:

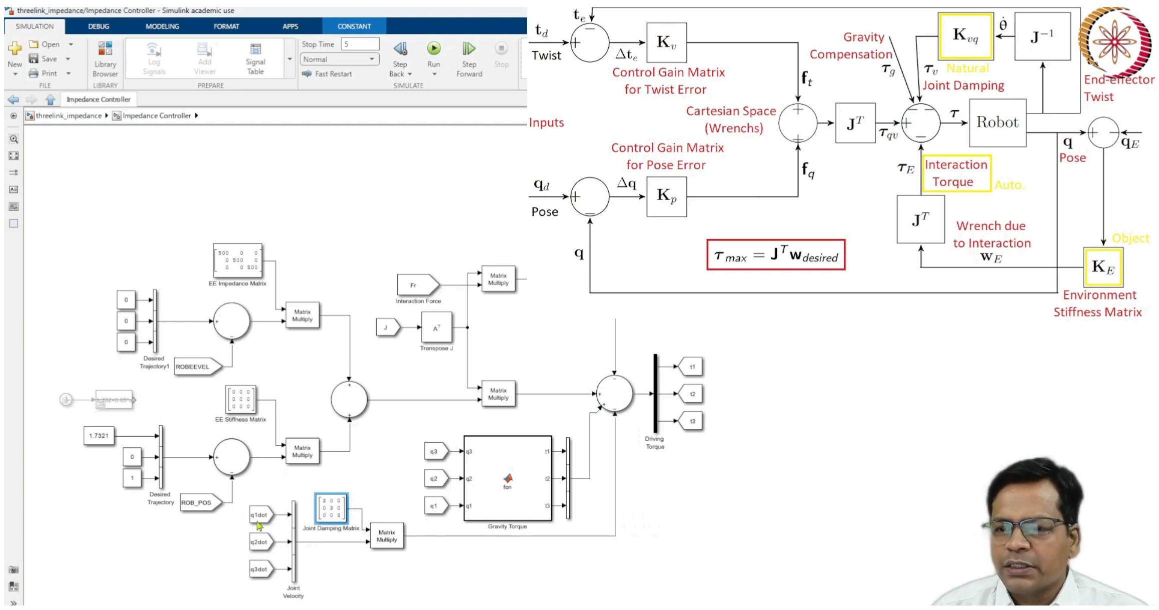
- MATLAB with standard Simulink environment.
- Simscape/Multibody Tool box.
- Robotics System Toolbox (Optional)

Refer: Simulation of Force Control Algorithms for Serial Robots, Udai. A. D., IEEE S

So, let us have a demonstration on impedance control using the 3R Spatial Arm, the way we did it. Using the same arm, I am going to do it. MATLAB with the standard Simulink environment will be required. I am using the Simscape Multibody toolbox only. So, the Robotic System Toolbox is optional. One can use it as well. So, let us go to that.



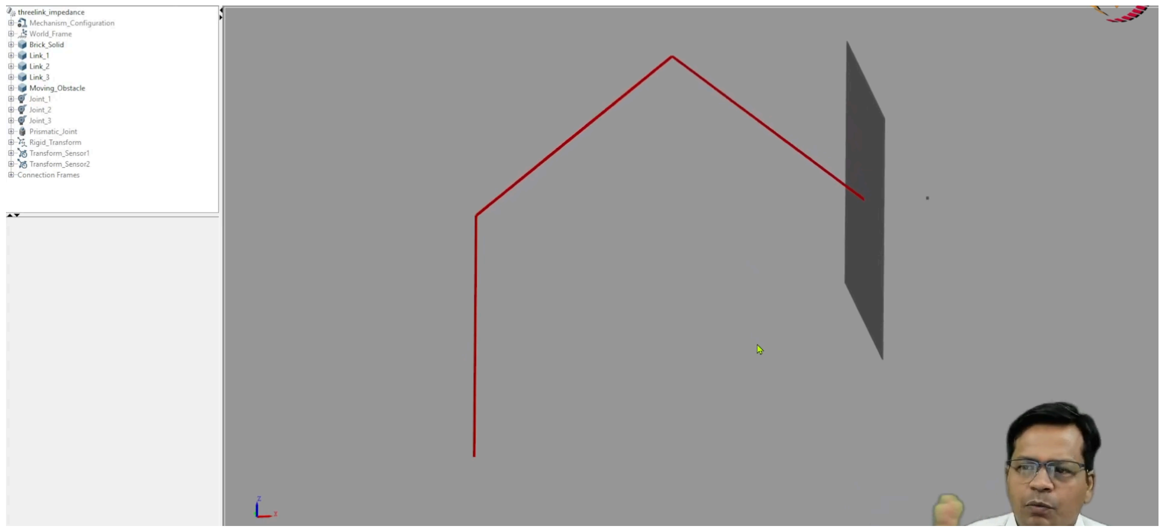
This is the controller that you can see here. The whole robot environment is the same as this robot, as we have used it in the stiffness controller. We are calculating the robot position and the robot velocity. The environment position as well as the robot position are both used to do this interaction model here. The interaction model is again the same. The Jacobian I am calculating manually will be used by the impedance controller. As you can see here. Interaction forces are again used by the impedance controller here.



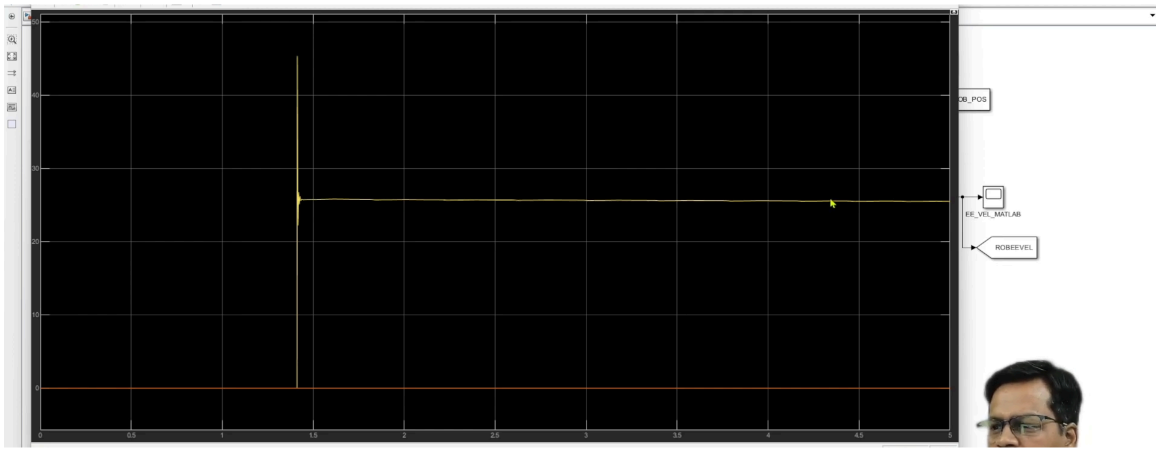
So, let us get into the impedance controller directly. I'll just open up the picture here so that you can refer to it quickly. So, here is the trajectory the desired trajectory input. This is the velocity input. So, I am taking the feedback also. Robot velocity taking the difference, multiplying it with the end-effector impedance matrix. So, I have kept it as 500, 500, and 500 along each direction Newton per meter per second. So, multiplied by that, it will generate forces in Newton.

Again, this is the desired position: I want my robot to be stationary, and I have kept zero velocity input here. So, the desired position is kept as the current end effector position. I am taking the feedback from the robot, that is, the robot's position. I am subtracting that, calculating the error in position, multiplied by stiffness. I have kept zero stiffness here. I want to see the impedance effect only. So these two are there. Both of them are summed here and sent to the robot. The force input combined is sent here. When multiplied by the Jacobian transpose, it generates the torque corresponding to the position and velocity error. So, that is standard, which is quite obvious in the control diagram also.

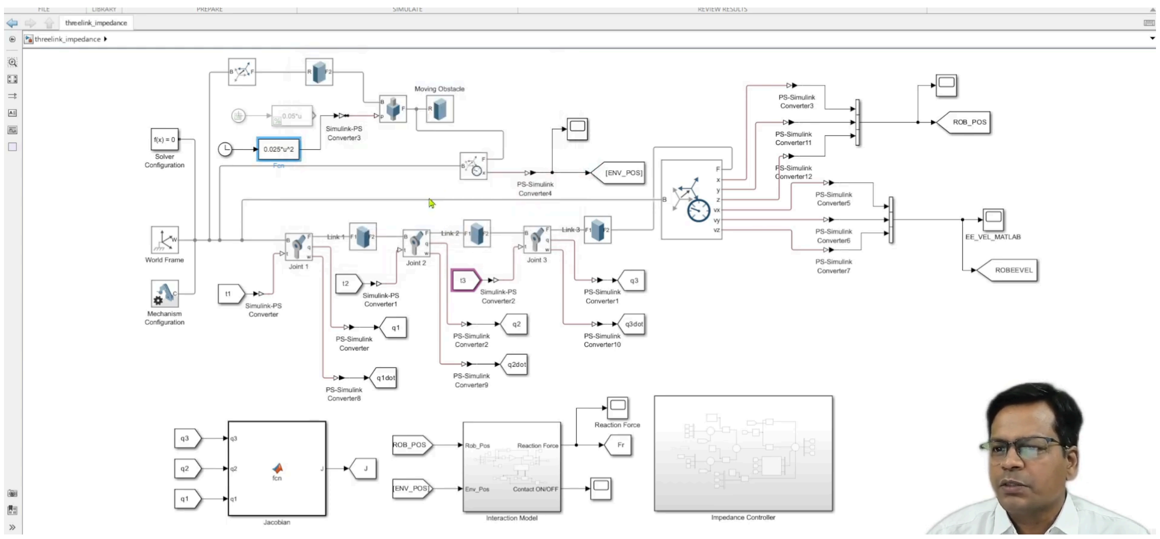
Environment torque is calculated from the interaction force multiplied by the Jacobian transpose. That also goes to the robot here through this line. Gravity torque is calculated here. That is going here, and joint damping, I have kept a very small value. 3, 3, and 3. 3 Newton meter per radians per second. That is N. Joint rate multiplied by the joint damping matrix actually gives you the joint torques once again, and that is also here. So, all the torque output goes to the robot. So, this is how the impedance controller will work. So let us now see the effect of this.



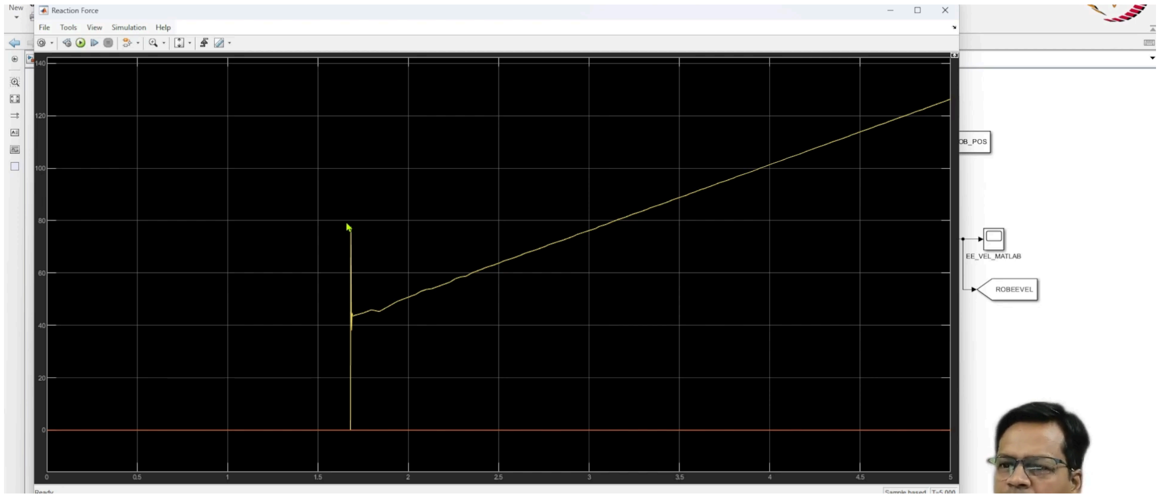
So, I will just run it, and you see, this wall is moving with a constant velocity towards the robot, and it is pressing the robot. If it is a constant velocity. So, that is obvious here. So, this is at a constant velocity, you see, at 0.05 meters per second. It is moving towards the robot, and it has interacted.



So, let me check the plot here. So, you see it is a straight line because the velocity is constant, and I have not set any stiffness along the interaction direction. I have not set any stiffness anywhere. So, only because of damping because which is proportional to the velocity. Velocity is constant; force has to be constant. So, this is generating a constant force interaction force and the robot is continuously moving along with the wall.



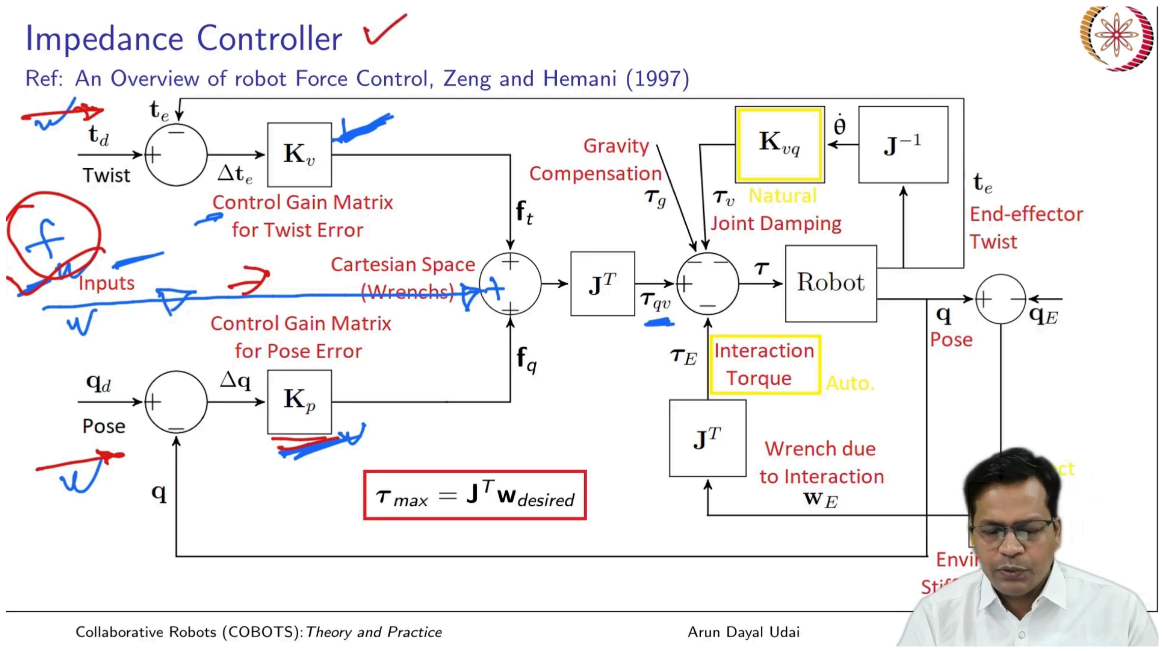
So now, let me just change the scenario, let me just accelerate the wall. So, that velocity keeps on increasing with time. So, how should it behave in this case? So I'll just put it here, uncomment this one. All these models I have shared with you in the resources, you can always use them, modify them, test them, and learn from them. So, this is the constant acceleration I have put. 0.025 t square, that is, the time square. So, velocity is equal to ut plus half a t square; a is 0.025 t square. So that is what is visible here. So, it is moving with an acceleration. Again, I will run it this time. This is accelerating. Gradually, the velocity is increased, and it will press a little more. So, this is the new scenario, and see how it behaves.



So, immediately upon contact, a huge amount of surge is there due to collusion and then the force rises gradually with the wall because the velocity is increasing as it approaches the robot, and it compresses the robot. So, the higher the velocity higher the force it generates. So, this is what is expected out of impedance.

So, you see, this is the whole case. You have different scenarios that you can test. You can remove the gravity compensation line, and you will find the robot falls freely. You can remove the joint damping, and you can see how it behaves. You can put some stiffness. So, it has an impedance controller. You see, it has a subset of stiffness controller also.

If you remove the top loop, make the Kv is equal to 0, which is here. So, all the Kv, if it is 0 and put some stiffness, it behaves like a stiffness controller, exactly the stiffness controller. So, this is how it is. But yes, this is the MATLAB simulation. Hope you are done with this.



Impedance Controller ✓

Ref: An Overview of robot Force Control, Zeng and Hemani (1997)

$$\boxed{\tau_{max} = \mathbf{J}^T \mathbf{w}_{desired}}$$

So, let me come back to the impedance controller here. So, this is my impedance controller. So, you have seen the effect of pose error that which is exactly the stiffness controller, which is here, and apart from that, you have impedance, which is here. You can have both. So, combined, it makes things stable, makes everything stable. So, it has no more tendency to bounce and create any jitter while it is contacting.

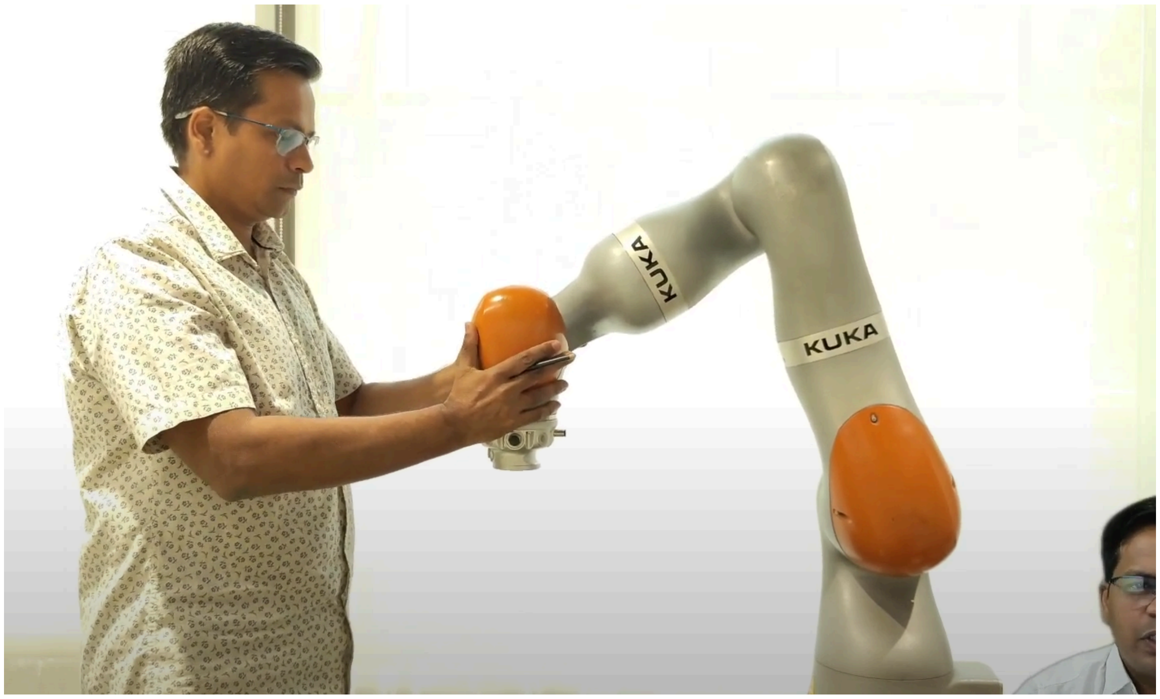
Apart from that, there are a few modified versions of this also. You can also directly put the force inputs into this loop. If you do that, you can set stiffnesses all equal to 0. You can also set all the impedances equal to 0, and you can directly put the desired force along any direction. So, that will directly generate the joint torque corresponding to the desired forces at the end effector, and it will run the robot to create the desired forces. So, force input can be directly given.

Now, you can set a certain stiffness along with the force. So, the robot has stiffness along a few directions and remaining directions, it will have forces. So, that is quite easily realizable in this kind of controller, where you can have force input, you can have velocity input, you can have pose input, also, you can have stiffness along a few directions, and you can have force along a few directions. So, accordingly, depending on the requirement, you can have everything else.

So, let us say I want to write on a board. So, I want to maintain a constant reaction force against the wall, against the whiteboard or a blackboard. But I can move along the plane of the surface. So, along the plane, I can have just the stiffness matrix or the impedance matrix. But yes, perpendicular to the surface, I need to apply the force input. So, that is what is visible through one of the experiments.

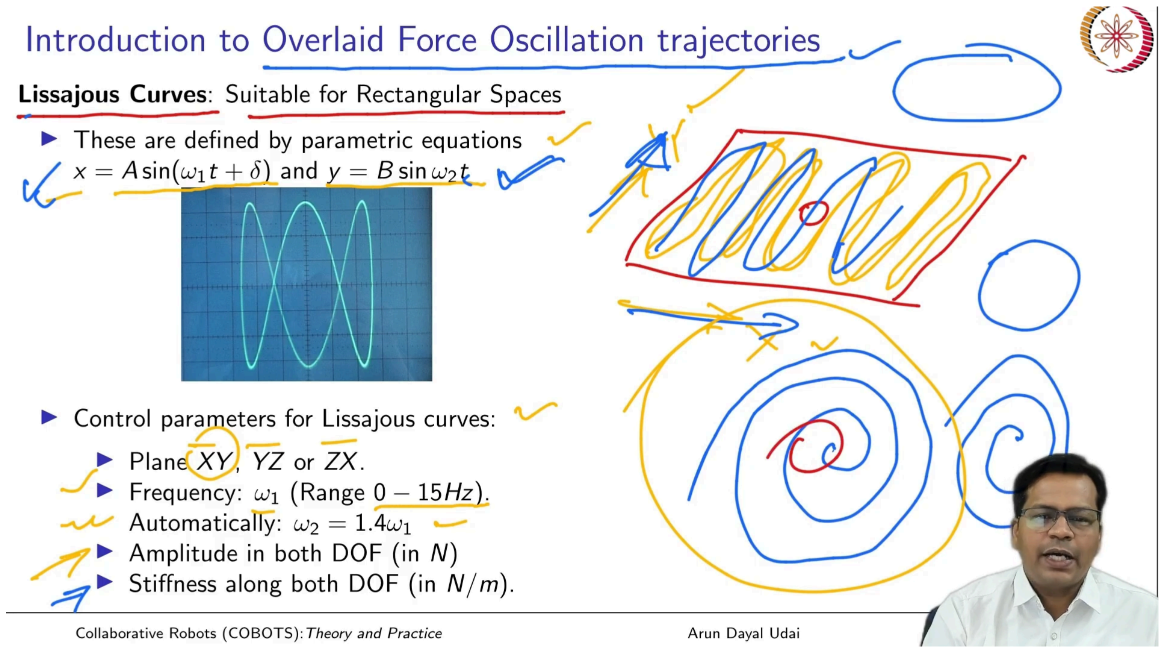
What I am trying to do here is the robot lowers at a constant velocity until it establishes contact. Upon establishing contact, it maintains a constant reaction force against the surface. While it also moves along one of the directions. You see, this time the matrix should be, and it is stiff along the orthogonal direction also. So, it is stiff along one direction, it is moving along one direction, and it is creating a normal force along one direction. So, all three directions have inputs. So, in this way, it helps a lot, you see.

Now, you come back to your thing. So, for this application, there was a force input which was perpendicular to the surface force input, along the vertical direction, and you have velocity inputs so that it can move on the table in a straight line. So, that was a constant velocity input which was done, and then it was said to have some stiffness where it is not supposed to move or maintain any forces. So, there was some stiffness set along the remaining orthogonal direction. So, all three inputs can be used for applications like this.



Normally, an impedance controller is like this, you see. In this case, I have set an Impedance and stiffness along all directions. So, one of the directions if I set the stiffness to be very low in that direction, I can move the robot. But yes, if impedance is there along the same direction, I will feel some resistance while I am moving with a certain velocity. So that it does not become very free. So it damps while it moves. So, this time, vertical direction very small amount of damping is kept. Stiffness is set to 0, and now, in this direction, I have kept a high stiffness value. So, I cannot move it. So, it behaves like a spring in the remaining direction. Low stiffness, I am able to move it, or 0 stiffness, I am able to move it. It would not come back. If it is like a spring, it will come back, but if it is a damping, it will allow you to move. With a small velocity, you keep on moving; it will generate a very small amount of force.

So, hope you got the essence of this kind of impedance controller. It is very potentially useful. So, these are the equations that I have used for this controller. This is the merit I have discussed, and these are the demerits.



Introduction to Overlaid Force Oscillation trajectories

Lissajous Curves: Suitable for Rectangular Spaces

- These are defined by parametric equations $x = A\sin(\omega_1 t + \delta)$ and $y = B\sin\omega_2 t$
- Control parameters for Lissajous curves:
  - Plane XY, YZ or ZX.
  - Frequency: ω_1 (Range $0 - 15Hz$).
  - Automatically: $\omega_2 = 1.4\omega_1$
  - Amplitude in both DOF (in N)
  - Stiffness along both DOF (in N/m).

Collaborative Robots (COBOTS): *Theory and Practice* Arun Dayal Udai

So, let us go to a case study here. So, yes, this is a small introduction I want to give you. You already know what a Lissajous curve is. It is suitable for Rectangular Workspaces. Let us say this robot has to generate a trajectory so that it searches for the hole on a table. So, let us say this is a table that is square, and there is a hole. So, the robot has to make a path which is something like this. So, these kinds of paths can be easily generated using these predefined parameters. So, it is an equation for Lissajous curves. XY being the plane of this table, so, x is equal to Asin omega 1 t plus delta, and y is equal to Bsin omega 2 plus delta.

$$x = A\sin(\omega_1 t + \delta) \text{ and } y = B\sin\omega_2 t$$

So, omega 1 and omega 2 are the frequencies along these two orthogonal directions, x and y. Control parameters for the Lissajous curve, the plane can be the XY plane, it can be YZ or the ZX plane. So, for this horizontal plane, it can be XY. Frequency by default, this is for the KUKA iiwa robot.

In its Sunrise operating system, it has by default frequency omega 1. It takes from 0 to 15 hertz, and automatically omega 2 is set as 1.4 times of omega 1. So, it creates an amplitude. So, see in the impedance controller, if you look carefully, so, if you put a centre force oscillation and keeping the stiffness along those directions. So, if you generate force, it will tend to deviate from the position. So, as soon as the position is deviated, what will happen? Desired because you want to be in a place. So, what will happen? The difference will be multiplied by the stiffness matrix, and it will tend to come back.

So, force oscillation actually creates a displacement at the end effector. So, this is a very nice application when you can have force oscillation along two orthogonal directions and create displacements. So, in that case, any external force will tend to stop the robot from going in that direction. Because they are not position oscillations, they are force oscillations. So, it can be stopped any external obstacle will tend to stop it. So, this is how it will do oscillations on this plane. As soon as the pellet if you want to inserted that pellet into that hole, it will have a tendency to get stuck in the hole, and as soon as it gets stuck, it will insert the pellet. So, this is what is the advantage of having a Lissajous curve. Lissajous is good for rectangular surfaces. So, a constant force, the normal force, can also be kept.

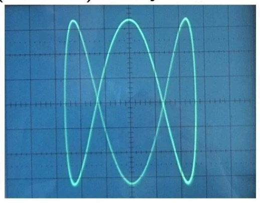
Apart from XY force oscillation and keeping stiffness in those two directions, in the vertical direction, put the stiffness to 0, and maybe you can have some damping, but yes, you can have a vertical constant reaction force applied across. So, that as soon as it finds the hole, it can get in. This is an overlaid orthogonal force oscillation trajectory that can be created. The position trajectory is created out of force oscillations. Stiffness can be set along both directions.

Similarly, you can also create spirals. The only equation will be different. The xy equation will be different, and you can create spirals- converging and diverging. You keep on increasing the amplitude. What will happen? So, x is equal to r cos theta, and y is equal to r sin theta. So, it creates an ellipse, it can create a circle, and it can create spirals with varying time. If you keep on increasing the radius, So, it creates something like this. So, you can have multiple equations and create. So, this is good for let us say a circular table in which if you have a hole here. In order to find that, you can use circular search paths.

Introduction to Overlaid Force Oscillation trajectories

Lissajous Curves: Suitable for Rectangular Spaces

- These are defined by parametric equations $x = A\sin(\omega_1 t + \delta)$ and $y = B\sin\omega_2 t$



- Control parameters for Lissajous curves:
  - Plane XY, YZ or ZX.
  - Frequency: ω_1 (Range $0 - 15Hz$). Automatically: $\omega_2 = 1.4\omega_1$
  - Amplitude in both DOF (in N)
  - Stiffness along both DOF (in N/m).

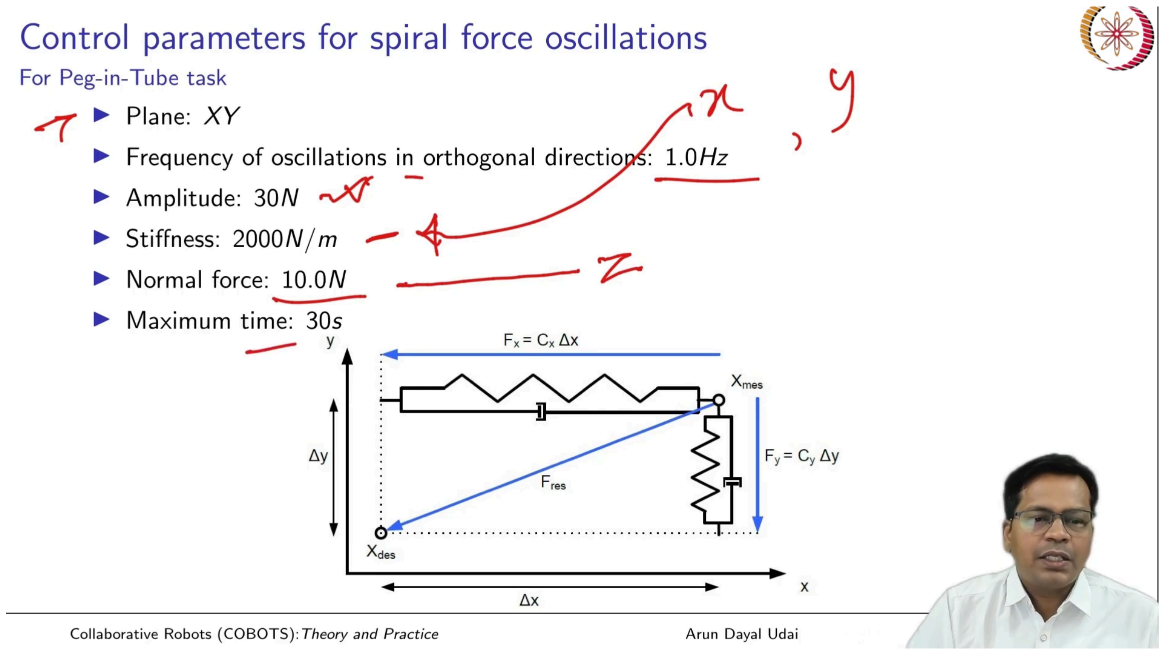
Spiral Path: Suitable for symmetric search

- The controller overlays two sinusoidal force oscillations phase shifted by $\pi/2$.
- The amplitudes of the oscillations rise constantly up to the defined value and then return to zero.
- This results in a spiral pattern diverges to the maximum amplitude then converges back to zero.
- The Cartesian extent of the spiral depends on the values defined for stiffness and amplitude as well as any obstacles present.

NOTE: The values mentioned here is for KUKA LBR iiwa COBOT Arm

So, the spiral path which is suitable for a symmetric search path. The controller overlays two sinusoidal force oscillations phase phase-shifted by pi by 2. Phase shifted means one side, we will have a sine profile other side, you will have a cosine oscillation. The amplitude of oscillation rises constantly up to the defined value and then returns to 0. So, what happens? It diverges, and finally it comes back. So, that is how you can do it. So, there is a higher probability that you will find the hole here.

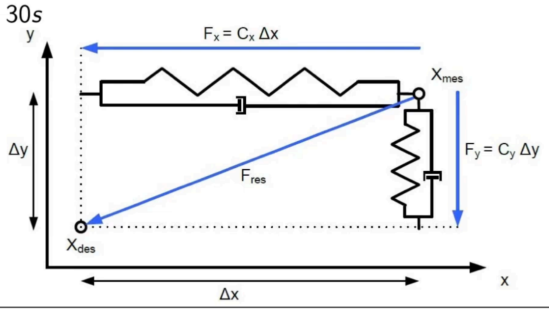
The Cartesian extends of the spiral depends on the values defined for stiffness. Just like this, you can have stiffness along two directions and apply force oscillation. So, the displacement will depend on the stiffness that you have set along both the direction and amplitude, as well as any obstacles if it is present. So, this is how you can create a spiral search path. The values mentioned here are for the KUKA LBR iiwa COBOT Arm.



Control parameters for spiral force oscillations

For Peg-in-Tube task

- Plane: XY
- Frequency of oscillations in orthogonal directions: $1.0Hz$ x, y
- Amplitude: $30N$
- Stiffness: $2000N/m$
- Normal force: $10.0N$ z
- Maximum time: $30s$

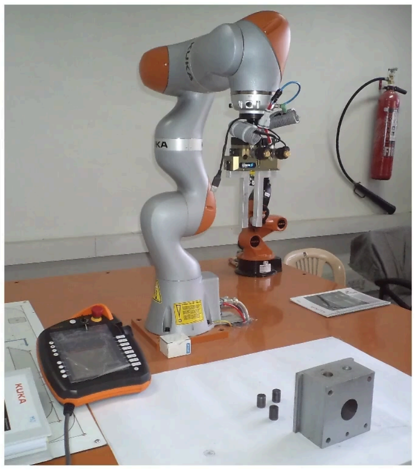


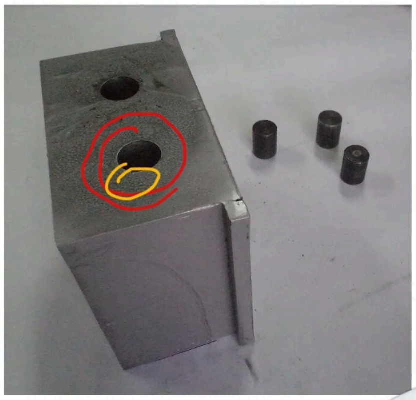
So, this is the way I was discussing now. So, the plane was XY. The frequency of oscillation along the orthogonal direction was kept at 1.0 Hz. Amplitude as x. 30 Newton. So, the amplitude of sine and cosine was 30 Newton. Stiffness along both directions was set as 2000 Newton per meter, and the normal force. So, along the XY plane, I have what? I have set the stiffness. Along the normal direction, along z, I have kept stiffness as 0, and then it will have a constant normal reaction force of 10 Newtons.

So, the maximum total time for the Lissajous or spiral curve, I have set it as 30 seconds. Accordingly, I have done all the calculations. So, this is how the trajectory can be generated and oscillations can be done.

Case Study: Blind Assembly Task (Peg-in-Hole operation)

Assembly Setup and Peg-in-Hole Assembly

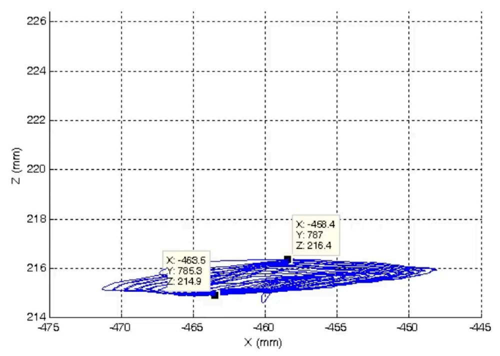


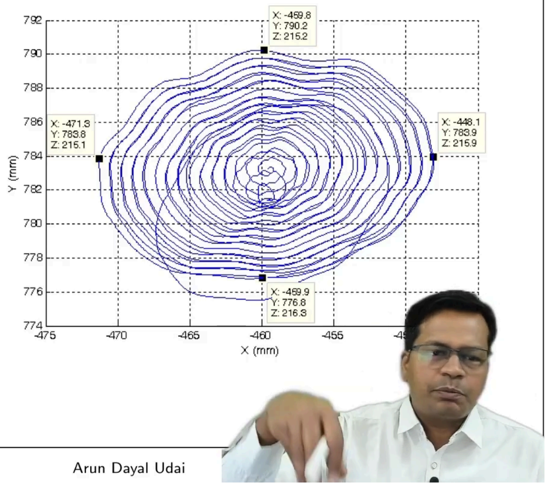


So, this is the whole setup. It is a blind assembly task when you do not know the position of the hole. The robot goes quite near to the hole and does a spiral or Lissajous path. In this case, it is a spiral because you have to search over here. So, it is symmetrical once you reach land somewhere, maintain a constant normal reaction force and start doing the spiral search path, diverging and converging. So, this is how I will be doing it with the parameters that I have shown here.

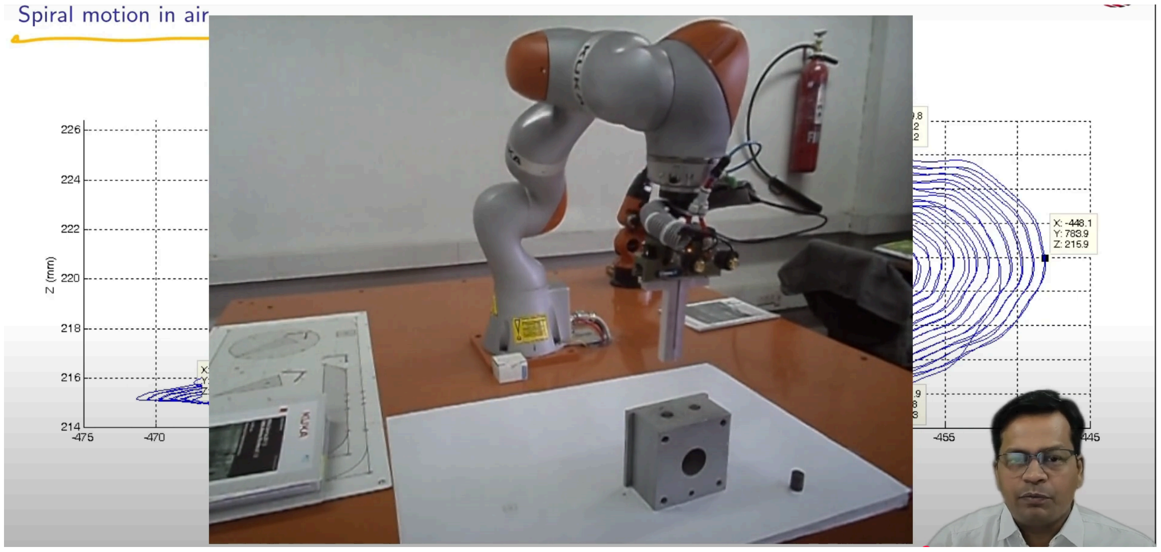
Plots for force oscillations based spiral search

Spiral motion in air



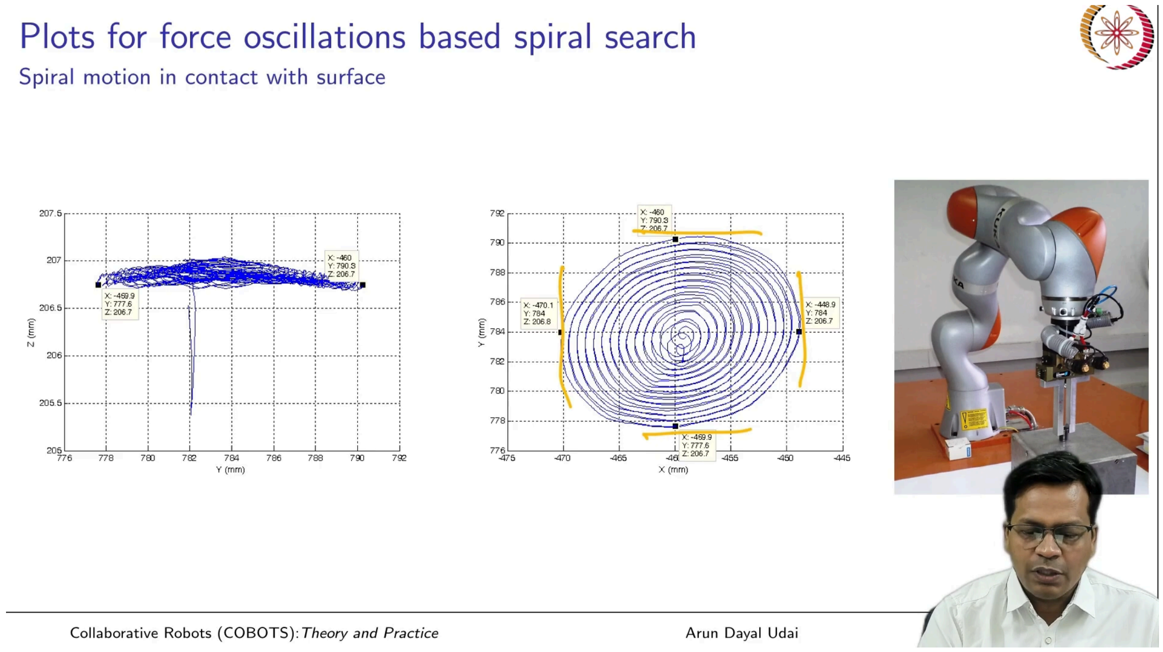


So, let us observe closely how the motion could be when it is in the air. In the air, there is no resistance. So, it tends to create a spiral path of some size.



Let me just show you how it behaves in the air. So, it does a mark here. I will not establish any contact. I will just go on top of the surface, and this is why I have not lowered. That means I have not put the vertical motion, and it is a diverging and converging spiral. This is the visible displacement. So, this is a force spiral. If you hold it, it tends to stop because of external forces that you are applying at the end effector. So, this is a spiral search path.

So, this plot is shown here. It tends to diverge and finally converge. It is not accurate, though. In this space, it is not very accurately circled as it is supposed to be. But yes, it has created some diameter along x and along y while it is in the air.

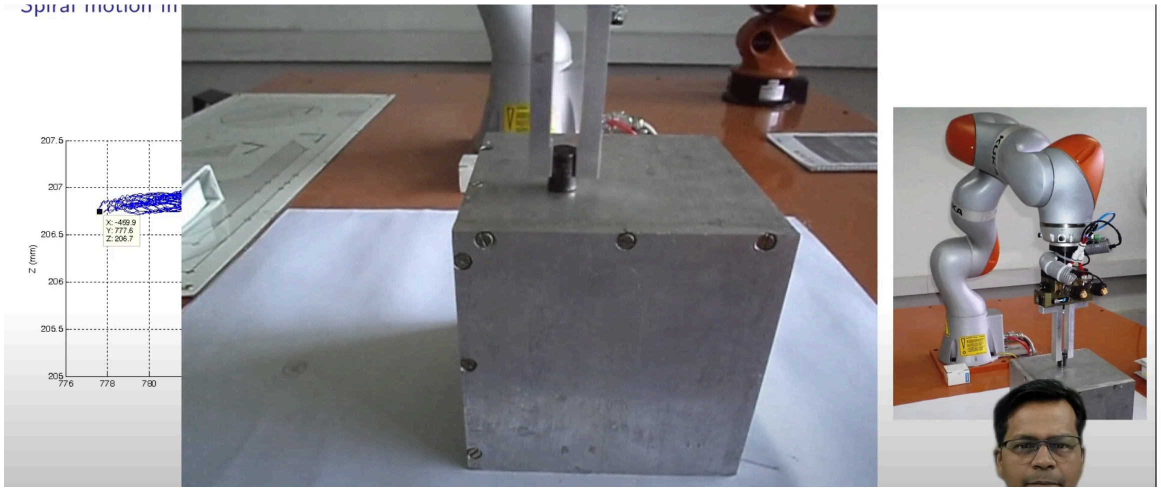


Plots for force oscillations based spiral search

Spiral motion in contact with surface

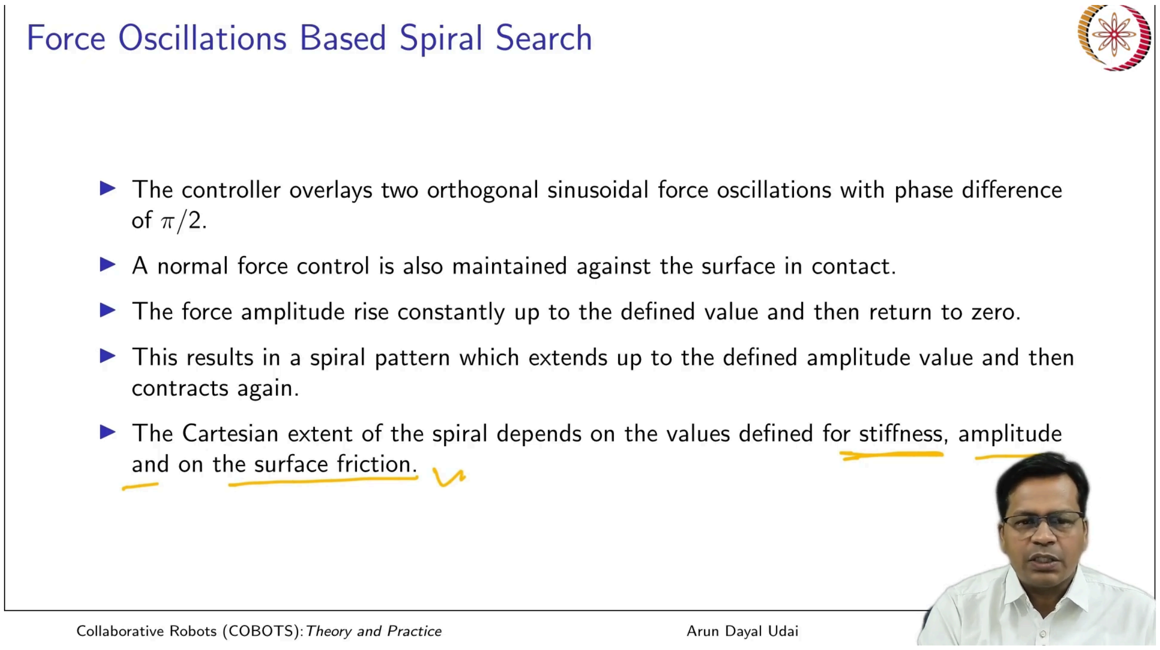
Collaborative Robots (COBOTS): *Theory and Practice* Arun Dayal Udai

So, when it establishes contact with the surface, how do you find it?



So, this time on the surface, you see again it will lower with a certain velocity, establish contact, maintain a constant reaction force, and it is doing the spiral diverging and converging search.

So, when it finishes, you see this time it is more uniform because it is not free in the air. It is rubbing against the surface, which is causing this diameter to reduce because of external forces. So, this is what is observed.



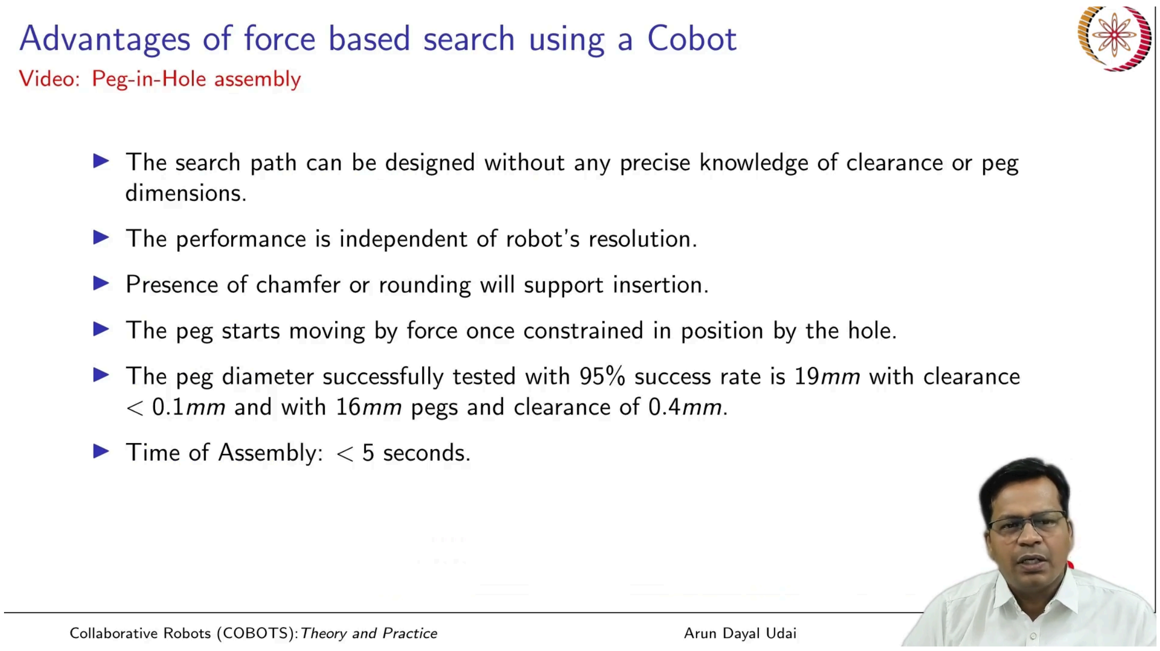
Force Oscillations Based Spiral Search

- The controller overlays two orthogonal sinusoidal force oscillations with phase difference of $\pi/2$.
- A normal force control is also maintained against the surface in contact.
- The force amplitude rise constantly up to the defined value and then return to zero.
- This results in a spiral pattern which extends up to the defined amplitude value and then contracts again.
- The Cartesian extent of the spiral depends on the values defined for stiffness, amplitude and on the surface friction.

Collaborative Robots (COBOTS): *Theory and Practice* Arun Dayal Udai

So, when I go further, the controller overlays two orthogonal sinusoidal force oscillations with a phase difference of pi by 2. A normal force is also maintained against the surface in contact.

Force amplitude rises constantly up to a defined value and then returns to zero. This results in a spiral pattern which extends up to the defined amplitude value and then contracts again. The Cartesian extents of the spiral depend on the values defined in the stiffness. So, how much will be the maximum displacement? That depends on the amplitude of the force that you have put because there is stiffness. So, force oscillations will create displacements. So, amplitude and surface friction also. If it is on the surface, there is some friction the amplitude may reduce. The displacement amplitude will reduce.



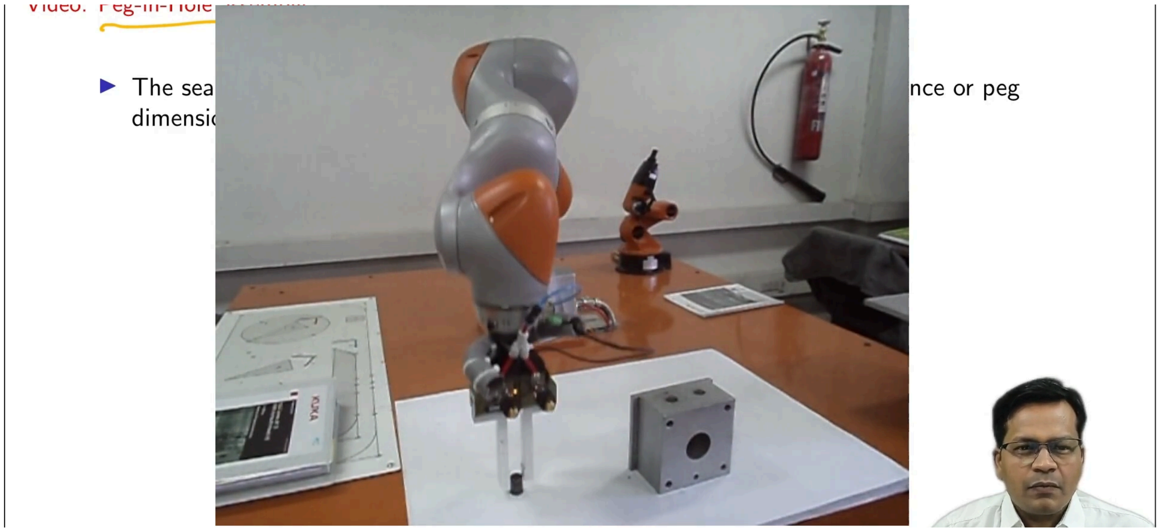
Advantages of force based search using a Cobot

Video: Peg-in-Hole assembly

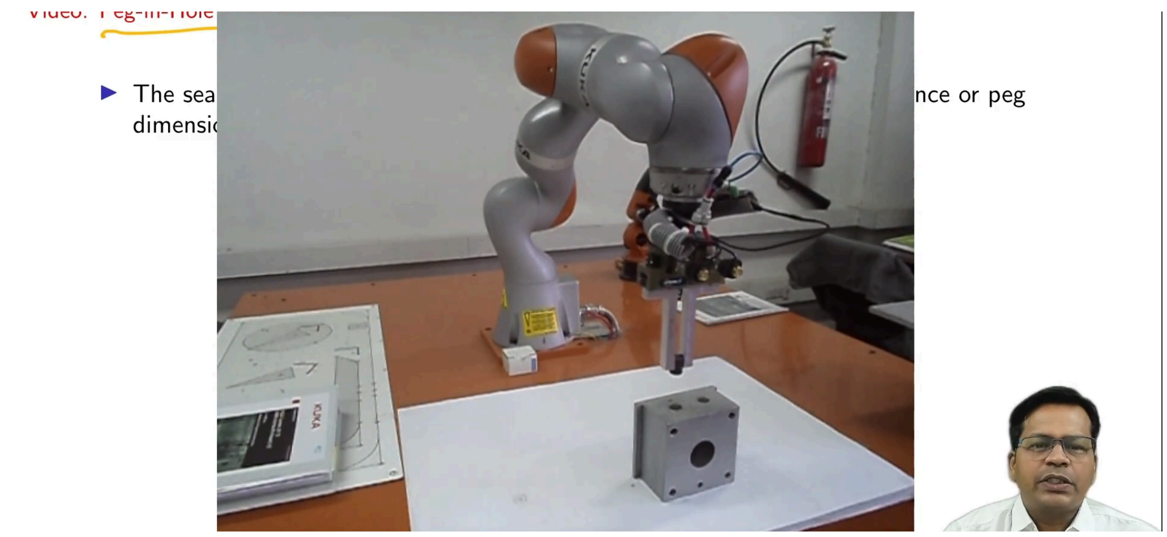
- The search path can be designed without any precise knowledge of clearance or peg dimensions.
- The performance is independent of robot's resolution.
- Presence of chamfer or rounding will support insertion.
- The peg starts moving by force once constrained in position by the hole.
- The peg diameter successfully tested with 95% success rate is 19mm with clearance $< 0.1mm$ and with 16mm pegs and clearance of 0.4mm.
- Time of Assembly: < 5 seconds.

Collaborative Robots (COBOTS): *Theory and Practice* Arun Dayal Udai

So, this is the case of a vertical peg in a hole. So, I will just show what I have done finally. So, this is what. So, now this time it will pick up the pellet.



Go very near to the surface. So, you see, such a path is not dependent on the positional accuracy of the robot.


Video: Peg-in-Hole

▶ The sea nce or peg dimensi

It is going to create a force oscillation. It runs purely on the interactions. The way we insert a key in a hole, how do we do it? Like a blind assembly, even when it is dark, you can do that. So, you see, robot lowers. Picks up the pellet, it goes vertically above the hole. It is not governed by the precision of the robot. It lowers, establishes the contact of 10 Newton, which was set, does diverging and converging spirals, and finally finds the hole and inserts it, goes back. So, this is how it works.

So, the spiral search path can be designed without any precise knowledge of the clearance or the peg dimensions or the location. You just go somewhere near the hole using the vision-assisted system or something like that, or maybe a pre-programmed position. So, it will take up all those uncertainties there.

The performance is independent of the robot's resolution, that is, the resolution, or maybe the repeatability of the robot, or by any chance. You just go there, do force oscillation, it gets stuck, and finally it inserts. So, performance is not dependent on the robot's resolution. Also, the presence of a chamfer or any lack of edges so sharp edges, so chamfer or rounding will support the insertion. The peg starts moving by force once constrained in position by the hole. So, as soon as it interacts with the edges of the hole, it gets constrained and does not allow movement along that direction, and it keeps on moving in the remaining direction. Keeps on doing that, doing that, and finally it completely gets trapped, and finally, the vertical allowance is there because there is a constant normal reaction force from the top, and it gets inserted.

The peg diameter that I have tested successfully with a 95% success rate is 19 mm with a clearance of 0.1 mm, and with 16 mm pegs, a clearance of 0.4 mm, I have tried out. So, the total time for assembly is less than 5 seconds, which is very close to the way we do it. When we insert a key inside a hole, we tend to do it in the same manner. We do not know the precise location of the hole.

How are we doing? We just loiter around, move the key, move something near to that, and keep on pressing. We find the hole, we insert it. We don't know the precise location of the hole, but still, we are successful. Same way, the robot can be made to do such tasks. Compliant assembly task. It may not be just linear oscillations you can create. You can create torsional oscillations also. The whole of the impedance controller is not only for positioning purposes; it is also for orientation stiffness, orientation damping, and torsional stiffness. So, this can be set as well.

[+]
Zusätzliche sinusförmige Kraftschwingung
Additional sine wave force profile
KUKA

This is quite clear through this example, you see (this video is from KUKA's website only, YouTube channel). So, it is doing that torsional motion, setting some torsional stiffness and giving that torsional trajectory. It does that, and you see all of the things are aligned. Watch it once again. Looking from the top. So, currently, you see that everything is not aligned. So, while it is doing that torsional oscillation, things get aligned, and it allows you to move further. A constant force from the top is being applied.

So, these are the potentials of impedance control, which is widely used in modern cobots as well.

With that, I will just end this course and this lecture. In the next lecture, I will do a course conclusion. Be there. Thanks a lot.