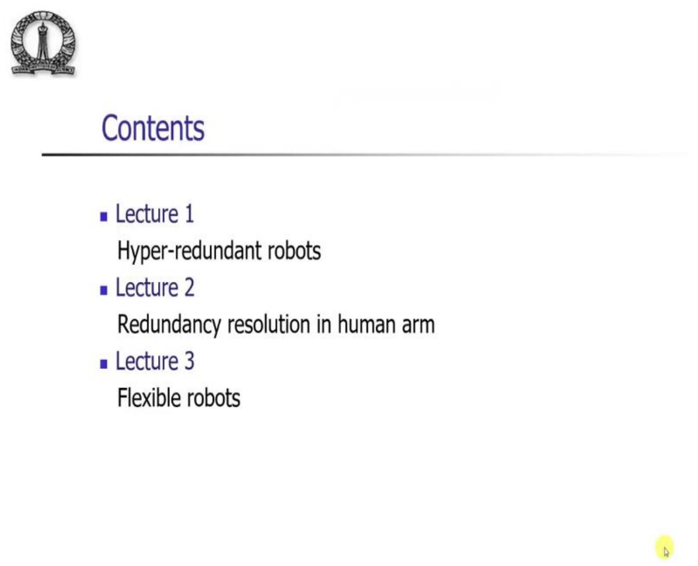
Robotics: Basics and Selected Advanced Concepts

Prof. Ashitava Ghosal
Department of Mechanical Engineering
Indian Institute of Science, Bengaluru

Lecture - 24
Hyper-redundant robots

Welcome to this NPTEL lectures on Robotics Basics and Advanced Concepts. In this week we will look at Redundancy in Robotic manipulators and other mechanical elements and even human arms and how we resolve those redundancies.

(Refer Slide Time: 00:38)

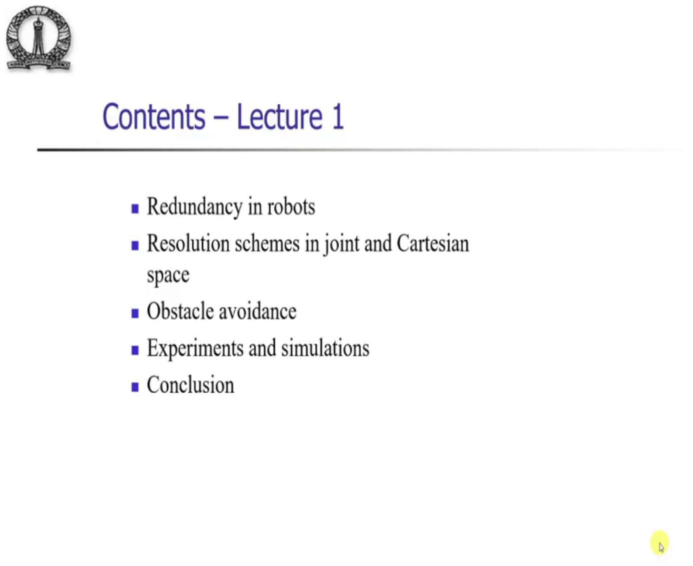


So, the 1st lecture will be on hyper redundant robots, the 2nd lecture will be on redundancy resolution in human arm, and I will show you that the human arm is redundant and what we try to do with the redundancy. In the 3rd lecture we look at flexible robots. The degree of freedom in a flexible robot is theoretically infinite, but many many more than what is required and how we can use this flexibility in flexible robots.

(Refer Slide Time: 01:10)

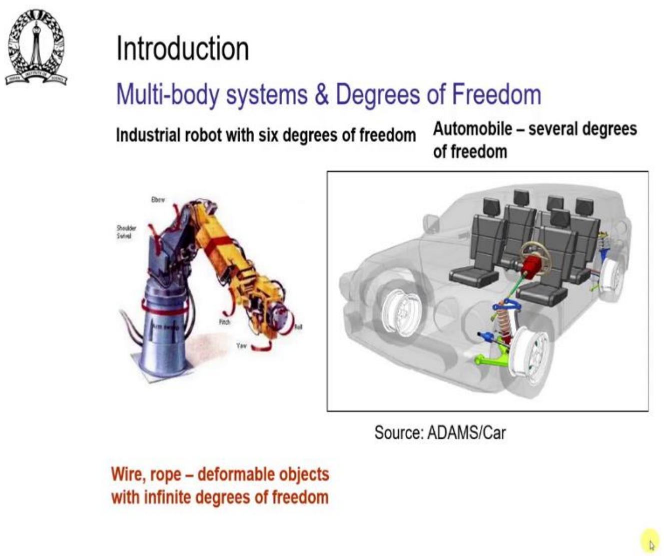
So, quick acknowledgement, this work has been done in the Robotics and Design Lab at IISc and I would like to acknowledge my PhD students Midhun S Menon, Puneet Singh, Ashwin Prabhakaran and also one master's students Arkadeep Narayan Choudhury. Some of the funding for this research was provided by the Robert Bosch Center for Cyber Physical Systems ok.

(Refer Slide Time: 01:36)



So, the contents of the lecture 1 are we look at redundancy in robots. How we can resolve redundancy in joint space and Cartesian space. We will also look at how redundancy can be used for obstacle avoidance and then I will show you some experiments and simulations.

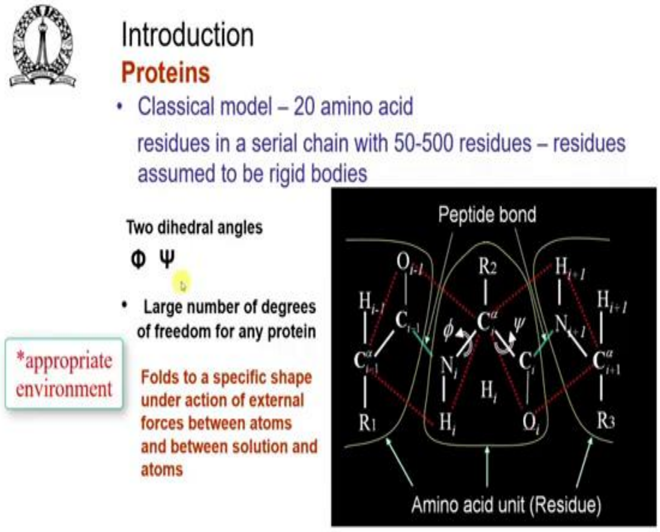
(Refer Slide Time: 01:59)



So, introduction: a multi body system could have many many degrees of freedom. So, for example, an industrial robot can have six degrees of freedom. So, this picture we have seen earlier, it has 1, 2, 3, 4, 5 and 6 joints and it is a six degree of freedom industrial robot. An automobile can have many many more degrees of freedom ok. So, the wheels can have several degrees of freedom, then steering and then the body itself and so on.

So, but still they are finite number of degrees of freedom, if you have a wire a rope or a deformable object, then these have infinite degrees of freedom.

(Refer Slide Time: 02:44)

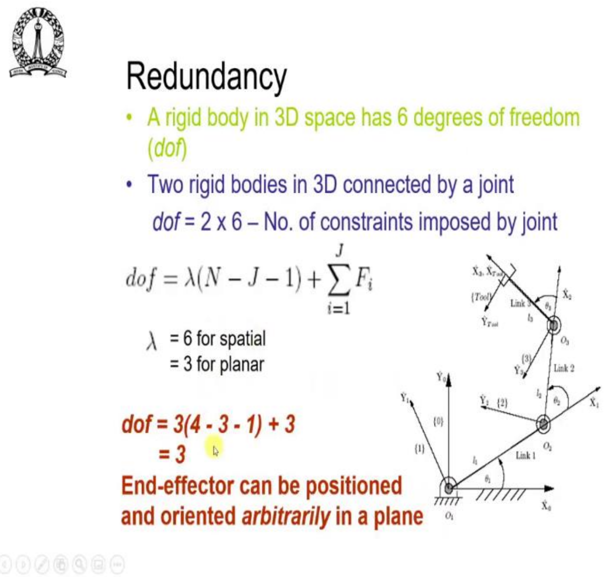


A new reason why people worry a lot about degrees of freedom is this problem of proteins ok. How do proteins fold? So, typically a classical model of a protein has 20 amino acids ok. So, these are residues in a serial chain and this serial chain can contain up to 500 residues ok. In the classical model each residue is assumed to be a rigid body ok.

So, this is one rigid body, this is another rigid body and between these two rigid bodies, there is 2 degrees of freedom. So, it can rotate about two directions. So, these are called ϕ and ψ angles ok. So, as a result a protein chain will have very very large number of degrees of freedom. So, it could have things like up to 100 degrees of freedom to 1000 degrees of freedom.

So, it has been seen that if you put this protein chain in some fluid or some appropriate environment, it will fold into a specific shape ok, and under the action of external forces between the atoms the protein and between the solution and atoms. So, we would like to see what is the motion of this protein chain. So, as you can see it is a highly redundant system.

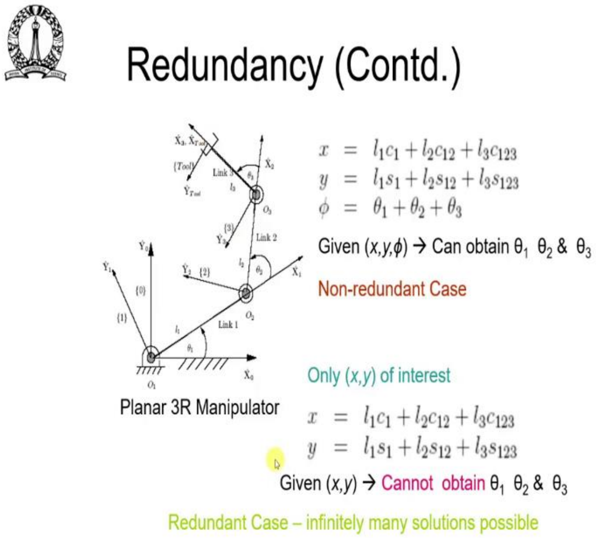
(Refer Slide Time: 04:08)



Let us do a little formally redundancy. What is meant by redundancy? A rigid body in 3D space has 6 degrees of freedom ok, this we have seen several times by now. Two rigid bodies in 3D space when it is connected by a joint as 2×6 degrees of freedom ok. If there are no constraints imposed by the joint ok, if the joints imposes a constraint which all joints do.

Then we can compute the degree of freedom of a mechanical system by this formula, which is $\lambda(N - J - 1)$ plus sum of degrees of freedom at each joint, where λ is 6 for spatial and 3 for planar. So, for example, this planar 3R manipulator has $N = 4$, $J = 3, \lambda = 3$ so $N = 4$, because we need to count the fixed place also ok.

(Refer Slide Time: 05:12)



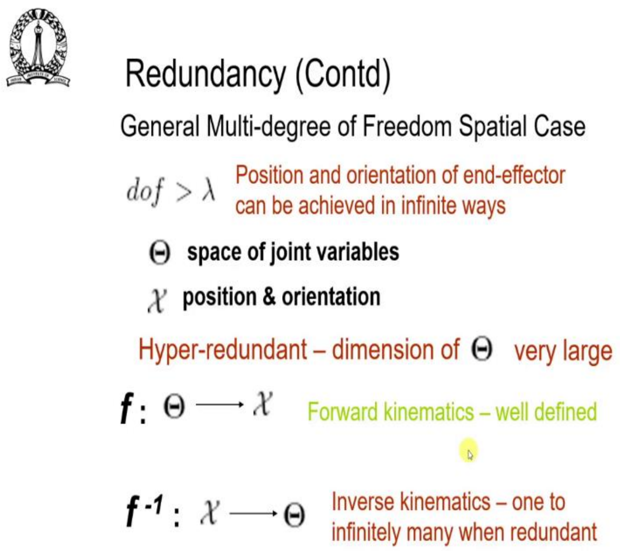
And hence the degree of freedom is 3. So, what it means is that the end-effector of this planar 3R robot can be positioned and oriented arbitrarily in this X-Y plane ok. So, let us continue with this formal way of looking at what is redundancy. I can find the position vector of this point, which is sort of at the middle of this gripper schematic gripper, x and y and I can also find the orientation of this link which is containing the gripper or the end-effector in terms of θ_1, θ_2 and θ_3 which are the rotations of the joint ok.

This we have seen also several times earlier now. So, we can write $x = l_1c_1 + l_2c_{12} + l_3c_{123}$, where c_{12} and c_{123} are nothing but $\cos(\theta_1 + \theta_2)$ and $\cos(\theta_1 + \theta_2 + \theta_3)$ and so on. So, s_{12} means $\sin(\theta_1 + \theta_2)$. So, in a typical kinematics problem if you are given x, y and ϕ we can obtain θ_1, θ_2, θ_3.

And we have seen this in the kinematics of serial robots, this is the non redundant case. So, if I give you x, y, ϕ, I can find the joint angles; however, let us consider that x and y are only of interest, I do not really care what is the orientation. So, in that case we have 2 equations in 3 unknowns. So, given (x, y) we cannot obtain θ_1, θ_2 and θ_3 uniquely or even finitely, many sets of θ_1, θ_2 and θ_3.

There are infinitely many possibilities of θ_1, θ_2, θ_3 which will give you the x and y. So, this is the redundant case. So, given end-effector position and orientation the joint angles cannot be found uniquely or in a finite number of sets ok, but there are infinite possible solutions.

(Refer Slide Time: 07:23)

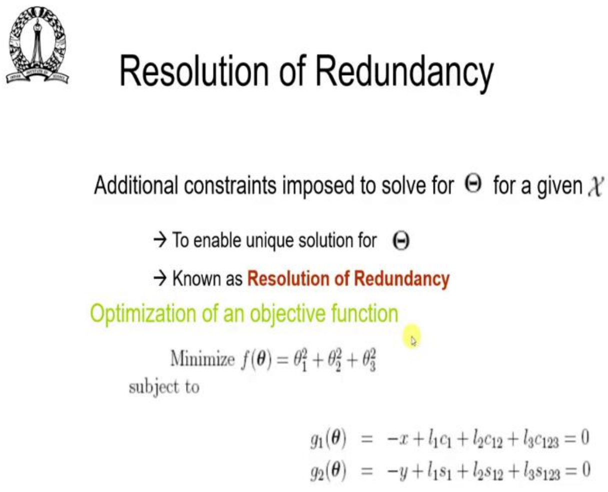


So, in a general multi degree of freedom spatial case the degree of freedom is greater than λ. So, the position and orientation of the end-effector can be achieved in infinite ways. So, for example, if λ were 6 which is what happens when it is moving in space and if the degree of freedom is 7 or 8 or more, then we have a redundant case.

Even more formally, if Θ is the space of joint variables and χ is the position and orientation of the end-effector, in a hyper redundant case the dimension of Θ is very large ok. So, Θ and χ if they are equal then it is non redundant ok, but if Θ is 7 or so then it is not called hyper redundant, if you have like Θ is let us say 20 or 12 or even larger than the they are called hyper redundant robots.

So, in any hyper redundant system the map between Θ and χ which is the forward kinematics or the direct kinematics is well defined, it does not matter whether it is redundant system or non redundant system. What happens is in the inverse direction. So, given χ if I want to find Θ which is the inverse map, then if there are redundancy there are infinitely many solutions ok.

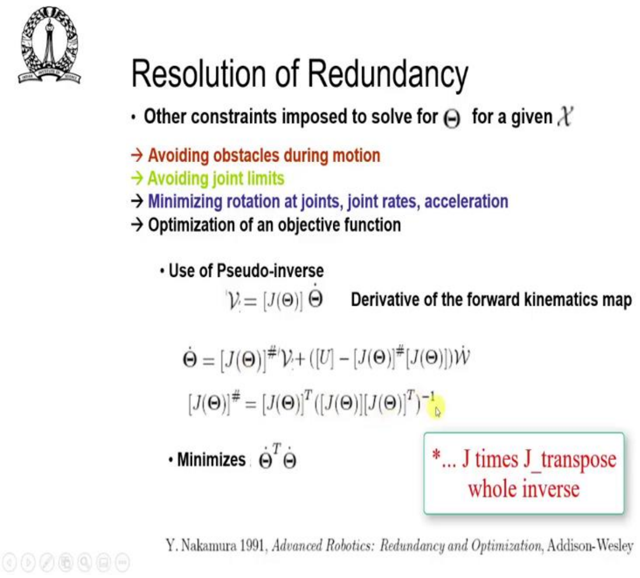
(Refer Slide Time: 08:47)



So, there are many ways to resolve redundancy, basic idea is we need to obtain additional constraints to be imposed on the joint variables for a given χ. So, that we can find a unique or a finitely many set of Θ values ok. So, this is known as resolution of redundancy and as I had shown earlier again in the chapter module on kinematics of serial robots, one possibility is to minimize the joint rotation.

So, if I could minimize $f(\boldsymbol{\theta})$ which is nothing but $(\theta_1^2 + \theta_2^2 + \theta_3^2)$ subject to the fact that x and y are given by the forward kinematics equations ok. So, subject to $g_1(\boldsymbol{\theta}) = 0$, $g_2(\boldsymbol{\theta}) = 0$, where $g_1(\boldsymbol{\theta}) = -x + l_1 c_1 + l_2 c_{12} + l_3 c_{123} = 0$.

(Refer Slide Time: 09:48)



And I have shown you we could solve it and we could obtain solutions for the optimization problem. There are other ways to resolve redundancy and this is what we will look at in little bit more detail in this lecture ok. So, one is we can avoid obstacles ok. So, I will show you how we can rotate the joints, such that the obstacles in the path of the robot are avoided.

We can also avoid joint limits I am not going to show anything here ok. So, again in the previous example of minimizing joint motion in that chapter on kinematics of serial robots, I showed you if there are joint limits how you can still find the solution ok. We can also find a constraint, which minimizes rotation at the joints joint rates or acceleration ok, so or optimization of an arbitrary objective function.

So, if you want to minimize joint rates which is this joints speeds of the rotations of the joints. So, there is a very well known solution which is called as a pseudo inverse solution ok. So, what do we do? We start with the forward kinematics map, which is nothing but given Θ find the χ, find the position and orientation of the end-effector.

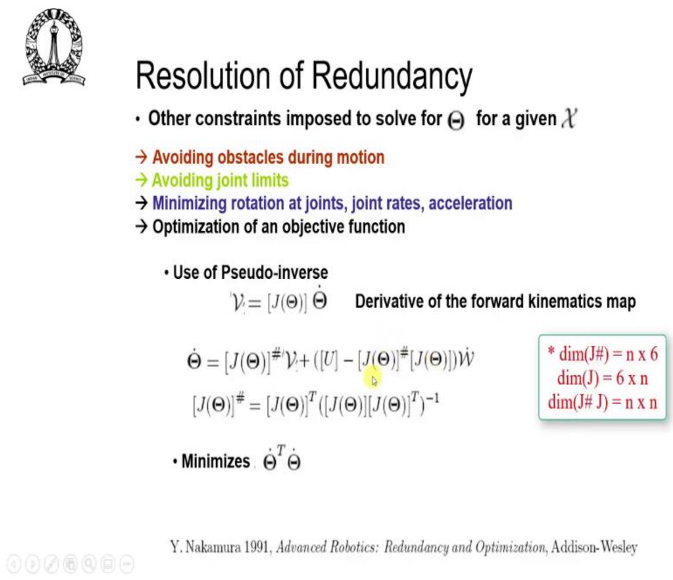
If you take the derivative of this set of equations, we can write that the linear and angular velocity of the end-effector can be written in terms of the Jacobean matrix, $[J(\Theta)]\dot{\Theta}$. So, we have seen this before earlier, if this $[J(\Theta)]$ is non square meaning that there are more joints than unknowns.

So, the number of rows in $[J(\Theta)]$ can at most be 6 ok, because the velocity vector is v_x, v_y, v_z and $\omega_x, \omega_y, \omega_z$ but the number of columns can be many more. It depends on the dimension of this Θ space or $\dot{\Theta}$ space. So, if $[J(\Theta)]$ is non square, there is a solution which is called as the pseudo inverse solution which is given by as follows.

$\dot{\Theta} = [J(\Theta)]^{\#}$ times this velocity vector plus some unity matrix minus $[J(\Theta)]^{\#}[J(\Theta)]$ times $\dot{\mathcal{W}}$. So, this term here is in the null space of the $[J(\Theta)]$ matrix ok. And $[J(\Theta)]^{\#}$ which is the pseudo inverse of the Jacobean matrix is given by $[J(\Theta)]^T([J(\Theta)]\ [J(\Theta)]^{\mathrm{T}})^{-1}$. So, if you see that this is like $6 \times n$ ok. So, J theta into J theta transpose will become a 6×6 matrix ok, $(6 \times n) \times (n \times 6)$ this is transpose ok.

So, hence an inverse exists it is a squared matrix now ok. The other important thing in this pseudo inverse is that you can see that this $[J(\Theta)]^{\#}\ [J(\Theta)]$ is not identity ok.

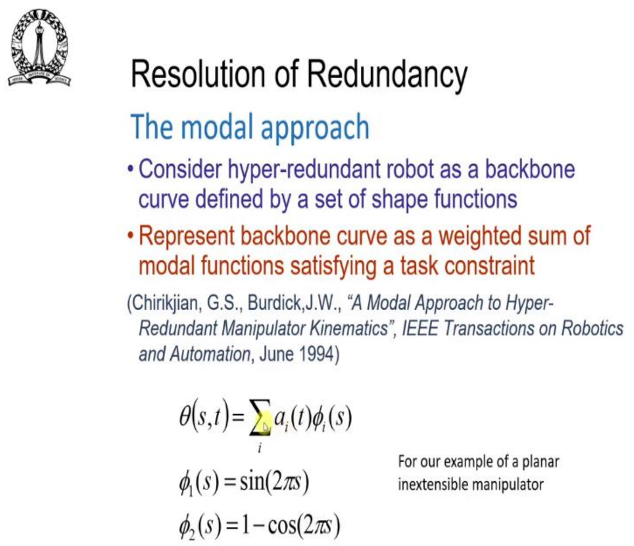
(Refer Slide Time: 13:05)



So, this is $n \times 6$ times $6 \times n$. So, this is a $n \times n$ matrix. So, this is not an identity. So, the, right multiplication of $[J(\Theta)]^{\#}$ by $[J(\Theta)]$ does not give you identity, but the left multiplication of $[J(\Theta)]^{\#}$, by $[J(\Theta)]$ will give you identity. So, you can work it out. And it turns out that this $[J(\Theta)]^{\#}$ solution the pseudo inverse of $[J(\Theta)]$.

So, $\dot{\Theta} = [J(\Theta)]^{\#}\mathcal{V}$ minimizes $\dot{\Theta}^T\dot{\Theta}$. So, previously we had looked at minimizing joint rotation, but the pseudo inverse minimizes the sum of the squares of the joint rates ok. It minimizes $\dot{\theta}_1^{\ 2} + \dot{\theta}_2^{\ 2}$ and so on till $\dot{\theta}_n^{\ 2}$.

(Refer Slide Time: 14:07)

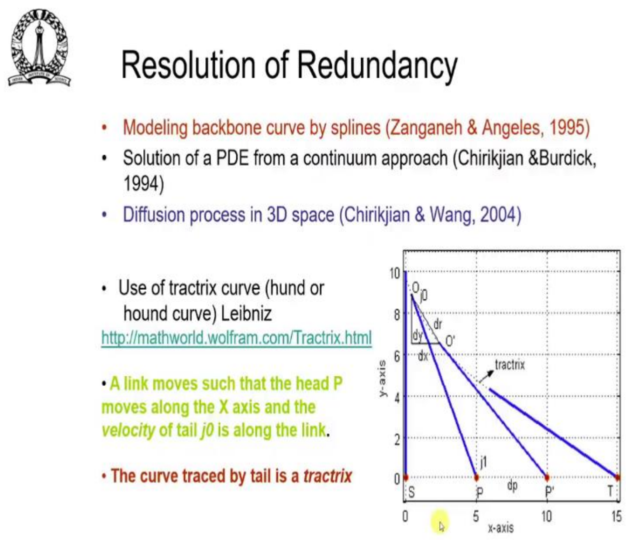


There is also one more very well known approach which is called the modal approach ok. So, basically in a modal approach what we do is we consider the hyper redundant robot as a backbone curve defined by a set of shape functions ok. So, those of you who have done any course in geometric modeling or CAD, you can define a curve in 3D space by means of some shape functions ok.

So, it could be parametric, cubic, it could be some NURBS or it could be some other shapes piecewise shapes. So, we represent this backbone using a 3D curve called the backbone curve by means of shape functions ok, and we represent this backbone curve as a weighted sum of modal functions satisfying task constraint ok. So, I want to move this robot, there is a certain task. So, I can find these modal functions and the weighted sum of these modal functions determine the task ok.

So, for example, $\theta(s,t) = \sum_i a_i(t)\phi_i(s)$. So, this $\phi_i(s)$ are the shape functions. So, these are nothing but there are two of them in this example which we will show later is $\sin(2\pi s)$ and $(1 - \cos(2\pi s))$. So, s is the arc length ok. So, θ at any joint is given by some linear combination of the shape functions and some functions which are functions of time. So, as the robot moves $a_i(t)$ will change.

(Refer Slide Time: 15:48)

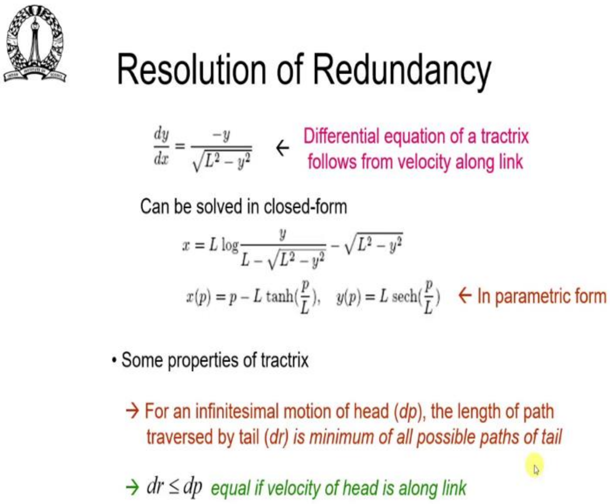


There is a third way to resolve redundancy ok, actually there are many ways to resolve redundancy. This whole idea of this backbone curve was also resolved using splines. We can also resolve this redundancy basically meaning impose extract constraints to solve for Θ by using something called as a PDE from a continuum approach ok, we do not want to get into this.

We can also resolve redundancy again by work by it is a diffusion process in 3D space. So, these two are very very advanced and we do not want to get into this. So, this modeling of backbone curves by splines can also be done ok. We had looked at one way to resolve redundancy by using something called the tractrix curve ok. So, I had discussed this earlier again in the module on kinematics of serial robots.

So, just to recapitulate a tractrix curve is nothing but the curve traced by the tail of a link, which is originally along the Y axis and we want to move the head of the link parallel to the X axis, such that the velocity vector of the tail is always at every instant of time along the link ok. So, to repeat a link moves such that the head P moves along the X axis and the velocity of the tail j0 in this figure is always along the length. And the curve which is obtained for such a constraint is this curve called tractrix ok.

(Refer Slide Time: 17:34)



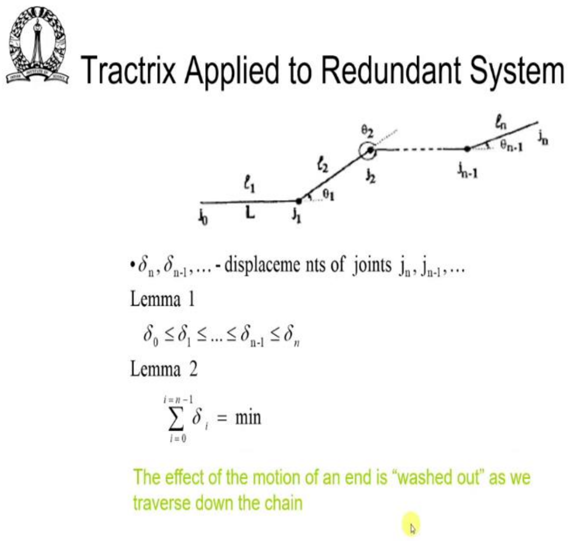
And I had looked at the equation of a tractrix last time and this is again to recapitulate the equation of tractrix is given by $\frac{dy}{dx} = \frac{-y}{\sqrt{L^2 - y^2}}$ where L is the length of the link and I had told you that we can solve this in closed form interestingly, we get a solution which is x as a function of y ok.

So, $x = L \log \frac{y}{L - \sqrt{L^2 - y^2}} - \sqrt{L^2 - y^2}$. Typically, we find y as a function of x, but in this case we find x as a function of y. We can also find this x and y in terms of a parametric variable p using tan hyperbolic and sec hyperbolic ok.

Again to recapitulate, some of the important properties of a tractrix are that for an infinite decimal motion of the head dp the length of the path traverse by the tail dr is minimum of all possible paths of the tail ok. So, the dr when we are following this tractrix constraint ok, or when it is following the curve on the tractrix it is $dr \leq dp$. It is only equal if the velocity of the head is along the link.

So, you can think of it that if the velocity of this head is this direction along the direction of the link, so then dr will be less than dp at any other place. So, dp is this way, dr is this one dr will be less than dp if it is moving in this direction head again dr will be less than dp. So, at every direction of motion of the head, except along the link itself, dr is less than or equal to dp ok.

(Refer Slide Time: 19:35)



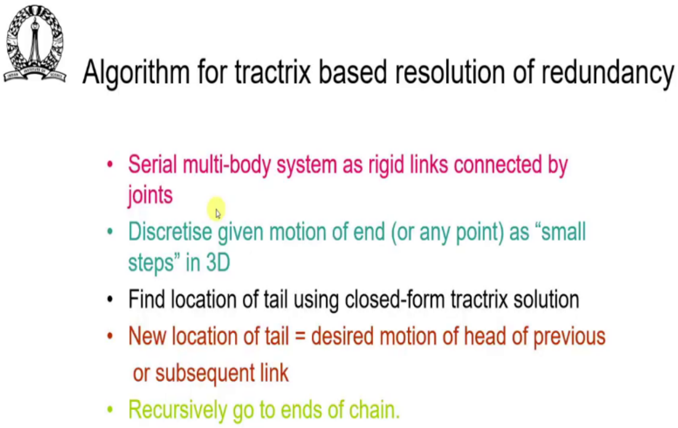
So, we applied this tractrix to a redundant or a hyper redundant system in the following way. So, this is a schematic of a hyper redundant robot with many joints, n joints ok. So, if I move the head of this hyper redundant chain by some dp, I can find out what is happening to the tail which is dr using the tractrix formula.

Now, for this previous link this dr will be dp and I can find again the motion of the tail of the previous link likewise I can go backwards and I can find the motion of the final end point of this robot ok. And in all such cases $\delta_0 \leq \delta_1 \leq \cdots \leq \delta_n$. So, if this is moving by δ_n, δ_0 will always be less than or equal to δ_n and this is true for each δ.

It is also true that the sum of all these δ's is minimum ok, this follows from what I discussed last time that the motion of the tail is the smallest ok, when it follows the tractrix otherwise it is equal. So, because of this δ_0 less than or equal to all the way to δ_n the effect of the motion of the end is washed out as we traverse down the chain. So, for example, if this is moving by 1 centimeter, this will be smaller than 1 centimeter, let us say 0.9, then at some this one will be 0.8.

And all the way, to some very small number, at the end ok, so the motion is getting washed out ok. So, this looks natural. So, if you think of a snake, if the snake is moving its head or if you think of a rope you move the one end of the rope, the further end of the rope or the furthest end from the head of the snake moves the least ok.

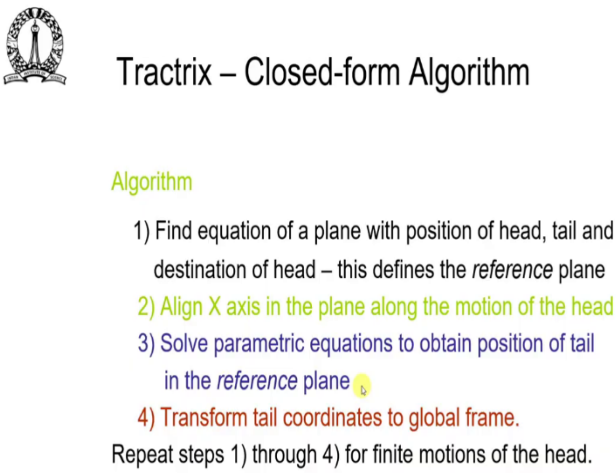
(Refer Slide Time: 21:36)



So, what is the algorithm for tractrix based resolution of redundancy we have a serial multi body chain as rigid links connected by joints. So, we have to assume that this rope or this snake has bunch of rigid links connected by joints ok, then we discretize the given motion of the end as small steps in 3D. So, instead of moving a lot, we take small small steps.

Find the location of the tail using closed form tractrix solution, the new location of the tail is equal to the desired motion of the head of the previous or subsequent link and we recursively go to the ends of the chain ok.

(Refer Slide Time: 22:22)

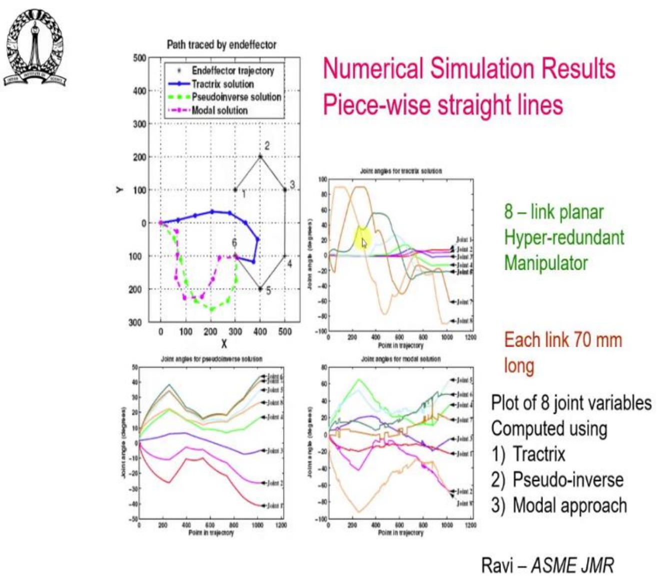


Just a little bit more detail, how do we find the motion of the tail when it is moving in 3D. So, we find the equation of a plane with the position of the head position of the tail and the destination of the head ok, these 3 points define a plane. So, we call this the reference plane, align the X axis in the plane along the motion of the head. So, it starts from somewhere and it wants to go to some destination point that is the X axis ok.

We solve the parametric equation to obtain the position of the tail in this reference plane. So, remember dp was along the X axis now also it is along the X axis except this X axis is some arbitrary plane in 3D ok. Then we transform the tail coordinates back to the global frame or reference coordinate system, because we know the orientation of this plane and we repeat these steps 1 through 4 for a finite motion of the head.

So, we want to take a small steps find the plane, find the tail location in this plane, then transform it back to the global, then for the previous link again you do the same thing and until and unless you find the motion of each one of these links ok.

(Refer Slide Time: 23:46)



So, as an example again we have briefly discussed this example earlier. So, I have an 8 link planar hyper redundant manipulator, which wants to trace this path which is basically a hexagon. So, it wants to go from 1 to 2, 2 to 3, 3 to 4, 4 to 5 and 5 to 6 and then back to 1 ok. So, we can solve this task for this 8 link robot using this 3 techniques, which I discussed; one is the tractrix based solution, one is the pseudo inverse based solution and one is the modal solution ok.

So, this dark blue is the solution of the point when it comes to 6 from 1 to 6 when we use the tractrix based solution, this is the shape of the robot. In the in the pseudo inverse solution this is the final shape of the robot, this green dotted line and this pink or purple dotted line is what we get if you use it and solve it for the modal approach ok, using the model approach.

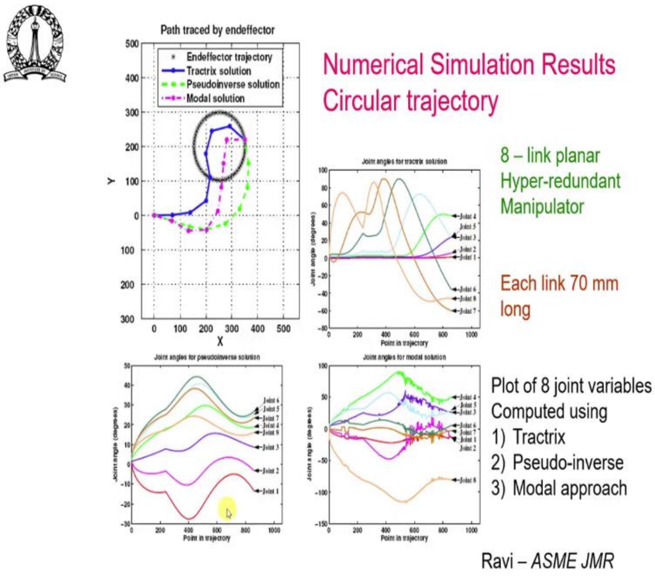
We can also plot all the joint variables ok. So, I did not tell you how to obtain the joint variables, but it is very easy. So, if you locate the head and the tail at 2 instants of time, then we can find how much the tail has moved. So, we can take the dot product of these 2 vectors and then that will be cosine of some angle ok, how much the joint has to rotate ok.

So, at two close by instants of time we find the dot product of the 2 vectors and then we obtain theta ok. So, we can plot θ for again for these 3 approaches. So, you can see that the tractrix solution looks like this. So, important thing here is this yellow line which is joint 8. So, as you can see that the joint 8 moves a lot ok, it moves the most. The joint 7 which is the next slightly darker yellow line, moves a little less than that ok.

And so and joint 6 moves even smaller and finally, joint 1 moves very little ok. So, what is happening in the tractrix based approach? The motion of the joint reduces as you go from the head of this 8 link hyper redundant robot all the way to the base. If you look at the pseudo inverse solution that is not true, the joint 1 which is this curve moves quite a bit ok, joint 2 also moves quite a bit, joint 3 also moves quite a bit ok.

So, all the joints including the 1st and 2nd, 3rd another to the end they all move quite a lot ok, the modal solution also is like that all the joints move quite a lot ok. So, the summary of this plot is that in the tractrix based solution as expected, the base joints the joints which are furthest from the head move the least ok.

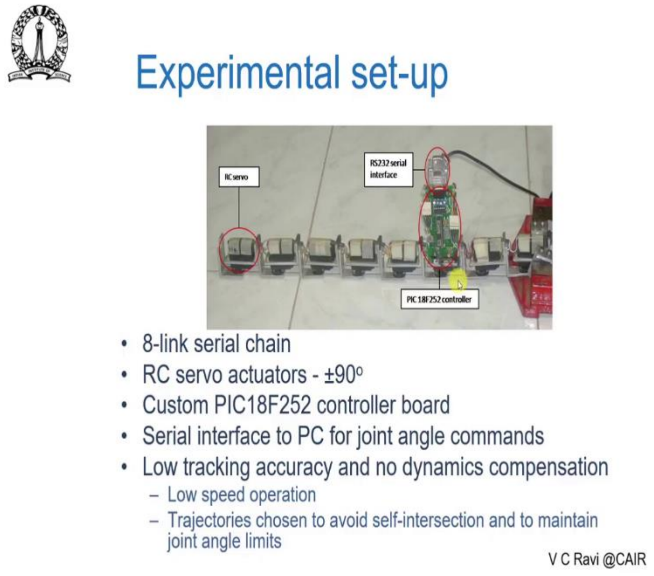
(Refer Slide Time: 27:11)



And we can try another trajectory this is the solution or resolution of redundancy for again that 8 link robot if the tip or the head moves in a circle. So, this dark circle is the end affected trajectory that we want this 8 link robot to move the tractrix solution, looks like this after it has moved and come back to its original point. The green again shows the pseudo inverse solution and this purple shows the modal solution and we can again plot the joints ok.

So, in the tractrix solution as you can see the 7 and 8 moves much more than 1 and 2 ok. So, in this case it turns out 7 moves a little bit more than 8, but approximately same but 6 7 and 8 which are for this joint they move much much smaller than 1 2 and 3 whereas, in the pseudo inverse solution they move quite a bit all the joints moves quite a bit ok.

(Refer Slide Time: 28:25)

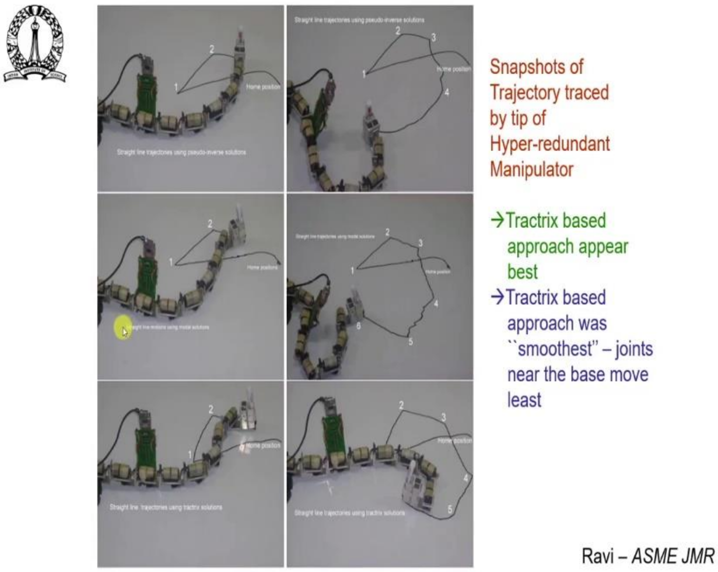


And likewise in the modal solution all the joints move quite a bit. So, we built one such 8 link robot with RC servo motors. And then there was a serial interface RS232 serial interface to connect to a computer. So, we can compute all the motion of the joints. So, this is a controller card which can control all these motors. So, it is fixed at the base ok, the motors these RC servo motors could move only plus minus 90 degrees ok.

And this is a some custom design controller board which could control all these 8 motors and this is the serial link to a PC for joint angle commands. So, we need to set at each instant of time ok. How much each joint should rotate? And in this example we are going to move slowly ok, because there is no way to compensate for the dynamics of the links in this experiment.

So, it is a low speed operation and also the trajectories are chosen to avoid self intersection and to maintain joint angle limits. So, there is a limit of 90 degrees plus minus and also while moving I do not want one link to go and hit another link ok. So, we have cleverly chosen some trajectories such that these things does not happen ok.

(Refer Slide Time: 29:49)



And the trajectories are the same as what we did for simulation. So, I want to trace a hexagon ok. So, on the left I am showing you a snapshot of this robot tracing that hexagon initially on the right, what happens after it has sort of come almost towards the end when it is coming to the point 6 ok.

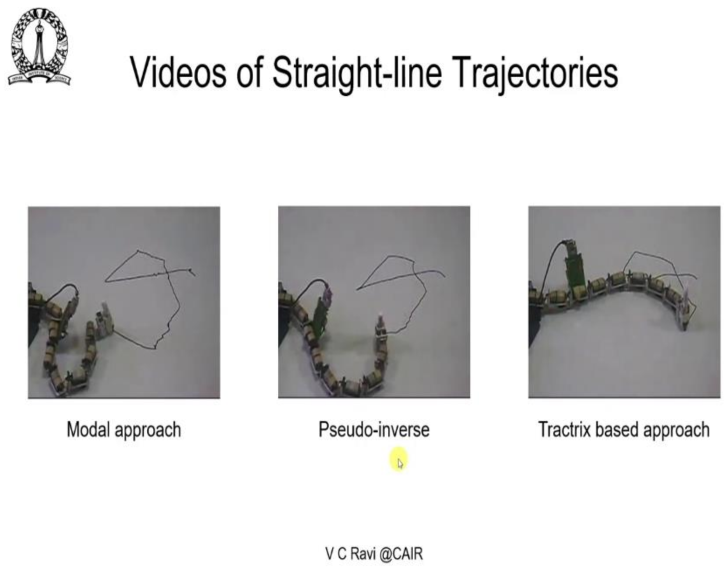
So, remember the hexagon is 1, 2, 3, 4, 5 and 6. So, the top one is the pseudo inverse solution, the bottom one is the tractrix based solution and this one is the modal solution ok. So, what you can see is that this robot starts from some home position, there is a pen which is attached to the tip of the robot ok, and it will draw these lines as it is moving and then it goes from this home position to 1, then 2, then 3, 4, 5 and 6 and at these are the snapshots of what is happening ok.

So, what you can see clearly is this hexagon which is traced by the tractrix based solution is sort of looking the best ok. So, some of this jerky thing is because the robot was slipping while it was moving, the pseudo inverse solution does not look too good. So, initially it is fine at least 2 legs of the 2 arms of the hexagon is there, but after between 3 and 4 and after 4 it you know it is doing some strange things.

So, it looks like the tractrix based approach was the smoothest ok, and again the basic idea is that we are only moving the joints, which are furthest away from the base. So, you can see here in the modal solution this joint is moving ok. In the pseudo inverse solution again the first joint is moving, all joints are moving whereas, in the tractrix based solution this joint moves very little, the base joints move very little and the furthest joints moves alot ok.

So, this is giving smoother motions ok. Is that make sense? Yes, because the first base joint must move all the links. So, it sees a lot of inertia ok. So, in the tractrix based approach the base joints do not move much. So, it is not moving a large inertia whereas, this other 2 they are moving the full robot the all the links and hence the base joints are seeing a lot of inertia and then it is not able to move properly ok.

(Refer Slide Time: 32:38)

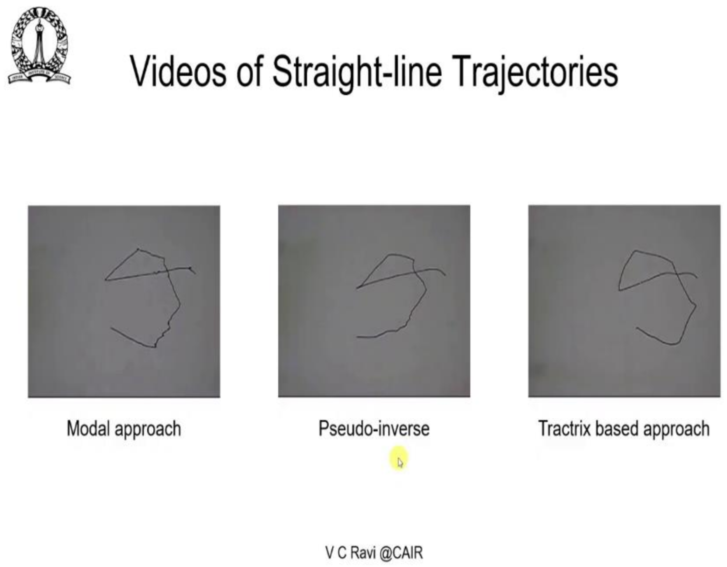


So, let us show you some videos of this 8 link robot trying to stress straight line trajectories. So, basically this is the same straight line motion which form that hexagon ok. So, this is the modal approach, this is the robot moving according to the pseudo inverse based resolution scheme and the last one is this robot moving where the resolution is based on the tractrix based approach.

So, as you can see in the modal approach and in the pseudo inverse approach all these links are moving quite a bit whereas, in the tractrix based approach as expected. So, it is trying to move as much as possible using the last few links, when it cannot do so, when it cannot trace the trajectory it is going backwards and it is moving the base motors or base links.

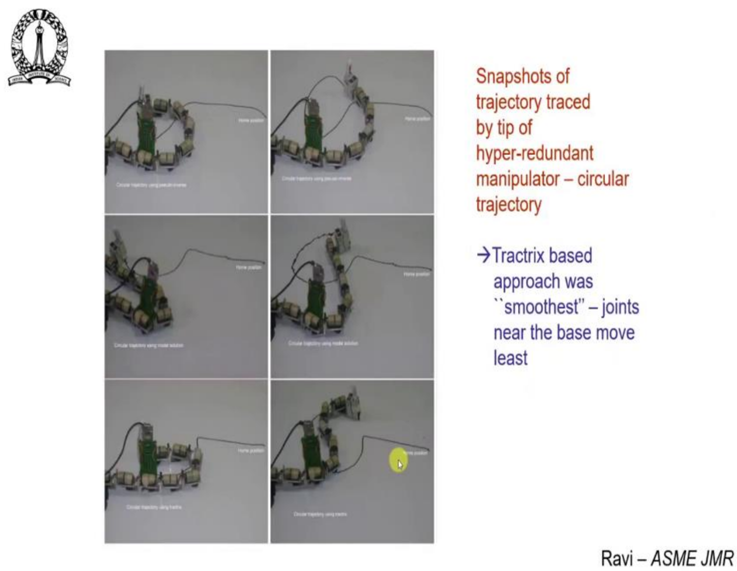
And as you can see sometimes it slips ok. So, the slipping is not due to the resolution scheme. It is due to the fact that this robot will slip on this, it was tracing this curve on a tile floor ok.

(Refer Slide Time: 34:20)



So, this problem of slipping and then jerking and so on is reduced in the tractrix base approach, but it is never the less, not due to the resolution scheme ok. So, these are the plots which are traced by a pen which is attached to the tip of the robot ok.

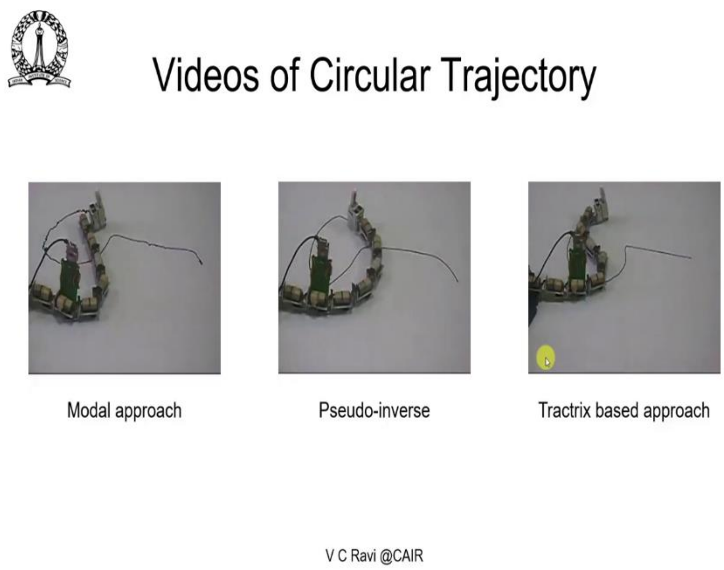
(Refer Slide Time: 34:43)



We can also try to see whether how good it is doing that circle ok, I showed you 2 simulation results and I am going to show you 2 experimental results. One for piecewise straight line which I showed last time and then for the circle ok. So, again the top portion is the pseudo the top 2 figures or pictures are for the pseudo inverse scheme ok.

So, this is some in between when it is tracing the circle, this is towards the end of the motion of when it is traced the circle, here it is the modal solution and this is also the modal solution and finally, this is the tractrix based solution. So, you cannot see the circle, but you can see it later or you can see it at the end, this looks more like a circle than these two ok. What the curve it has traced and again we will see that the smoothest motion is when the joints near the base moves the least.

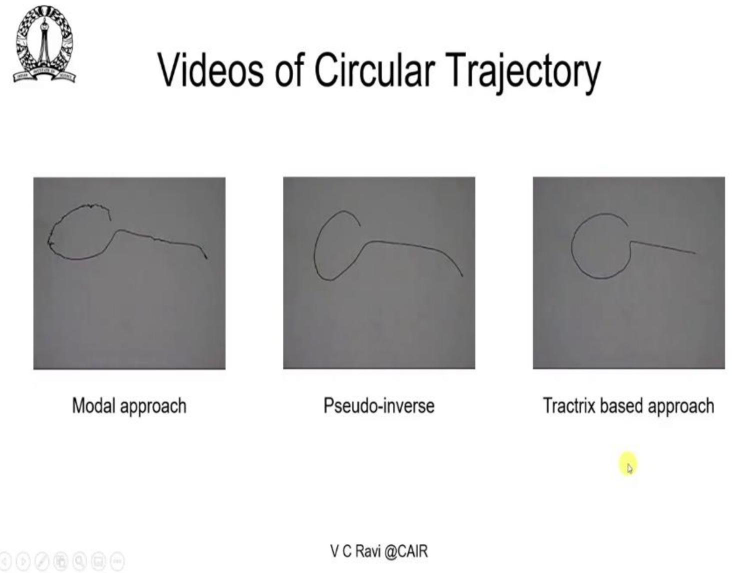
(Refer Slide Time: 35:46)



So, here are 3 videos again the 1st one is the modal approach, 2nd one is the pseudo inverse based approach and the 3rd one is the tractrix based approach. So, in all cases the trajectory is the same, the controller is the same, only the joint angles are calculated by different resolution schemes ok. So, timing, velocities everything is same, it is the same tiled floor ok.

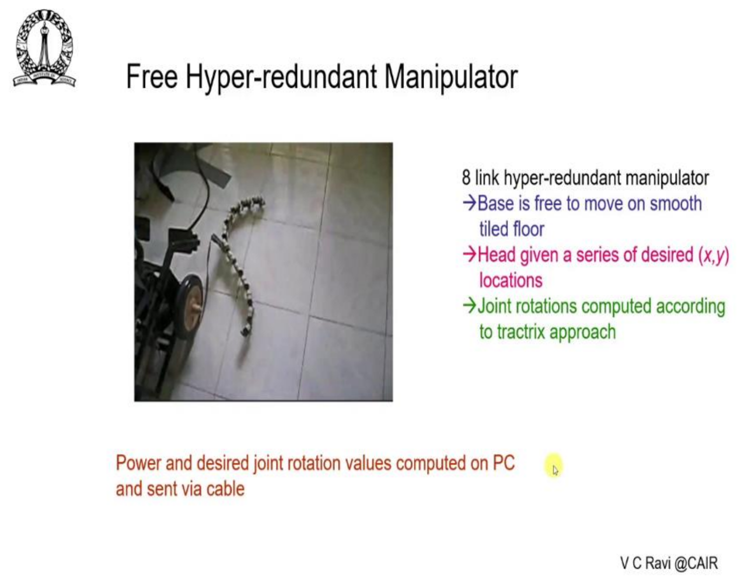
So, most of the other conditions are similar except that The resolution scheme is one case it is modal approach, one case it is pseudo inverse approach, one case it is tractrix based approach. So, as you can see in the modal and as well as the pseudo inverse approach, all the links are moving. Some of this jerkiness which you see in the modal approach is due to the fact, that it is slipping even though we tried our best to move slowly, but it never the less will always slip a little bit.

(Refer Slide Time: 37:03)



So, it starts from some home position traces the circle and then it stops ok. So, again you can see in the tractrix based approach, the motion is smoothest and the tip trajectory traced is closest to a circle.

(Refer Slide Time: 37:27)



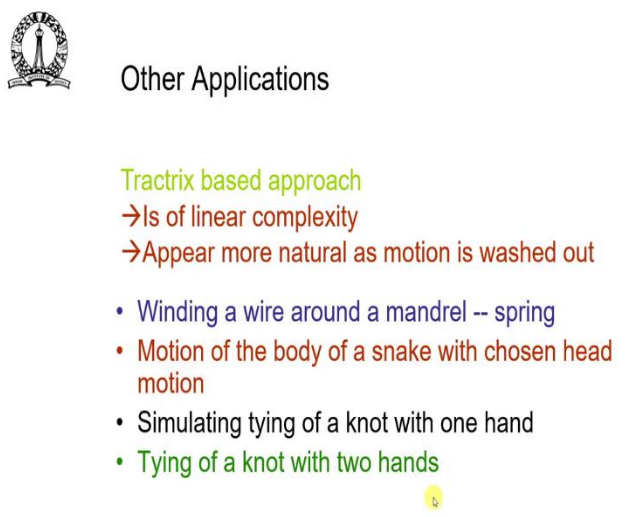
We can also use this tractrix based approach, if you have a free hyper redundant manipulator ok. A free hyper redundant manipulator means that the base is not fixed ok.

So, as the head moves the tail can also move ok, so I am going to show you a video where I have given the desired motion of this head ok.

The joint rotations are computed according to the tractrix based approach and then we will see this 8 link robot move on this tile floor ok. So, we have intentionally given the motion of the head in some sort of like a sinusoidal motion. So, you can see what the rest of the body is doing ok. So, I think you will agree with me that this motion looks quite natural ok, this is as if it is a snake is moving through this tile floor ok.

So, like in the previous case the power and the desired joint motion joint values are computed on a PC and sent via cable ok.

(Refer Slide Time: 38:59)

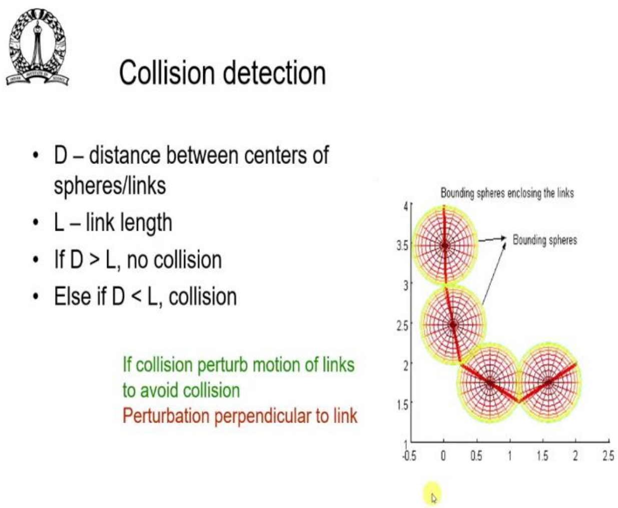


Laptop and sent via cable. We can also use this tractrix based approach for some other interesting simulations ok. So, it turns out that this tractrix based approach is of linear complexity, the motion appears to be very natural, the tail motion is washed out. So, we can use this idea or this technique to show things like if I want to wind the wire around a mandrel to make a spring ok.

So, anyone who has made a spring like this, you can see that as the wire is wound ok, the portion which is free moves a lot and it sort of winds around the mandrel, we can also show the motion of a snake with the chosen head motion again the snake is discretized into several rigid bodies with some joints.

I can also show the tying of a knot with one hand and with two hands. So, I take a rope in one hand means I move only one hand and tie a knot ok, with two hand means you move both hands. So, in the two handed case I will show you what it looks like. So, you can visualize these motions ok.

(Refer Slide Time: 40:12)



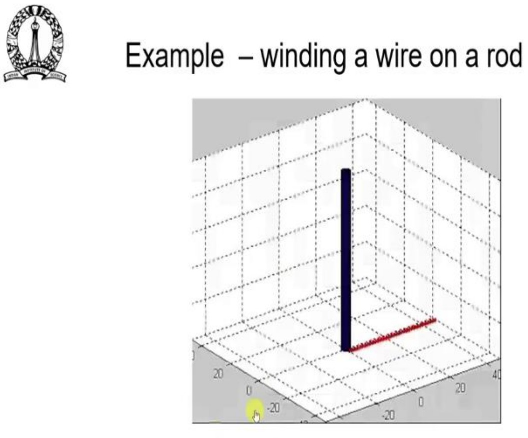
So, one important thing when you want to do these things for flexible ropes and snakes and various other things we need to make sure that they are not hitting each other. So, one part of the rope is not hitting each other and when you discretize this rope into rigid bodies, the rigid body should not collide with the previous or some other rigid body far away. So, for collision detection and avoidance we have used a very crude approach.

So, basically this shows here that these are the links of the robot; we bound these links by spheres ok. So, there are 4 links and there are 4 spheres and the distance between the centers of the spheres or links is D, L is the link length. So, if D is greater than L, then there is no collision ok, if D is less than L then there will be collision.

So, you can see if this link is folding, then this sphere will collide with the other sphere. So, the distance between the center of the spheres and links will be less than the link length ok. So, this is $L/2$, $L/2$. So, the distance will be less than L, if it is colliding ok. So, if there is a collision we basically stop the simulation, we perturb the motion of the links in a perpendicular direction to the link.

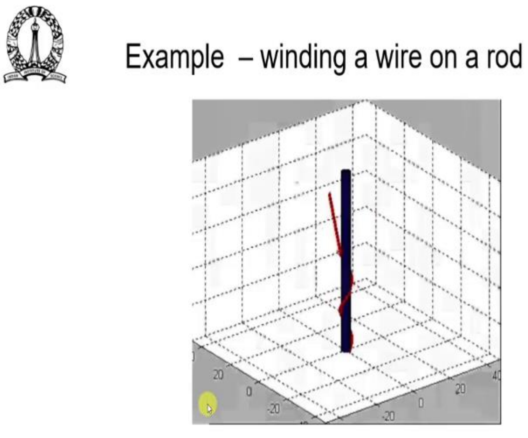
So, hence it will no longer collide. So, this is a crude way of doing it that there are much better ways to detect collision and avoid collision, but nevertheless for these simulations we have used this strategy ok.

(Refer Slide Time: 41:59)



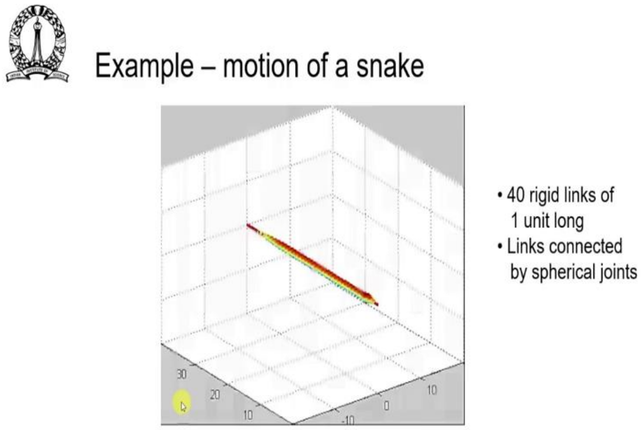
So, I am going to show you a winding of a wire on a spring ok.

(Refer Slide Time: 42:09)



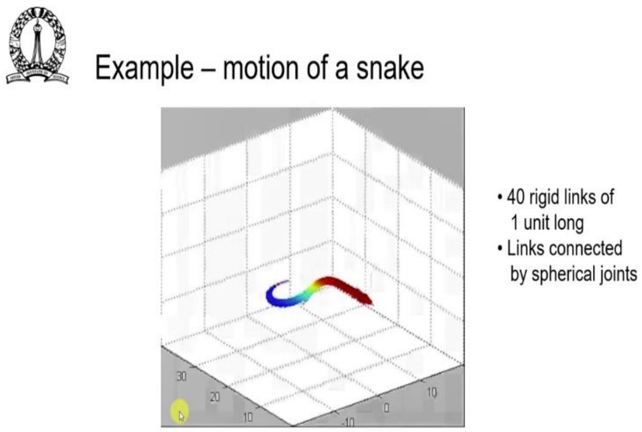
So, we can use this to show that the motion of the wire on this mandrel, it looks very natural ok. So, let us show it once more ok.

(Refer Slide Time: 42:25)



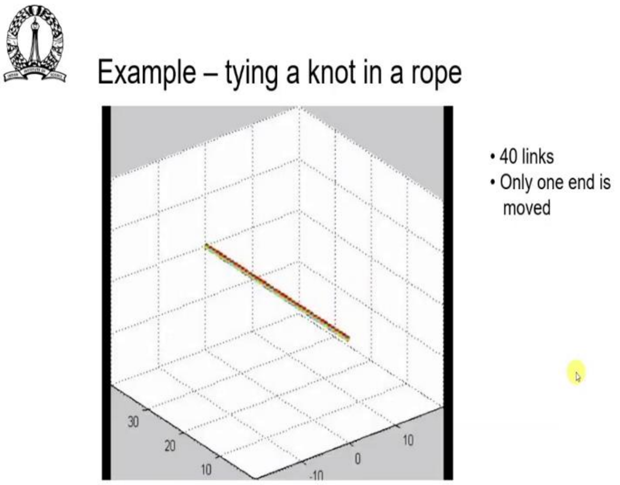
So, we can also show the motion of a snake ok. So, in this example the snake is modeled as 40 rigid links of 1 unit long, the links are connected by spherical joints ok.

(Refer Slide Time: 42:41)



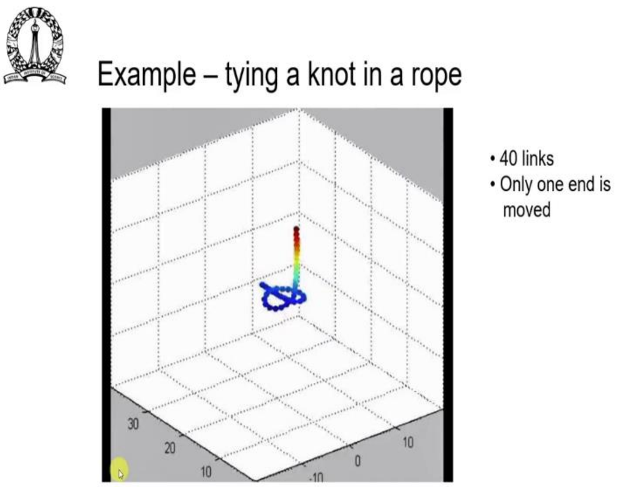
So, there is the chosen motion of the head and then we compute all the motion of each link according to the tractrix based approach, and then we simulate this and then we animate this links motion and show it that how it moves how the whole snake moves ok.

(Refer Slide Time: 43:05)



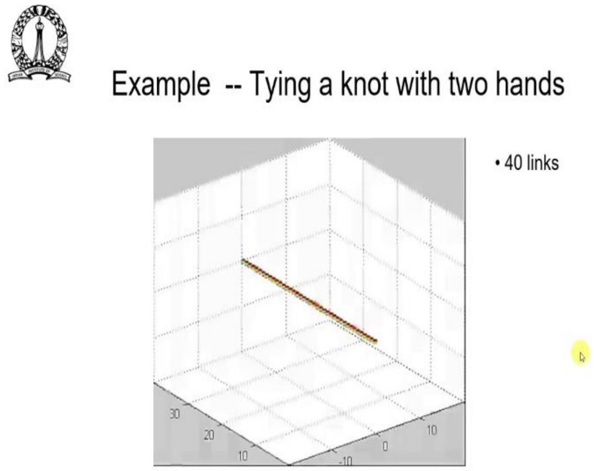
This is the example showing tying a knot in a rope ok, again there are 40 links and one end is moved ok.

(Refer Slide Time: 43:16)



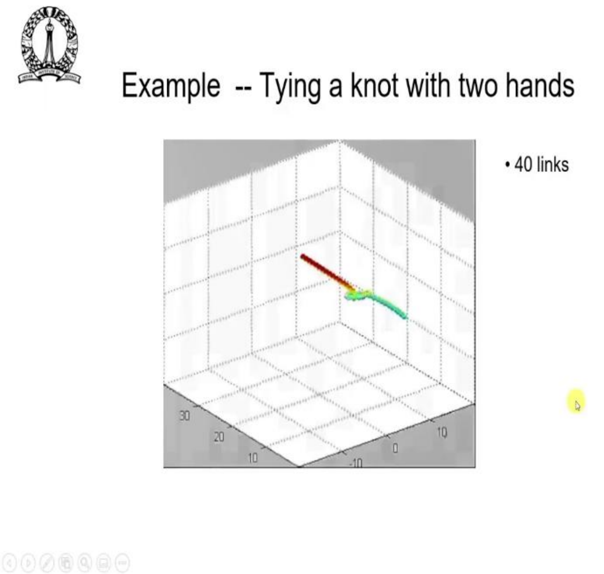
So, if I want to show in a computer screen tying off a knot, you can see it is not a very easy thing to do to make it look natural ok. So, in this case there are 40 links each link is like a sphere ok, and then you move these links, such that it looks like as if it is tying a knot. This is when two hands are moved to tie a knot with two hands.

(Refer Slide Time: 43:42)

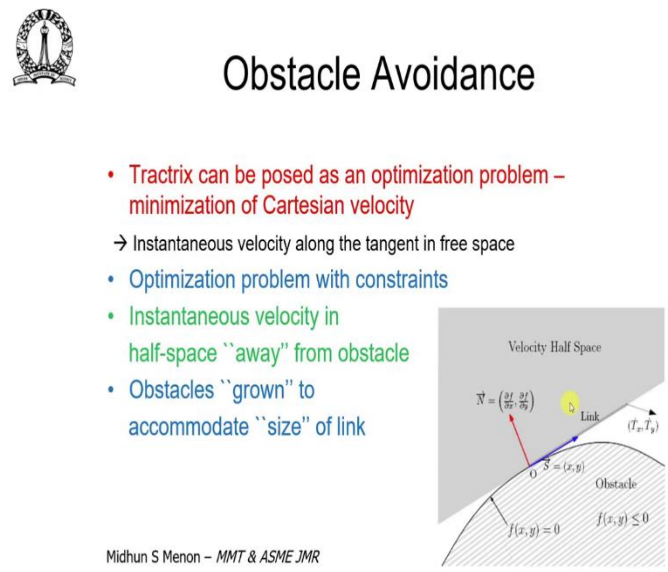


So, in this example the simulation is done that you move one hand stop, move the second hand stop move. So, alternately we move both hands and we store these values and then we animate it. So, here also there are 40 links.

(Refer Slide Time: 44:05)



(Refer Slide Time: 44:14)



So, the tractrix based approach can also be extended to do obstacle avoidance, and the way to do it is following. So, the tractrix based approach in general is basically an optimization problem. What are we doing? We are minimizing the Cartesian velocity of the tail ok. So, if you pose it as an optimization problem we can also include constraints ok.

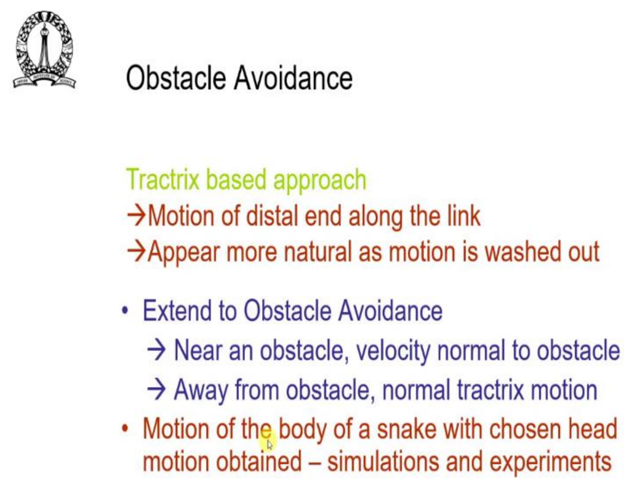
So, if it is in free space the instantaneous velocity is along the tangent in free space of this curve or of this object ok. So, velocity is along the tangent. So, if you take a full curve, you can show that the tractrix base approach, every point has a velocity which are tangent to the curve ok.

So, this is without constraint, when you optimize with constraints, then we can set that the instantaneous velocity is not always along the tangent to the curve, but it is in the half space away from the obstacle ok. So, for example, if there is an obstacle here and this curve is there. So, at some point the velocity is not along this tangent to this link, but the velocity in this red direction which is perpendicular to the link ok.

In free space the velocity will be along this blue line it is always along the link, but when it is close to an obstacle. We can make it go in the perpendicular direction and if you know the obstacle, we can find the normal which is $\left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$, and then we can give the velocity in this red direction ok. So, for a curve which is $f(x, y) = 0$, I know the normal and then the link which is lying on this curve, the velocity is along the red direction ok, we do a little bit of trick.

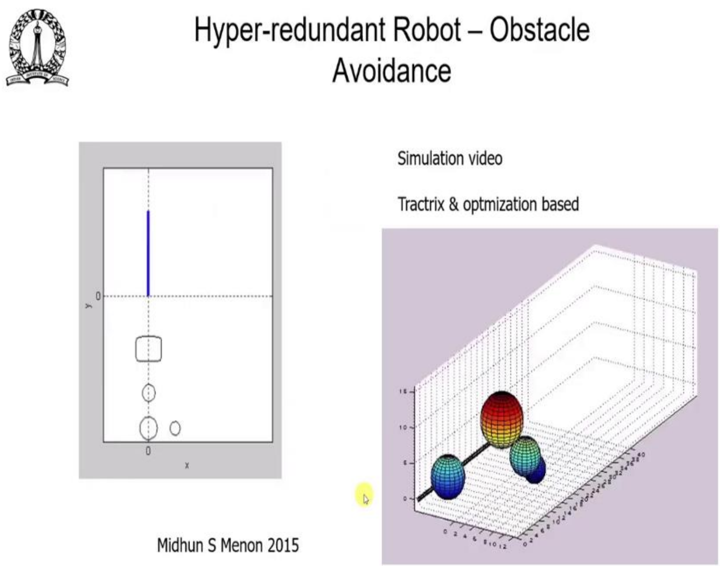
We can increase the size of the obstacles, such that no part of the link will hit the obstacle and we plan for the center of the link ok, because it is sort of hard to compute the trajectory for all points on a finite length link. So, we grow the obstacle we make the plan for the center of the link and if the obstacle is grown a little bit in the actual implementation it will not hit ok.

(Refer Slide Time: 46:48)



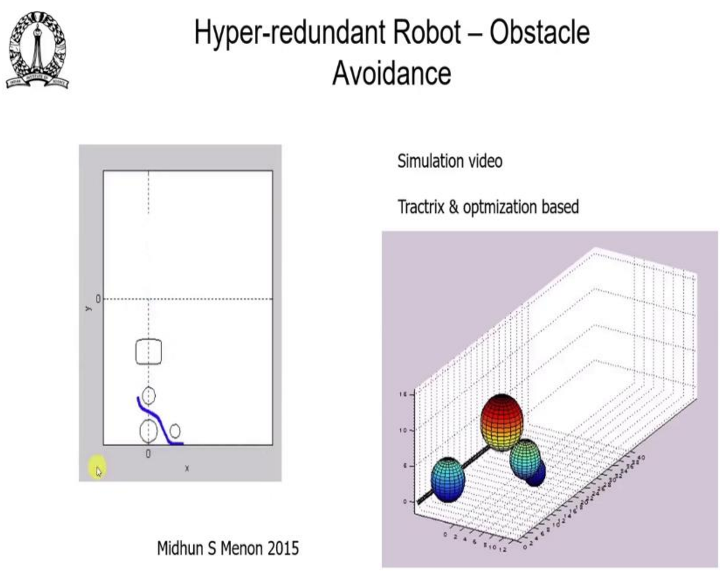
So, in the tractrix based approach the motion of the distal end is along the link. So, hence it appears more natural and the motion is washed out. We extend to obstacle avoidance near an obstacle. The velocity is normal to the obstacle and away from the obstacle, it is the normal tractrix solution along the link ok. So, we have a motion of the body of the snake with chosen head motion and I will show you some simulation and experiments ok.

(Refer Slide Time: 47:21)



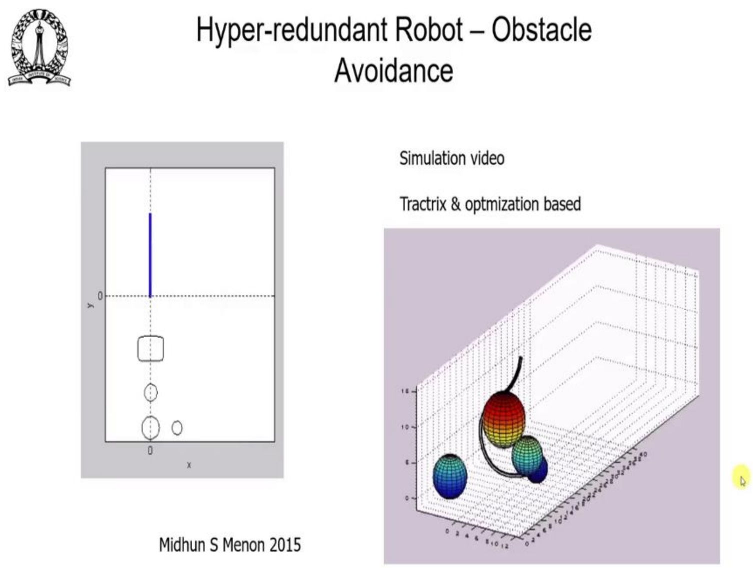
So, this is a planar simulation. So, these are these obstacles one object some sphere or square and then there are the circular obstacles and this blue is the snake or this hyper redundant robot which is discretized into small links. So, I want to pass this head around this obstacle, such that no portion of this hyper redundant robot hits any of these obstacles. So, this is what I will show you ok.

(Refer Slide Time: 47:56)



So, as you could see that none of the part of this blue snake or hyper redundant robot would hit the object, we can do this in 3D also.

(Refer Slide Time: 48:12)

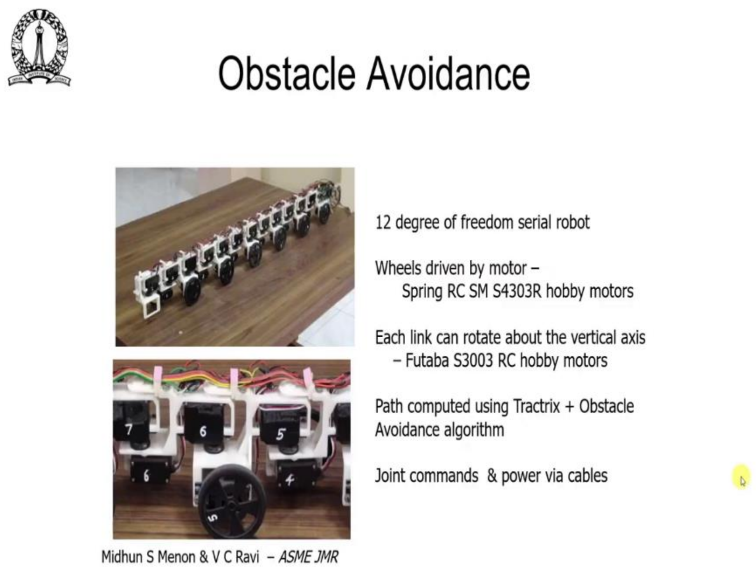


So, in 3D there are these spheres which are these obstacles and this black thing is this hyper redundant robot. So, we have plotted or chosen a path of the head, such that it does not hit any of the obstacle, but how do we ensure that none of the following links ok, also does not hit the obstacle we use this idea of obstacle avoidance ok.

So, when it is near to the object that small link, the motion will be perpendicular to the object ok. So, and you can see it works quite well, and it shows that as if this snake or this hyper redundant robot is going very close to the obstacle sort of gliding over it, and then it is moving. So, once it has moved far away you can see that the normal tractrix based approach is followed ok.

So, in free space it is the normal tractrix based approach and when it is close to the obstacle it is perpendicular, the velocity component is perpendicular ok. So, let us stop here and continue. So, this simulation will continue for a while.

(Refer Slide Time: 49:39)

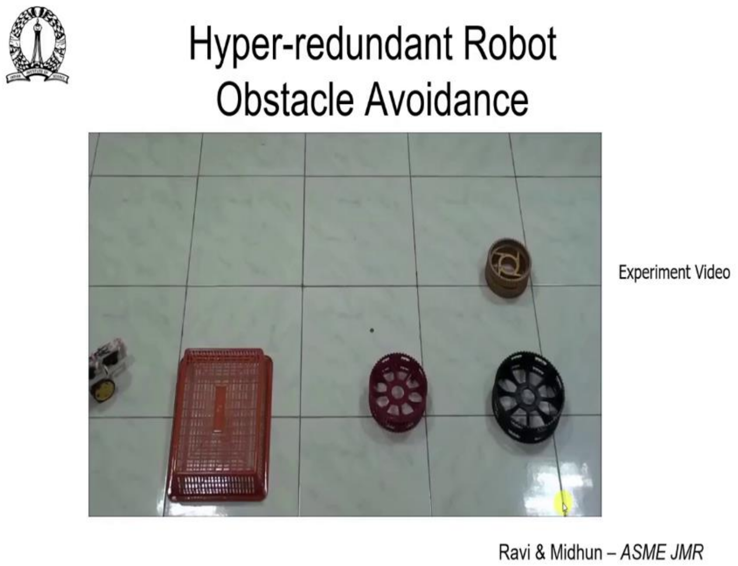


We also built a robot its very similar to the previous hyper redundant robot, but now it has 12 links, it has 12 degrees of freedom. Again each link is driven by another different RC servo motor, it is this name ok, the wheels are driven by these motors, but each link are rotated using this another set of robot ok.

So, as you can see here there are wheels here and then there are motors to rotate each link ok. These are 5, 6, 7th link ok, the 6th motor is this one, the 4th motor is for this one ok. And this wheel is rotated by another motor and the whole idea is that we do not want it to skid. We want to slowly roll ok, these wheels will roll and drive this robot forward at the same time not hitting any obstacle.

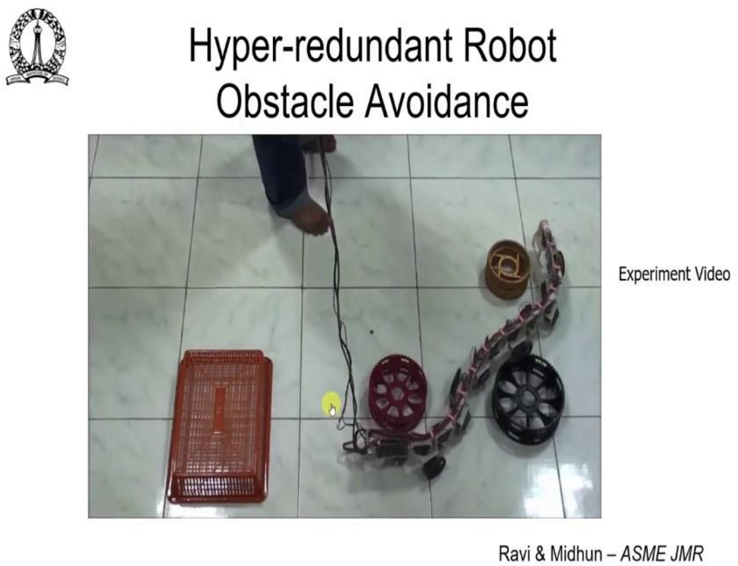
So, in the previous thing you saw that it was slipping a bit ok, sometimes quite a bit, but if you make it roll with this wheel then it can move forward much more smoothly ok. So, again the path is computed using tractrix and obstacle avoidance algorithm the commands to the joints and the wheels are given via cables ok.

(Refer Slide Time: 51:07)



So, here is a video of what this 12 link robot is capable of. So, this is one obstacle, this is another obstacle, these are 4 obstacles in this scene. It is on the floor also and this is the starting point of this robot you can see little bit of the head.

(Refer Slide Time: 51:26)

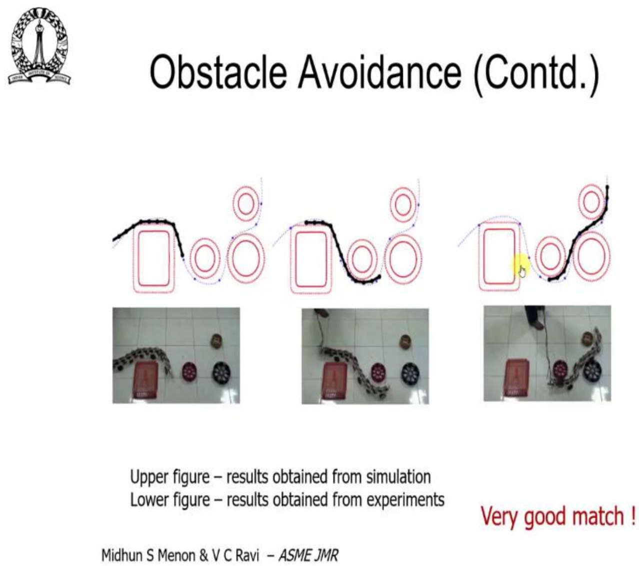


So, the trajectory of the head has been chosen such that in some parts. So, it follows what my cursor is doing, it is supposed to do that the commands to the motors are computed based on this tractrix based approach with obstacle avoidance, and then they are given by this cable, the power to this motors are also coming from this cable.

If the path of the head is pre chosen, if you had a sensor which could look ahead then that could also be computed by the robot itself. In an actual robot, hyper redundant robot we will have the sensors which will detect the obstacles in front of it and it will also have batteries and motors everything should be self contained ok. So, as you can see now that the head has cleared all the obstacles the head is moving straight, only the links which are close to this obstacle and bending they are rotating.

So, that is the power, the beauty of the tractrix based approach, only near the obstacle it will try to avoid only those links will rotate or joints will rotate to avoid the obstacle ok, elsewhere it looks like a smooth you know attenuating type of motion. So, because of the wheels and because you are slowly rotating the wheels there is not much slip like the previous case ok.

(Refer Slide Time: 53:24)

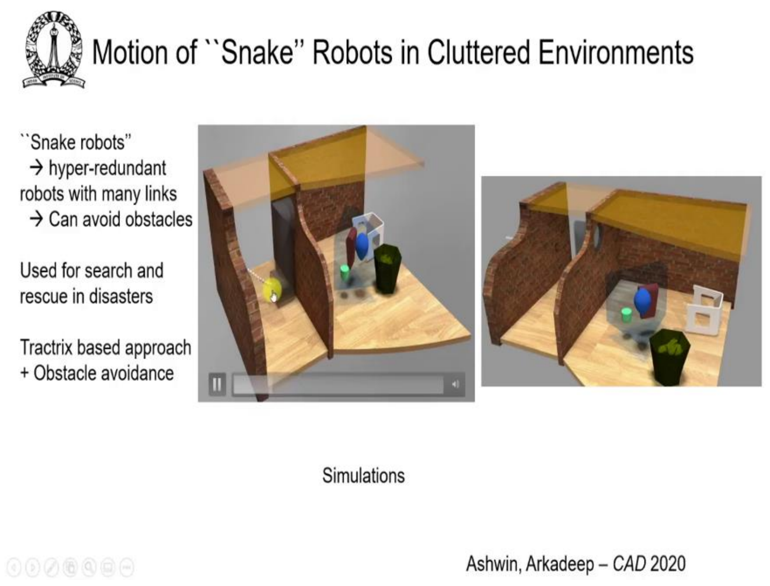


So, since we could do experiments, we can measure what is happening to each of this links ok, we can also do simulation. So, as I said we have the actual dark red is obstacle, we grow it by each link length such that even the center of the link is avoiding the obstacle, the rest of the link will also avoid the obstacle ok. So, these dotted lines are the grown obstacles a little bit more than what actually it is.

The blue line is the chosen path of the head ok, and then these are 3 pictures showing what is the simulated results for this 12 link robot ok. This is how the 12 links of the robot will look at some one instant, this is the second instant, and this is the third instant. We also can take the pictures at those instants. So, this is simulation, this is the actual picture taken at that instant.

Similarly, this is the picture taken at that instant when the robot should be at this place in simulation. So, what you can see is, the simulation results and the actual pictures are very close ok, these are very good match. So, this gives a lot of confidence in the algorithms and in the simulation results, that we have developed because it is more or less verified by actual experiments.

(Refer Slide Time: 55:00)

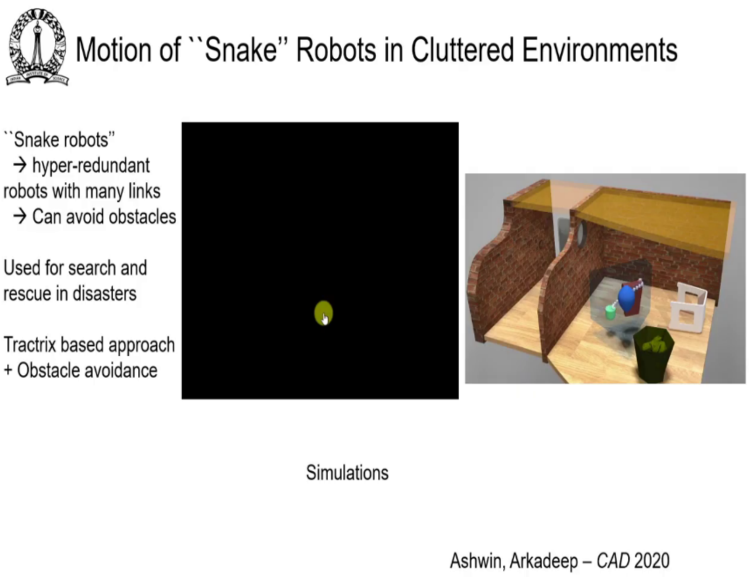


We can also extend this tractrix based approach to cluttered environment ok. So, that is the ultimate goal, we would like a snake robot to go into a disaster prone area like let us say an earthquake has occurred. So, the snake robot can go through all the gaps and cracks maybe it can carry a camera and then take pictures and see if there is any person living if there is any life still there ok. So, that is what I am going to show you here. So, this is a simulation results ok.

So, it is a snake robot with a hyper redundant robot with many links, it can avoid obstacles and it can be used for search and rescue operations in disaster prone areas or disaster areas. So, basically what we are using is the tractrix based approach and obstacle avoidance what I have discussed ok. So, here is a video of a synthetic generated scene, this white dotted thing is that robot ok.

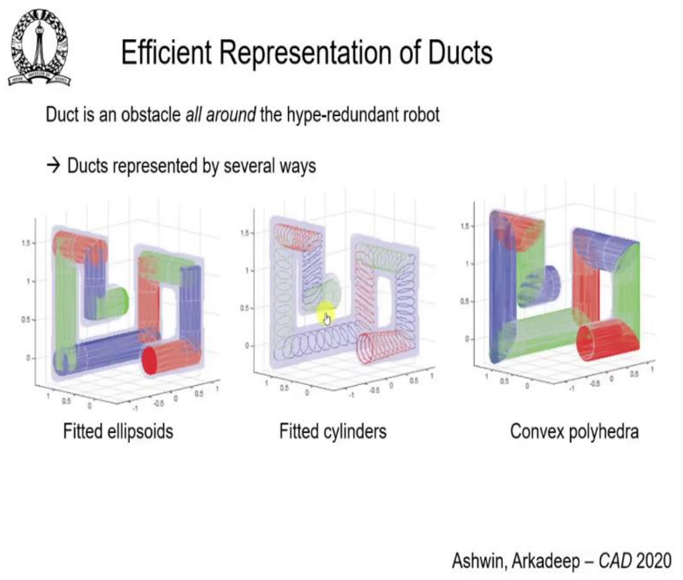
So, you can see it is moving in 3D space without, but this is a simulation. So, we can do all these things. So, here are 2 views; one is what it is doing, and this is one view of the same scene looking from some other angle. So, you can see it goes through some hole and then it will climb up it will go through another hole.

(Refer Slide Time: 56:28)



And then it will go around all these obstacles at the same time, making sure nothing is hitting any of the obstacles ok. So, you can think of this like a room and this robot is going through. Of course, this is the simulation you know this robot cannot float on air. So, it must be anchored or it must be moved on the ground, but never the less. This is an idea to show that we can simulate motions of this hyper redundant robot in cluttered environment.

(Refer Slide Time: 57:03)



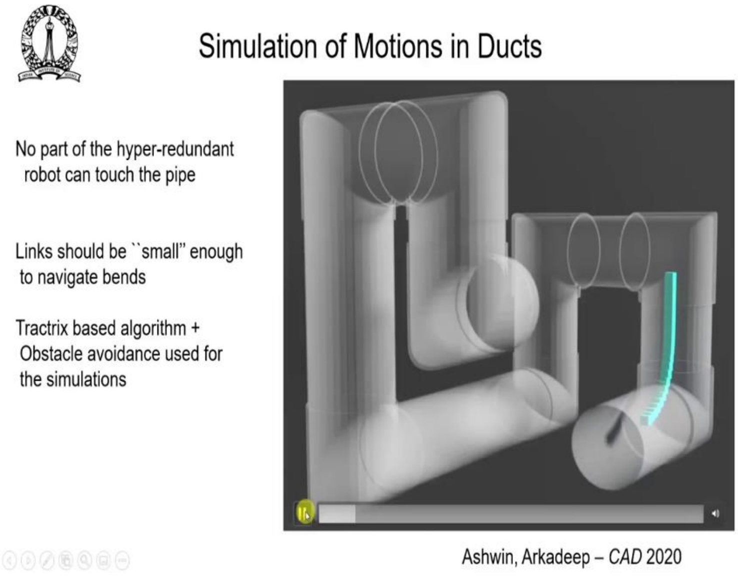
We can also simulate the motion of these robots in ducts ok. So, you can think of a pipeline and we want to inspect this pipeline from inside, we want to send this snake robot, such that it can go and inspect ok, to see if there are any cracks if something has happened in the pipeline. So, what is the difference between the previous and this in a duct actually the obstacle is all around you ok. So, in the case of the previous example the obstacles are on one side.

So, you can go along the normal ok, perpendicular to the direction of motion. However, in a duct the obstacle is all around you ok. So, then the path planning becomes more tricky. So, you cannot really move in a perpendicular direction, because you will hit some other wall of the duct. So, first thing is how do you represent ducts. So, there are various ways to represent ducts and this has come out in a paper recently in computer aided design.

So, we can have what are called as fitted ellipsoids. So, each so this is like a duct and same duct showed, how to represent in as fitted ellipsoid, fitted cylinders and has convex polyhedral ellipsoids, means in 3D objects. So, you have set of ellipsoids, which you connect each one of them you fit each one after another to represent this duct ok.

If you have a cylinder then these are some small cylinders, which you attach and make it into the duct and finally, we can have convex polyhedral. So, the whole idea is there are various ways to represent ducts and some are better than others.

(Refer Slide Time: 58:53)

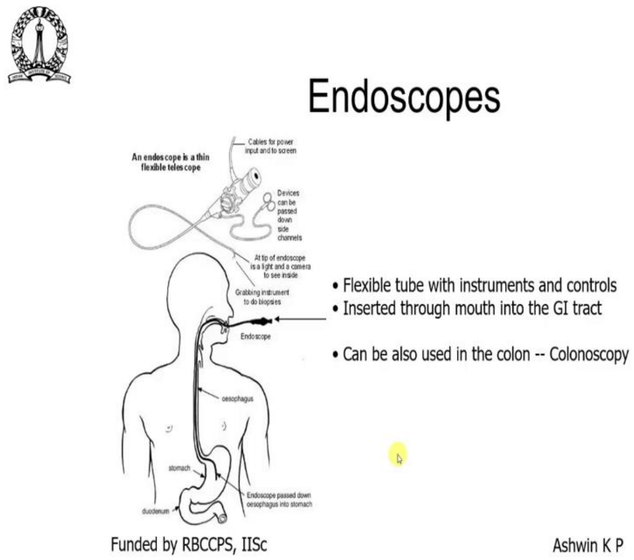


So, if I want to simulate the motion of a hyper redundant robot in that duct, which I showed you. The main constraint is that no part of the hyper redundant robot can touch the pipe or the duct. So, which basically means; that each link of this robot should be small enough, to navigate the bends.

So, if you have a very large link and if you have a bend and if it cannot navigate the bend, then it is not worthwhile. So, that is one constraint we have assumed, it is also using this tractrix based algorithm and obstacle avoidance for the simulations, ok. Again in this case you can see that the robot is sort of floating in here, but the basic idea is that I can rotate these joints and move this robot along this duct, such that it never hits any of the obstacles.

So, let us play it once more, you can see it is not hitting any of the sides of the pipe of the pipes.

(Refer Slide Time: 60:17)



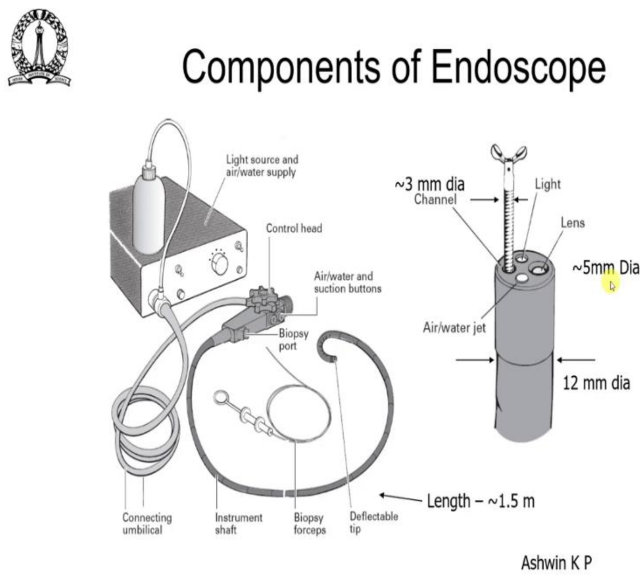
Finally, we can also simulate the motion of something called as an endoscope ok. A very quick introduction to what is an endoscope and what it is used for. So, basically an endoscope is a thin flexible pipe ok, at the end there is some camera and some lighting system. So, what it is used is it is sent through your mouth through the GI tract into your stomach and then you can take pictures.

So, if you have cancer or if you have ulcers ok, we can take pictures it can even take samples of these cancerous tissue or some ulcers and come out. So, it is inserted through the mouth of a person by a trained technician ok, then it is sent down it is you have to make sure that it will not go into the lungs. So, there is one part which there is a passage which goes into the lungs, because it is the same wind pipe ok.

And then you go and take some pictures and come out ok, the endoscope is basically a pipe like this it is a flexible pipe at the tip there is a camera to see inside ok, and then you can rotate the end by means of these knobs. So, as to look in different directions when you are inside the stomach ok, it can be also used in colonoscopy as in the colon for colonoscopy, but we are not going to go into that ok.

So, this work was funded by Robert Bosch Center for Cyber Physical systems and it was done by this PhD student Ashwin.

(Refer Slide Time: 62:01)

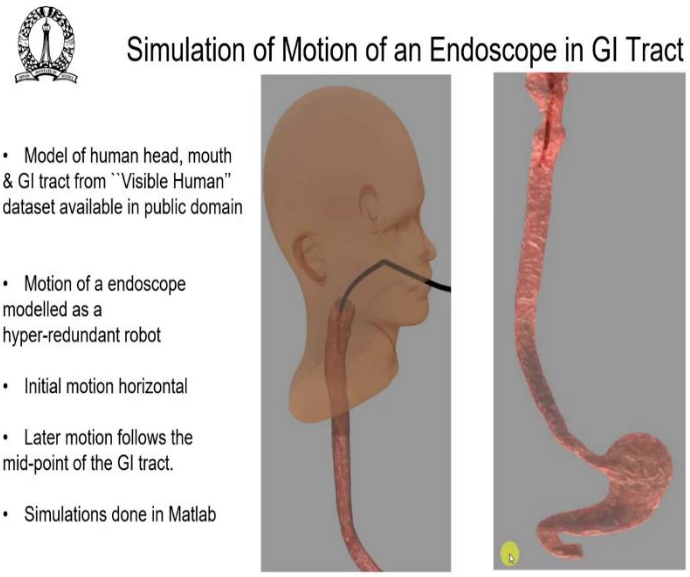


So, again quickly it is a typical components of an endoscope are this long flexible pipe which is about 1.5 meter long. There is a detachable tip and then there are these channels through which you can put in some wires these are force biopsy channels. So, you can put in this and at the end you can take out some small sample and then there are these buttons for air and water, because while you are going through there might be some food in your GI tract.

So, you can force it out and then there is a box which tells you how to control the air and water supply, it also gives a light source and then there are these two knobs which can rotate the end in two directions at the end it looks like this. So, this is roughly 12 mm diameter because the gap in your throat is around that.

So, you cannot force something through your throat because you will injure something ok. So, there is a biopsy channel where you can take this smalls graspers through it and you can take out some material, this is about 3 millimeter diameter there is a light, there is a lens and this is about 5 millimeter diameter ok. So, as you can see it is a very compact very sophisticated device, which you push through your GI tract and take pictures.

(Refer Slide Time: 63:32)

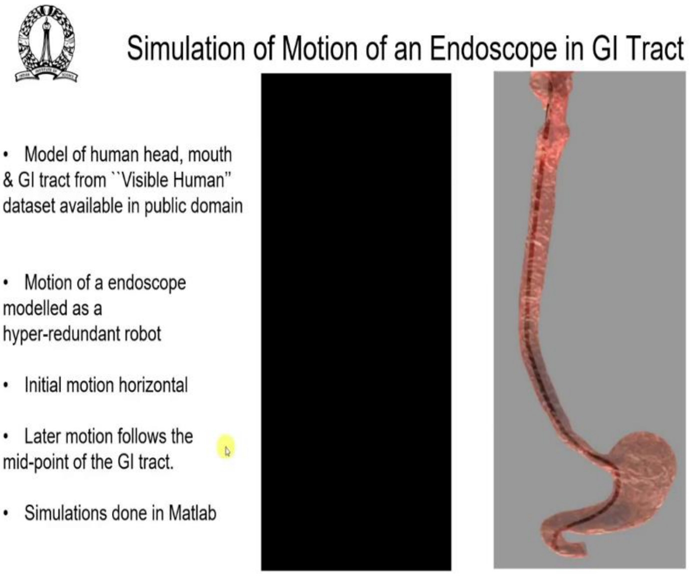


So, we wanted to show a simulated motion of this endoscope ok. So, this was a project when we were trying to show to a surgeon, what it what we can do first by simulation. So, that something like this can be used to train later on novice surgeons or novice technicians. So, first we need to model the human head, mouth and GI tract this is possible, because there is a data set called visible human ok. It is available in the public domain.

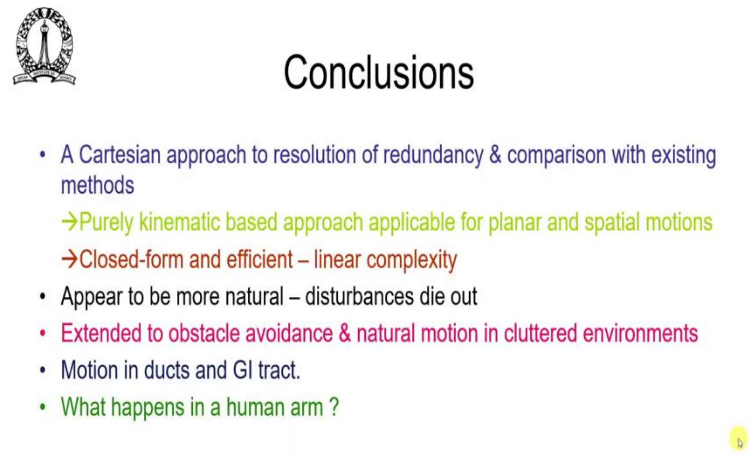
So, the motion of the endoscope is modeled as a hyper redundant robot. Now, we are good at modeling hyper redundant robot and we can pass it through a duct. So, this is like a very sophisticated duct ok, very complicated duct. The initial motion of the endoscope is horizontal and then it should follow the center line of this GI tract ok. So, it follows the midpoint and we did all these simulations in MATLAB.

So, I am going to show you these 2 movies, which were generated by the student ok. Where we show what is the motion of a simulated endoscope in the GI tract.

(Refer Slide Time: 64:57)



(Refer Slide Time: 65:23)



So, in conclusion a Cartesian based approach to resolve the redundancy and comparison with existing methods has been developed. It is a purely kinematic based approach applicable for planar and spatial motions. It is for closed form solutions of the tractrix -- uses the closed form solutions for the tractrix.

It has linear complexity, the motion appears to be natural and the disturbances die out we can extend the tractrix based approach to obstacle avoidance and natural motion in cluttered environments. We can show the motion in ducts and GI tract. In the next lecture we will see what happens in the human arm ok, the human arm is also a redundant system, it has more joints than required to position and orient an object. So, we will look at that.