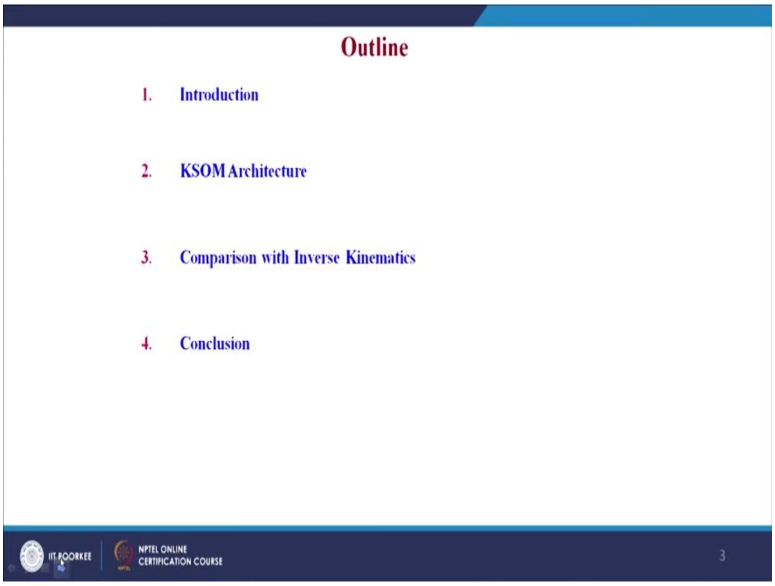
Prof. Felix Orlando
Department of Electrical Engineering
Indian Institute of Technology, Roorkee

Lecture – 29
Neural Control of a Hand Exoskeleton

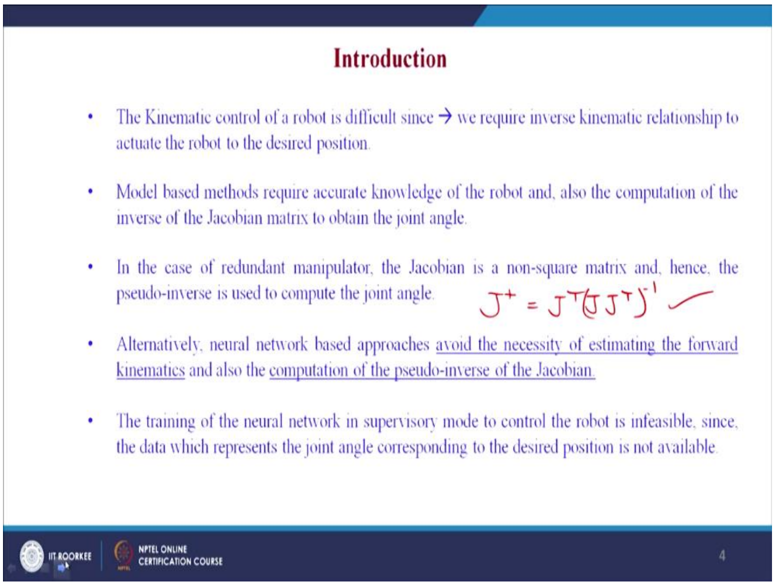
Good morning, today we are going to see the lecture on Neural Control of your Hand Exoskeleton. The outline of this lecture will be as follows; first we see the introduction why we go for neural control.

(Refer Slide Time: 00:38)



And then we see the Kohonen self-organizing map architecture and then we compare the results obtained from the Kohonen self-organized based mapping with that of the inverse kinematics; then we finally, conclude the remarks.

(Refer Slide Time: 00:51)

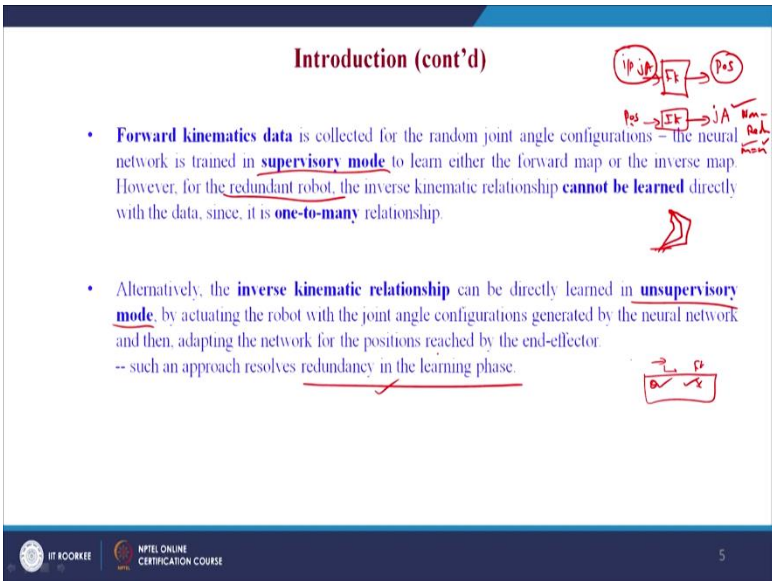


Now, coming to the introduction; the Kinematic control of a robotic arm is difficult since we require the inverse kinematics relationship between the joint space and the Cartesian space through the inverse of the Jacobian matrix. Model based systems or methods require accurate knowledge of the robot and, also the computation of the inverse of the Jacobian matrix to obtain the joint angle. In the case of a redundant manipulator, the Jacobian is a non-square matrix and hence the pseudo-inverse of the Jacobian matrix is required instead of inverse.

So, pseudo inverse is given by $J^+ = J^T(J^TJ)^{-1}$; J transpose right inverse. So, this is the generalized pseudo inverse if you take if the manipulators degrees of freedom is greater than that of the coordinates of the Cartesian space. So, alternatively neural network based approaches avoid the necessity of estimating the follower kinematics and also the computation of the pseudo inverse of the Jacobian matrix.

Now, the training of the neural network in supervisory mode to control the robot is infeasible, since the data which represents joint angle corresponding to the desired position is not available. Due to that reason supervisory mode is a infeasible methodology, because of the non availability of the data which corresponds to the desired the position.

(Refer Slide Time: 02:15)



So, basically we can do from the forward kinematics like for example; if we take the forward kinematic relationship that is give the joint angles varying the joint angle 1 from 0 to its limit and also joint two up to joint n and obtain the forward kinematics unique solution. Then we do the reverse in order to train the system that is we do the reverse data; the system is trained with the data and you reverse it, reverse give the joint angle and obtain the position that is the forward kinematics.

And then you give the corresponding joint angular, corresponding tip position as input and then try to obtain the joint angles corresponding to the tip position you have given. But in this case there will be possibility to obtain the accurate result, because the system is non redundant; that means, you have the neural network you give input joint angle and you obtain the position of the end effector, this is the forward kinematics.

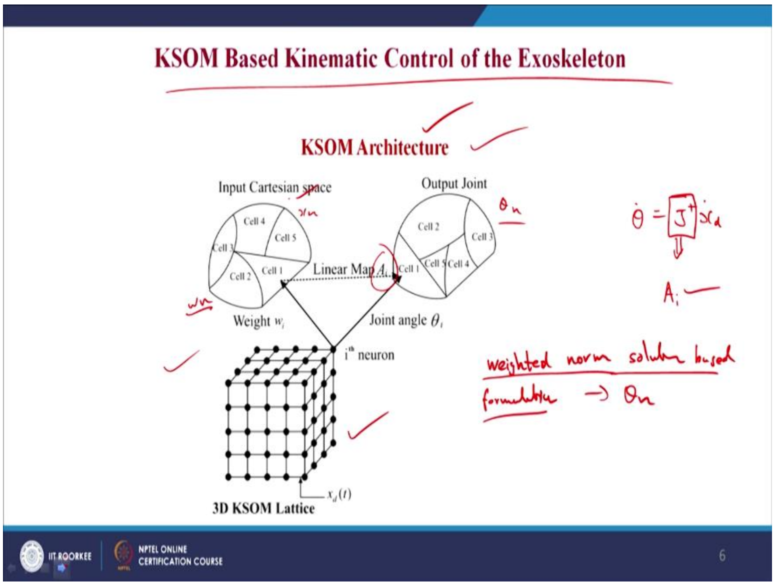
So, you take this data and take this data. Now, for the inverse kinematics, we feed this position data and try to obtain the joint angle as the output; but this works well provided the system is a non-redundant system, where m equal to n, that is the Cartesian space coordinates is equal to the number of degrees of freedom of the system.

But in the case of redundant robotic system, the inverse kinematic relationship cannot be learned directly with this data; because we have one to many solutions. That means, if you give this position, for this position you may get different solutions possible; that is the joint angles for this tip position will be many that is one to many relationships. In this case the system network gets confused; thus supervisory mode will not be feasible to do the inverse kinematic control.

An alternative method is that, the inverse kinematics relationship can be directly learned in unsupervised mode of learning, by actuating the robot with the joint configurations generated by the neural network and then, adapting the network for the positions reached by the end effector; such an approach resolves the redundancy in the learning phase.

So, what the system is generating to that output, we train the network and then by this method given the desired position we get the joint angles; according to this joint angle we can get the forward kinematics, so we have theta and the x. So, this with this relationship we can train the network, so that the system the network gets adapt to this type of relationship x and theta; with this approach we can able to resolve the redundancy in the learning phase; how we are going to that we are going to see in this coming lecture.

(Refer Slide Time: 06:02)



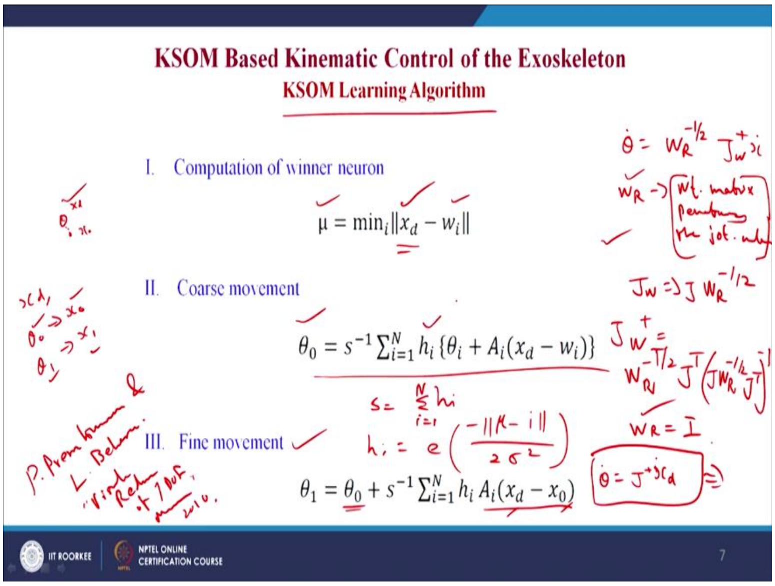
So, now, we see Kohonen self-organizing based kinematic control of the exoskeleton. First we see the Kohonen's self-organizing map architecture; the lattice is given here that is 3 D Kohonen self-organizing map lattice. So, the systems perfect feature is, the main important feature of this case of learning is its topology, the arrangement of the neurons in 2 D as well as 3 D structure.

So, each neuron is discretized into the input Cartesian space and its corresponding output joint angular space. The input space is mapped in a linear manner by A_i, which is the approximation of the inverse of the Jacobian matrix; because in general theta dot is equal to J pseudo inverse x d dot. So, this mapping is done by the pseudo inverse of the Jacobian.

So, here in our KSOM based learning approach, we are approximating this Jacobian inverse by the matrix A_i in our KSOM based learning. So, here the system is in such a way that each neuron is arranged in the lattice is associated with the input space which is categorized by the Cartesian space; and the output space which is generalized by the coordinates θ_n. Input space it is x_n; which is basically the weight vector w_n, which corresponds to the weight vector.

And thus we can say that input space is discretized by the vector w_n and the output space is discretized by the vector θ_n. And a linear map connects them which is given by A_n; this A_n represents the inverse Jacobian that is the inverse Jacobian. Jacobian pseudo inverse or inverse is approximated by this linear map which is A_n. And the KSOM based inverse kinematic control scheme uses weighted norm; weighted norm solution based formulation, this approach utilizes or uses the weighted norm solution based formulation to compute the joint angle corresponding to the given the position in the workspace.

(Refer Slide Time: 09:13)



So, here the learning is done in order to obtain the weighted norm solution is given by

$$\dot{\theta} = W_R^{-\frac{1}{2}} J_w^+ \dot{x}$$

where W_R is nothing, but the weight matrix penalizing the joint motion to achieve the desired task that is desired a secondary task. What you want to achieve as a secondary sub task is obtained by this weight matrix and we have the matrix J_W which is given by $JW_R^{-\frac{1}{2}}$. Accordingly,

$$J_w^+ = W_R^{-\frac{T}{2}} J^T (JW_R^{-\frac{1}{2}} J^T)^{-1}$$

So, the weighted norm formulation gives minimum norm solution when $W_R = I$; that means, when $W_R = I$ that gives a solution which is given by. If this is the solution then it means that; $W_R = I$ this is corresponds to the minimum norm solution. The detailed case of algorithm is given by the reference where the authors are P Prem Kumar and L Behera that is in, they have the visuals are going redundancy of robotic manipulator; they have made visual redundancy of 7 degrees of freedom manipulator that is in 2010. In robotics and autonomous system they have published this paper, the detailed approach is given in that paper with the stability analysis.

So, now coming to the learning algorithm based on KSOM. First the winner neuron is selected by given the desired tip position based on the lattice of the KSOM. A winner neuron is selected which has the minimum distance with respect to the given decide trajectory. So, the winner each neuron is taken and it is position with respect to the desired trajectory, desired given position is considered. So, which neuron is having a minimum discrepancy between the desired one and its position that is considered as a winner neuron.

From the winner neuron coarse motion or coarse movement of the robot manipulator is computed; which is given by this expression that is for coarse movement. Given the position that theta naught that is a coarse movement will bring the system close to the tip, close to the desired tip x_d. For $x_d \theta_0$ will bring x_0, from x_0 we are going to do the fine moment which depends on the x_0 that is given by

$$\theta_1 = \theta_0 + S^{-1} \sum_{i=1}^{N} h_i A_i (x_d - x_0)$$

that will bring the system much much closer finally, to the desired position.

Thus for a given x_d, we have now θ_0 corresponds to x_0 by forward kinematics and θ_1 fine motion which corresponds to x_1. So, we have this data now and we are going to make the system get adapted to this pattern of input and output, Where here the function

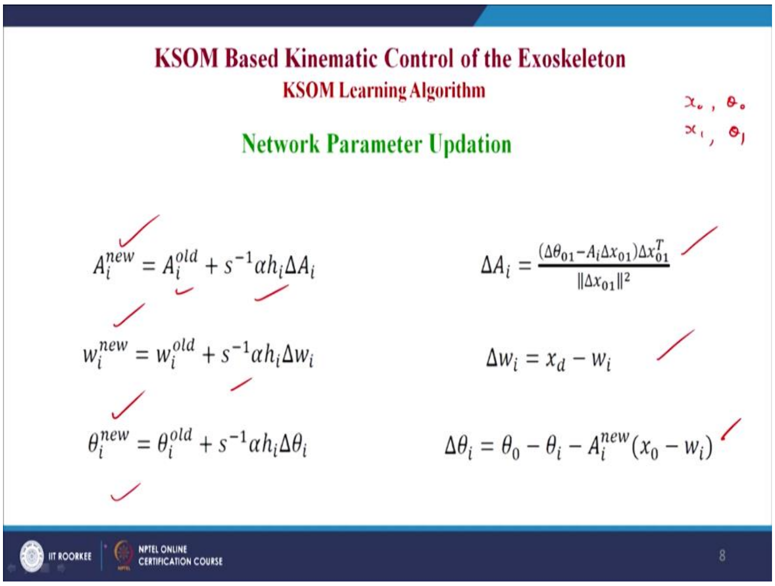
$$s = \sum_{i=1}^{N} h_i$$

where

$$h_i = e^{\left(\frac{-\|\mu - i\|}{2\sigma^2}\right)}$$

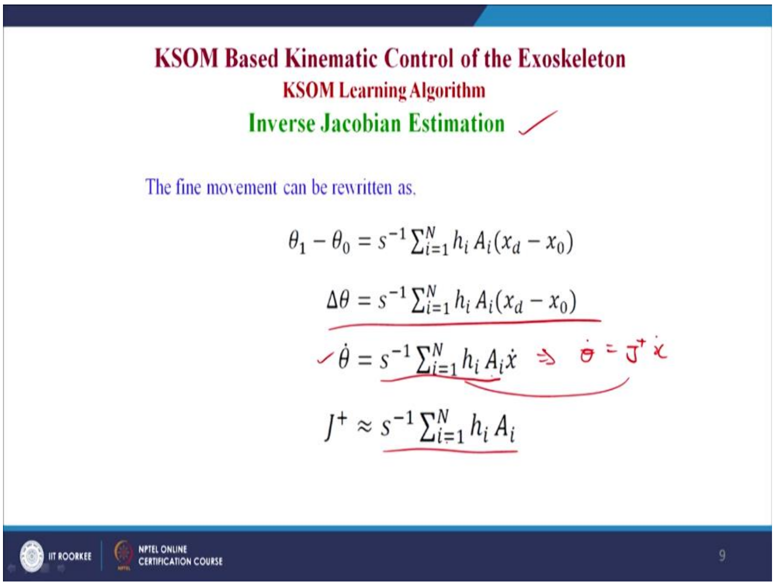
this is the representation for h_i and s, that is the case.

(Refer Slide Time: 14:13)



And now after getting the two data; which is $x_0\theta_0$ and x_1 θ_1, we are going to adapt the make the system get adapted to this data by this network adaptation formula. Where A_i is getting adapted with A_i^{old} and W_i^{new}, the new weight update is done by this; and accordingly the θ_i^{new}, is getting updated with this weight update rule.

(Refer Slide Time: 14:48)



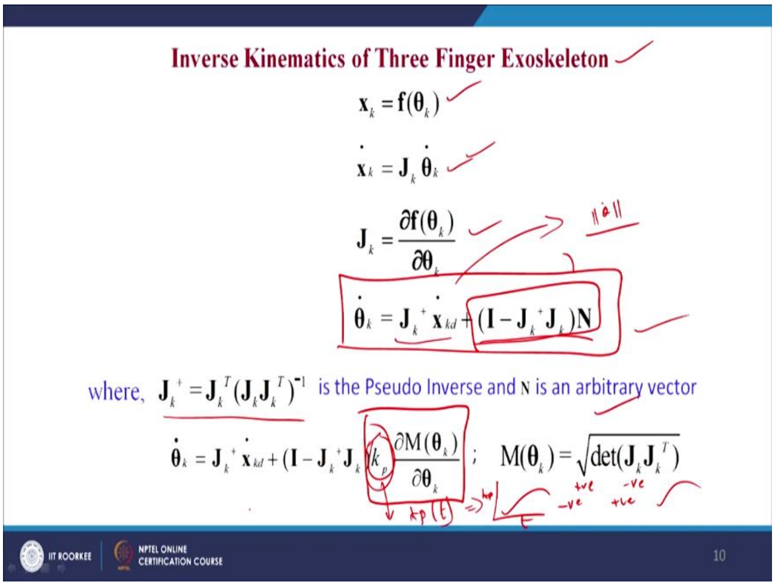
And now coming to the inverse Jacobian approximation; the inverse Jacobian approximation is done in such a way that taking the fine moment, we rewrite it by $\theta_1 - \theta_0$. Bringing the θ_0 on the right hand side to the left hand side, we have the difference $\Delta\theta$ is given by

$$= S^{-1}\sum_{i=1}^{N} h_i A_i (x_d - x_0)$$

And due to which the $\Delta\theta$ can be written as $\dot{\theta} = S^{-1}\sum_{\iota=1}^{N} h_\iota A_\iota \dot{x}$

Thus, $J^{+} \approx S^{-1}\sum_{i=1}^{N} h_i A_i$

(Refer Slide Time: 15:48)



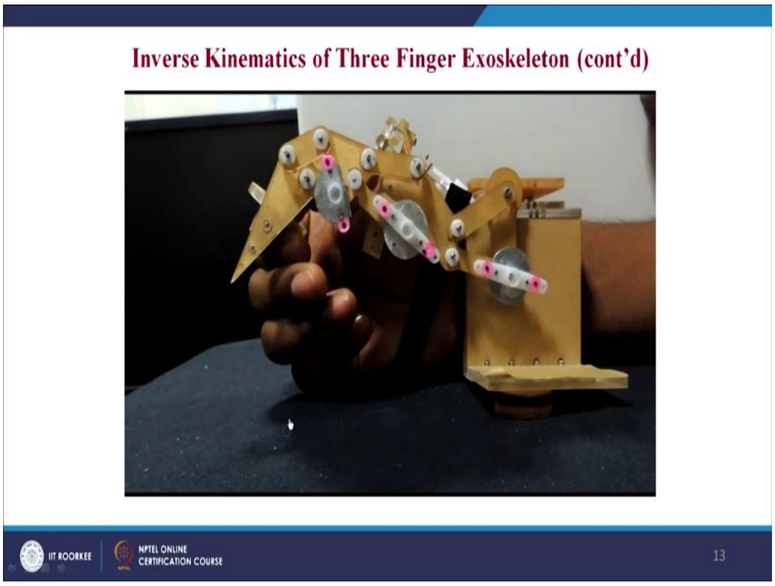
Thus we have approximated the Jacobian inverse by the KSOM learning approach under unsupervised technique. And now we compare that approach with that of the inverse kinematics of the same three finger exoskeleton or hand exoskeleton. So, the inverse kinematic approach for redundancy resolution is given by this technique; which is a forward kinematics $X_k = f(\theta_k)$ and the differential kinematics is given by $\dot{x}_k = J\dot{\theta}_k$. And the Jacobian matrix is the derivative matrix which is given by the partial derivative of the forward kinematics equation with respect to the joint variable.

And the generalized solution for this differential equation is given by $\dot{\theta}_k$ equal to $\dot{\theta}_k = J^+\dot{x}_{kd} + (I - J^+J)N$. Where the first term is the one that gives the solution which is corresponding to the minimum norm solution; whereas, the second term considers the redundancy present in the robotic system. So, which is basically the mapping from the space to the null space of the Jacobian.

So, the null space is now getting occupied by this term, this vector. So, due to which the null space of the Jacobian matrix is now filled with a certain domain, so that that part will be utilized in the joint space of the Jacobian matrix in order to utilize the redundancy of the system. Where pseudo inverse is given by the right inverse which is given by $J^TJ; J^{T^{-1}}$; and N is an arbitrary vector which involves the instantaneous optimization of a performance criterion.

In our case it is taken as the manipulative measures instantaneous optimization, which is we are maximizing with the positive scalar with the first derivative. The derivative of the manipulative measure which is given by a root of determent of J J transpose of the manipulative, because it is a redundant; if it is non-redundant, it is determinant of J which is the manipulability.

(Refer Slide Time: 18:17)

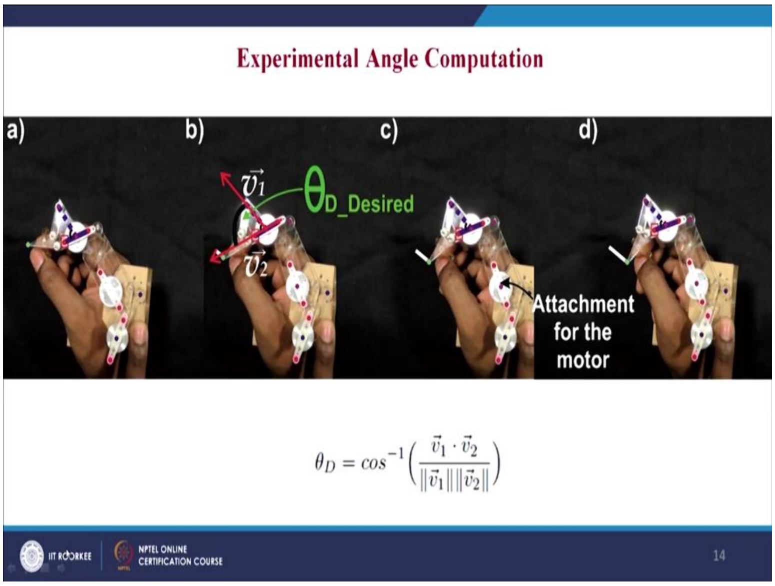


And now we have taken this video; that is the fingertip trajectory is to trace the straight line. So, from this straight line trajectory, we have made the system to trace this given decided straight line trajectory. I repeat again, rest to initial posture, from initial posture to a straight line final destination. So, this straight line posture trajectory of, in the Cartesian space will provide or will make the system to have redundancy in the joint angular solution. So, how we are going to do this type of redundancy resolution for under the KSOM compared with that of the generalized inverse kinematics approach.

Similarly, we have done for the middle finger exoskeleton; the previous was the index finger exoskeleton, because it is a three finger exoskeleton. We are done for first for the index figure, now for the middle finger; and then for the thumb. Because as you can see that, the actuators are not own or attached to the exoskeleton, which makes that system in such a way that, the exoskeleton now without the actuator acts as a slave; whereas, the human finger or a hand acts as a master.

So, the trajectory is now given by the human healthy human hand, so that the exoskeleton follows it in order to take or derive the desired tip trajectory. So, that can be fed to the inverse kinematics approach as well as the Kohonen self-organizing based neural control.

(Refer Slide Time: 20:05)

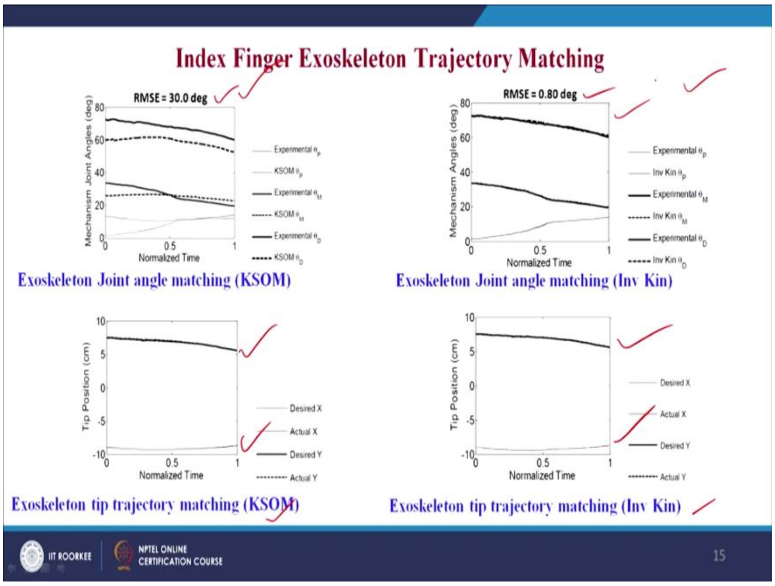


So, now after doing that, we also computed the desired joint angle in such a way that it is given by this approach; this is what the marker just let me get back. So, the marker 1, you see this the each strip in the motor or actuator attachment has two markers which will form a vector. From the image we could able to obtain the joint angle that is the desired joint angle, so that can be given to the desired joint angle for the Kohonen self-organizing map also for the inverse kinematic approach in order to have the redundancy parameter being an optimal trajectory.

That is k_p in our case we have taken this as a spline; a cubic spline in such a way that, this optimally varies with respect to time, so that this can take a value which is different which is changing not only in magnitude, but also in it is sign. It can take from negative to positive or positive to negative and the magnitude varies also, so that we do not have to maintain this as a positive scalar. Instead we made it as a variable that varies in sign as well as in its magnitude.

So, to obtain this one, we are instantaneously varying the or optimizing the manipulability measure and also we have to have the discrepancy between the desired joint angle that we obtained from this videos of this real human angle by the exoskeleton. And that angle is the desired angle, between this desired angle and the inverse kinematics angle to be minimized and that minimized one will provide the optimal k_p that is a redundancy trajectory. So, the angle is given consider, the angle joint angle decide joint angle θ_d is computed in this way that is the cos angle between these two vectors V 1 and V 2.

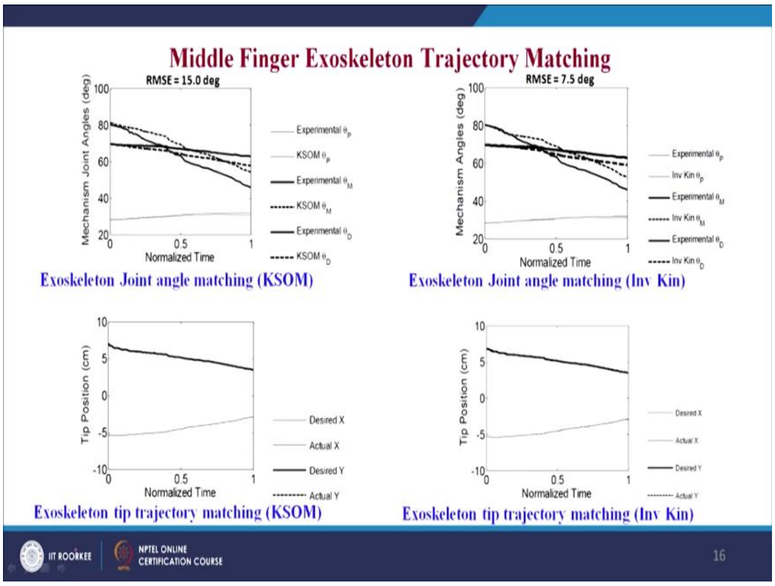
(Refer Slide Time: 22:38)



So, this is how the joint angle desired one is computed that has been given as the computation angle in order to have the root mean square error to be minimized. So, we have done that and compare the results now between the; joint angles obtained from KSOM based joint angle joint control of the exoskeleton. And also we have done the inverse kinematics based redundancy resolution with the redundancy parameter being optimal redundancy parameter. So, we have comfort that for the index finger exoskeleton; the tip trajectory is accurately mapping in the case of both KSOM as well as in the inverse kinematic based approach.

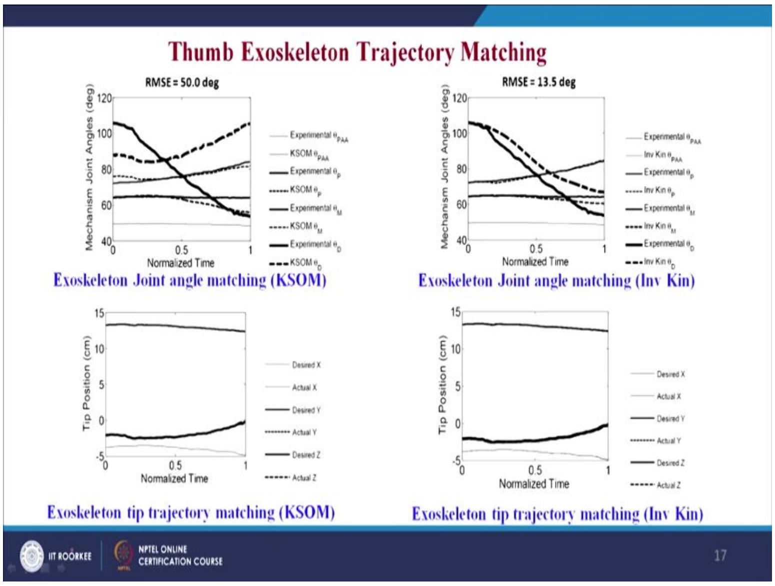
But the joint angular trajectory is very good matching obtained in the inverse kinematics based approach; whereas, the redundancy resolution done through KSOM based approach gives the root mean square error 30 degree. Because we are supposed to include manipulability measure; whereas, what we have obtained is the weighted norm solution. So, because we obtain the weighted norm solution from the KSOM based approach; that is why the error here root mean square error is larger compared to that of the error obtained from inverse kinematics based approach. This is for the index finger exoskeleton.

(Refer Slide Time: 24:04)



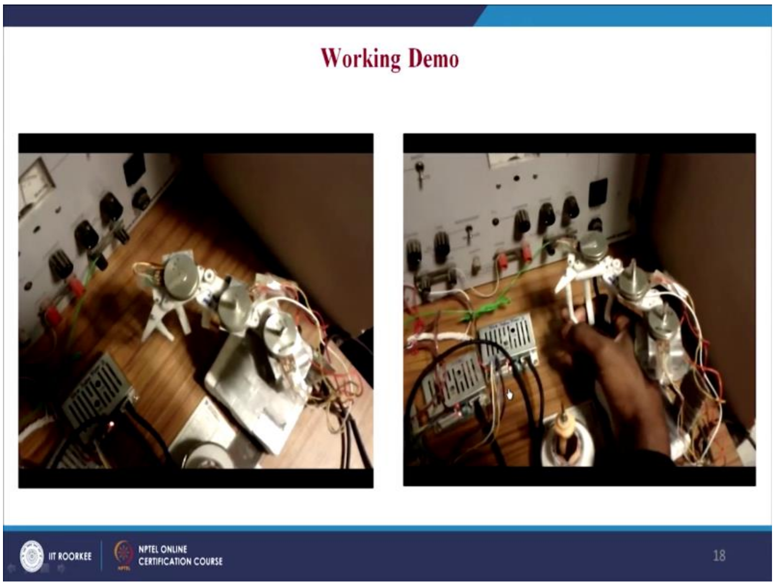
Then for the middle finger we have nearly 50 percentages greater than that of the root mean square error approach obtained from the inverse kinematics approach. Thus it is 7.5 degrees root mean square error with inverse kinematics approach for the middle finger exoskeleton and it is 15 degrees for KSOM based approach. But the tip trajectories are perfectly matching, as it is the primary sub task to track the given desired tip trajectory.

(Refer Slide Time: 24:32)



Similarly, for the thumb exoskeleton we have the root mean square error being 50 degrees for the KSOM based approach. And then for the root mean square error, for the inverse kinematics based approach it is 13.5 degrees; whereas, the tip trajectory accurate matching is given here.

(Refer Slide Time: 24:53)

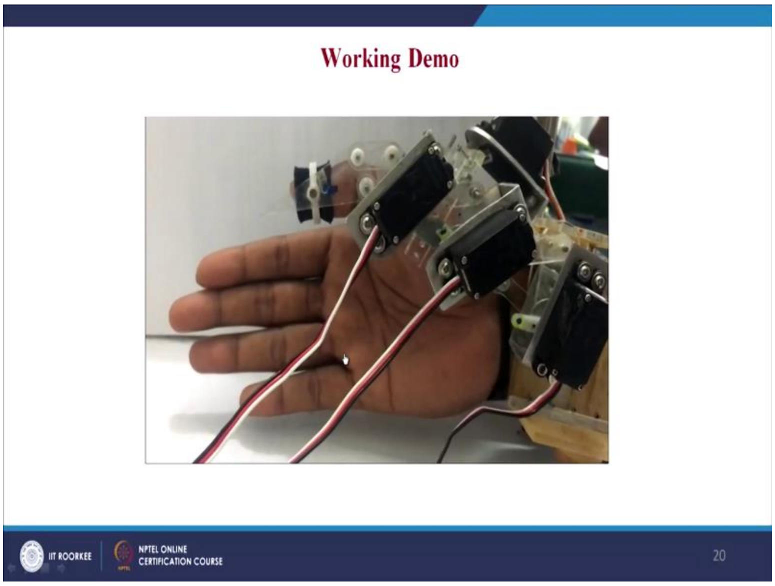


Now, coming to the working demo we did it for the index finger exoskeleton based on KSOM in order to trace the desired trajectory; this is straightening trajectory and this is without the human hand and this is with the human hand, so that straightened territory is clearly observed. What is observed here is the, exoskeleton now accessing master that makes the slave human finger to trace the straight line trajectory.

This is a first prototype; what we are having here is the, ultrasonic actuators that is having a good weight to good power to weight ratio; whereas, the latest model what we have made is with DC servo motors, so that the cost of the system can be reduced. Because with this ultrasound system, with this ultrasonic system we have the actuator cost being very heavy and hence very costly; and hence the expenditure of this system is very high.

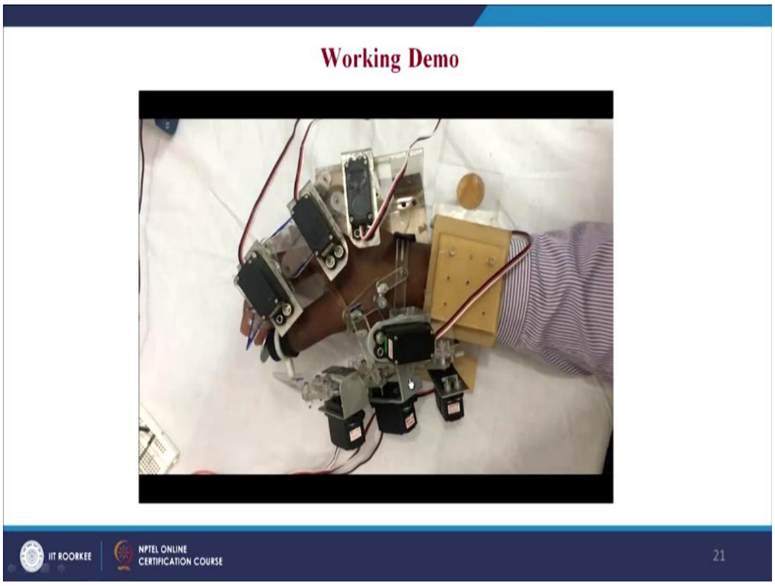
In order to reduce the cost of the system, overall system we have replaced the ultrasonic motors with that of the simple DC servo motors, so that the required torque of 5.2-kilogram centimeter is obtained with these motors as well. And the demo is, the working demo in tracing the straight line trajectory is shown here.

(Refer Slide Time: 26:39)



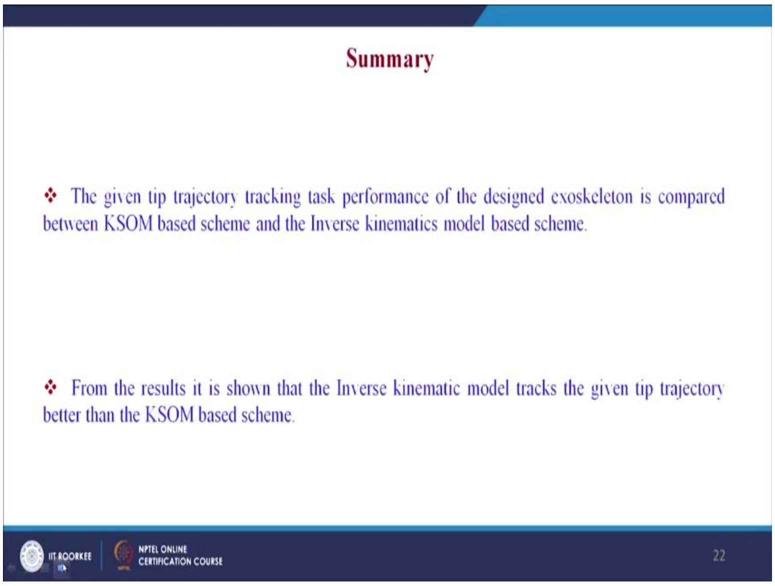
Similarly, for the thumb this is the way, because once the more actuators are attached to the exoskeleton, now the exoskeleton cannot be acting as a slave. Because the exoskeletons are getting moved by the actuation of the actuators and it cannot be freely moved by the human hand. And thus with this motors attached to the exoskeleton, the exoskeleton becomes a master; and the human finger has to get traced by the trajectory provided by the master system.

(Refer Slide Time: 27:16)



Now, this is the demo which shows the coin tracing one, so that you can make the coin to move in a cylinder straight line trajectory.

(Refer Slide Time: 27:26)



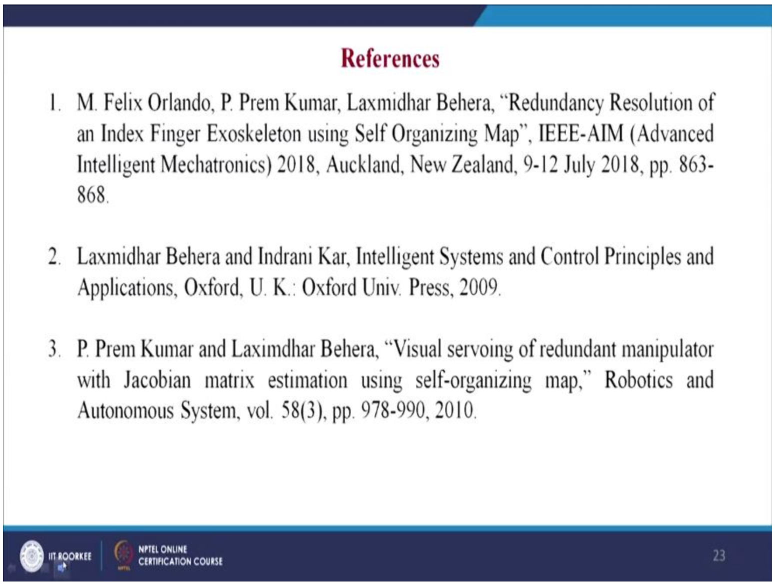
Now, coming to the summary of this lecture it is stated; that in the starting we have seen what is the significance of unsupervised learning. Because the simple neural network based supervised learning cannot be useful in order to resolve the redundancy; whereas, unsupervised learning can be helpful or useful to resolve the redundancy associated with the manipulators.

And from that we conclude that because of the comparison with that of the inverse kinematics approach with the optimal redundancy parameter. We have observed that, at the given tip trajectory tasks can be, the performance of the designed exoskeleton is comparable between the KSOM based scheme and that of the inverse kinematics based scheme.

But the for all the three exoskeletons that is the index finger, thumb and the middle finger what you have observed is; the root mean square error that comes from the KSOM is significantly higher than that of the inverse kinematics based approach. And the reason is, in the KSOM based approach we did not include the instantaneous optimization approach of manipulability measure.

What we have done is; we have done the weighted norm based minimum solution under the KSOM based approach; that is why the significant difference happens here. And from the results it is shown that the inverse kinematic based trajectory gives a better result than that of the KSOM based scheme.

(Refer Slide Time: 29:06)



And the references what we have followed for this lecture are basically from the paper by authors Prem Kumar and professor Lakshmidhar Behera; that is the title of the paper is visual servoing of redundant manipulator with Jacobian matrix estimation using self-organizing map. That has been published in robotics and autonomous system in LC where the volume is 53, the issue number is 3.

And the next paper followed from this paper is the recent paper which is published in a conference iterable advanced intelligent mechatronics conference in 2018; which we are focused only in the inverse that is the exoskeleton part of the inverse of the index finger exoskeleton. And the KSOM based topology approach or architecture has been studied from this book, which is by Professor Laxmidhar Behera and Indrani Kher. The book entitled as intelligent systems and control principles and applications.

Thank you, with this we wind up this lecture.