Intelligent Systems and Control

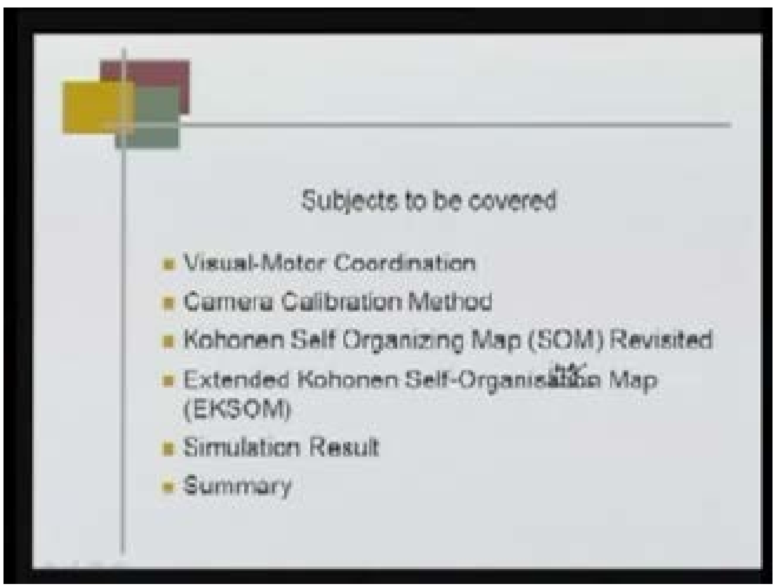
Prof. Laxmidhar Behera
Department of Electrical Engineering
Indian Institute of Technology, Madras

Module - 3 Lecture - 7

Visual Motor Coordination using KSOM

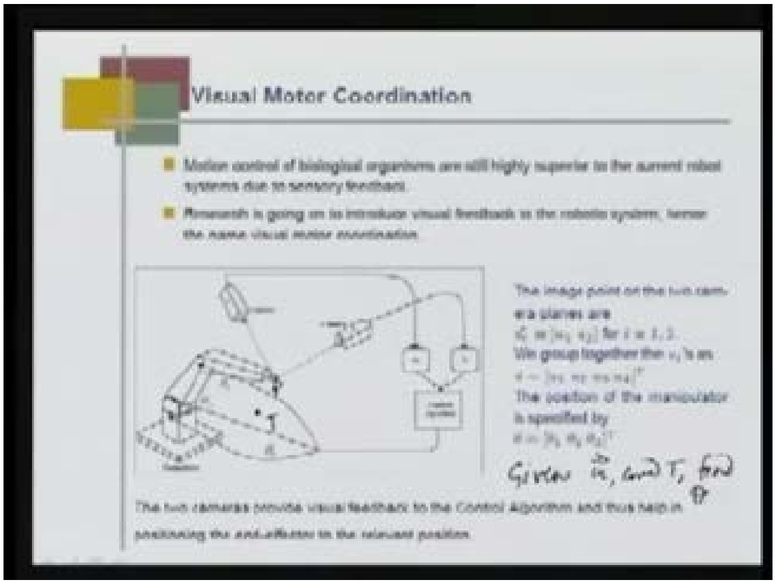
Our topic today is Visual motor coordination using Kohonen self organizing map; a little difference to what we have been talking on intelligent control, specifically neural control. We have been doing dynamic control but today, we will be learning how we can learn the inverse kinematics of a robot manipulator simply by learning. This is the seventh lecture on neural control.

(Refer Slide Time: 00:57)



Subjects that we will be covering today: Visual motor coordination problem, Camera calibration method, Kohonen self organizing map, Extended Kohonen self organizing map, Simulation results and summary.

(Refer Slide Time: 01: 22)



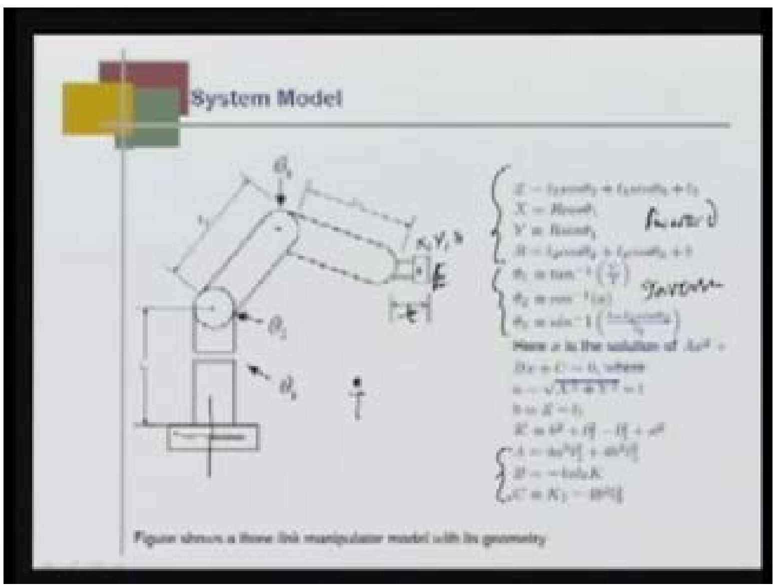
Visual motor coordination motion control of biological organisms are still highly superior to the current robot systems not only due to sensitive feedback but, also very improved information processing capability of human brain. Many things also are not known about how information processing takes place in the brain. Research is going on to introduce visual feedback to the robotic system; hence the name visual motor coordination.

What you are seeing here is a robot manipulator. It is a three link robot manipulator. Two cameras are attached here camera 1, here camera 2. Camera 1 gives the input. Now camera looks at the work space and gives the information about the target point as well as the end-effector point u_1 and u_2. This is the control algorithm and from control algorithm gives the command what should be $theta_{1,}$ $theta_{2,}$ $theta_3$ such that, the end-effector reaches any point in the work space. For example, this is my target point t and this is my end-effector point p. The p should match t by proper coordination $theta_{1,}$ $theta_{2,}$ $theta_3$ that has to be actuated such that, the end-effector finally comes to the t and this coordination takes place through visual feedback.

So, just like given an object, how I go and catch it? Now for example: this tip, I can touch the tip in this way I can touch the tip in this way. So, varieties of ways I can reach this point through visual feedback. This is called hand eye coordination. The image points on the two camera planes are u_1 and u_2 we group together the u_i's as one vector that is u vector u_1 u_2 u_3 u_4. You see although the object is in 3-D camera always gives the 2-D information.

The position of the manipulator is specified by $theta_1$ $theta_2$ $theta_3$. The two cameras provide visual feedback to the control algorithm (04:11) to the relevant position. Obviously, what we are interested is given u vector and the target vector t; find theta in steps such that, the end point reaches the target point. This is called visual motor coordination.

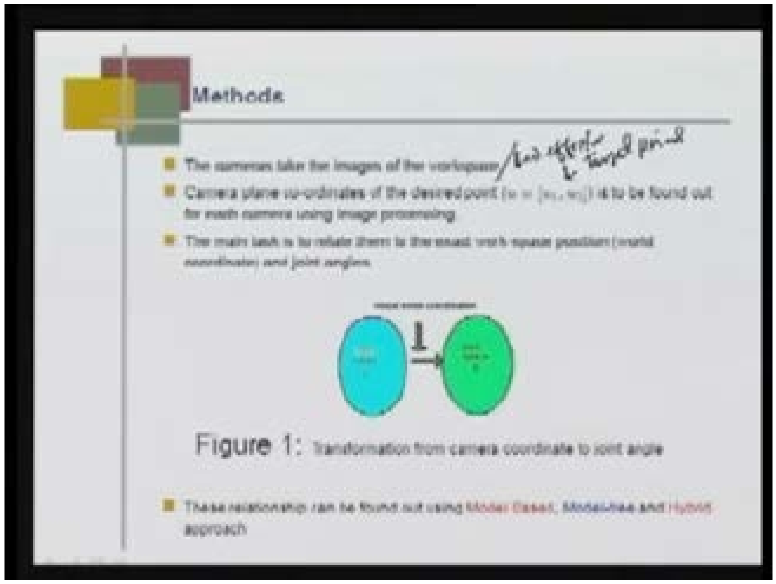
(Refer Slide Time: 04:52)



The system model is more accurate than what we said this base which we will say first link, this is the second link and the third link. The base moment is circular that is $theta_1$ and $theta_2$ up and down and $theta_3$ also up and down in the vertical plane and this rotates in the horizontal plane. If you look at the relationship any point, say this point its coordinate is z x and y with respect to some main reference frame. This z x y I mean, this reference is in this form from this point from the base z x y and the z x y is for example: given $theta_1$ $theta_2$ $theta_3$, the end-effector has a specific position z x y, so this is my end-effector point.

A special position is x y and z this x y z means this end-effector has a position in the space in the special coordinates x y z that can be correlated with theta_1 theta_2 theta_3 in this particular manner. z is l_2 sine theta_2 l_2 is the length of the second link plus l_3 sine theta_3 here which is the length of the third link sine theta_3 plus l_1, l_1 is the height. Similarly, the x position from here is fixed; this does not move; this simply rotates. Obviously, the x is r cos theta_1 where, r is l_2 cos theta_2 plus l_3 cos theta_3 plus t and t is this distance t. Similarly theta_1 is tan inverse y upon x. This is the forward kinematics that is given, theta_1 theta_2 $\text{theta}_{3.}$ How do I compute the end-effector position in the special coordinate in the space x y z? Now, from this equation I derive, given x y z what should be theta_1 theta_2 and theta_3? This is called inverse kinematics. That is, theta_1 is tan inverse y upon x; you can easily verify this theta_2 is cos inverse small x, where x is the solution of a x square plus b x plus c equal to 0 where a b c are given by this formulas and theta_3 is sine inverse b minus l_2 sine theta_2 by l_3. In this the three things that you need are a b and k you see that, you need here a b and k a, is given by x square plus y square root over this no time. This time is this from third link the end-effector as certain distance certain length. The end-effector link length is the t. This is forward kinematics and this is inverse kinematics. There is nothing to be solved. Mathematically, if I know this equation, if I know these parameters perfectly, inverse kinematics is solved. There is nothing to be done here. We are not interested in the mathematical way of computing inverse kinematics what we are interested is given a target point how do I learn step by step based on visual feedback such that, the error between this point and this point, this target point end point and target point, they match or this target point comes here. This is the problem that we are going to solve.

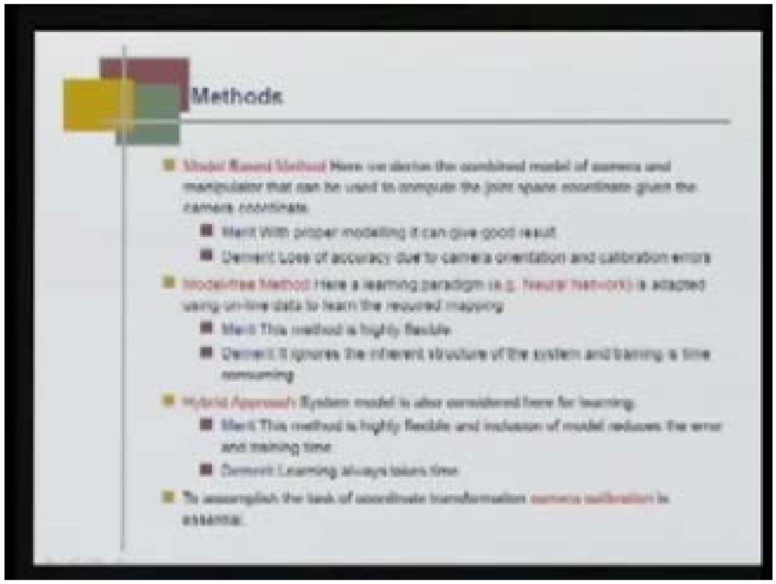
(Refer Slide Time: 09:49)



Methods: The cameras take the measures of the work space. Work space means it takes the image of end-effector it locates, where is the end-effector and the target point. Cameras give us the information where is the end-effector at the moment and where the target point is or how far is target point from the end-effector. Camera plane coordinates of the desired point u is to be found out for each camera using image processing. We exactly find out what is the target position and the end-effector position from camera in terms of some vector quantity u.

The main task is, to relate them to exact workspace position world coordinate and joint angles. Given this camera input, means what camera gives you an image where, we get the object in terms of some pixels. Given that information, how do I correlate this information to the actual theta$_1$ theta$_2$ theta$_3$ such that, the robot manipulator end-effector finally reaches the end target point? So, these relationships can be found out using model based, model free and hybrid approach.

(Refer Slide Time: 11:33)



Model based method here; we derive the combined model of camera and manipulator that can be used to compute the joint space coordinate given the camera coordinate. Merit is with proper modeling it can give good result; demerit is loss of accuracy due to camera orientation and calibration errors. Model based means, we involve the camera. The camera gives us the position. Earlier, the models that we showed you - this (Refer Slide Time: 12:11) model is inter-twined with the model of the camera and then we again go backward. Given the camera input what should be the $theta_1$ $theta_2$ $theta_3$. This is the calculation; what is normally done is calibration of camera means. We take the known (Refer Slide Time: 12:38) x y z points and from there, we see the camera output. We correlate them in terms of certain parameters identify those parameters.

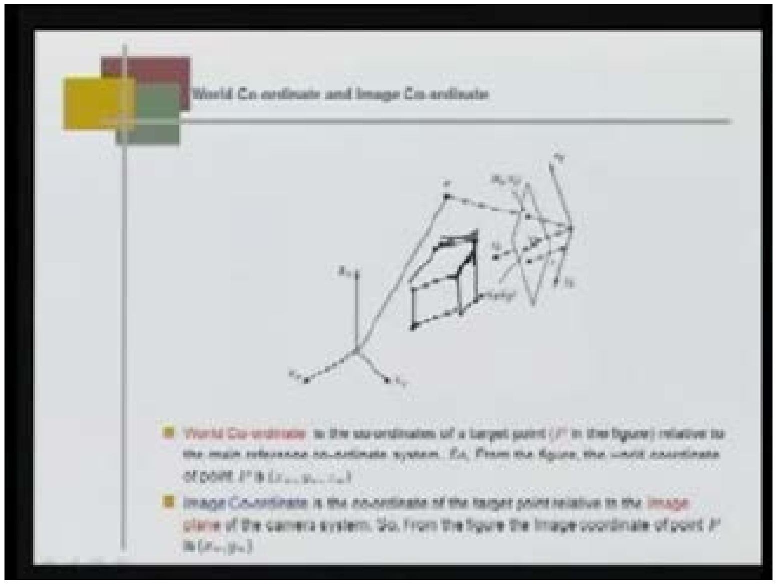
Then, inverse computing gives us from camera plane to x y z plane. This is called model based method with proper modeling; it can give good result but loss of accuracy due to camera orientation and calibration errors. So, calibration has to be done perfectly.

Model free method: here, we do not assume any model of the camera. Camera output is taken simply whatever is u_1 and u_2. We really do not correlate that to what is x y z; there is no need. But based on that, we learn using the, Kohonen base self organizing map; we learn how we reach the target. It is just like hand eye coordination that is, if my object is

(13:53) somewhere how I catch it is based on visual; like I throw and then hold it - hold. By throwing the object up and holding it, I have the hand eye coordination that means, I am looking at the thing, I am observing the object moment and then I am trying to catch it. This is called visual motor coordination. When I am doing this job, I am not actually of aware of any model on which I am working. Rather this whole thing I do while learning by practicing many times I learn how to catch. I really do not use the model of my arm; I do not really use the model of my eye but, it is simply a learning process this is called model free approach; this method is highly flexible. It ignores the inherent structure of the system training is time consuming- demerit. Hybrid approach system model is also considered here for learning. This method is highly flexible and inclusion of model reduces the error and training time; demerit learning always takes time to accomplish the task of coordinate transformation; camera calibration is essential.

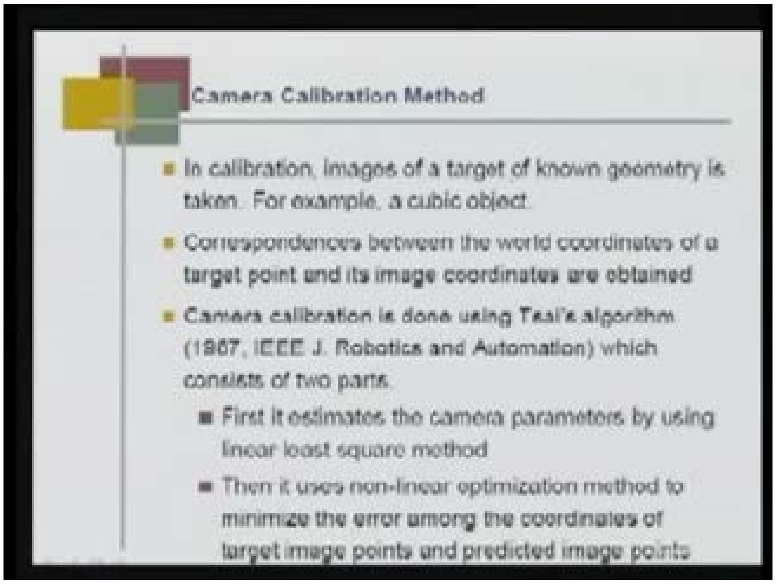
What is the overall thing that I told just now is, by putting a camera in the work space, robotic manipulation can be done using model based method where camera calibration is a necessity. Second is, I do not need camera calibration and still can do the learning simply by learning mechanism. The third approach is that, I can still introduce the camera calibration, do offline simulation for generating data and training. After I have done training on the offline model that includes the robot model as well as camera model, then I am almost done but, still calibration error may be there and the model also maybe there; there is some discrepancy. So, I start with these parameters; whatever the training parameters; I have already learnt and I do fine tuning in the field that is called hybrid approach. Combination of these two - this does not require any model, neither the model of the robot manipulator nor the camera model. This requires both the model of camera as well as robot manipulator. In hybrid approach, we take the positive of both and make a hybrid approach.

(Refer Slide Time: 16:53)



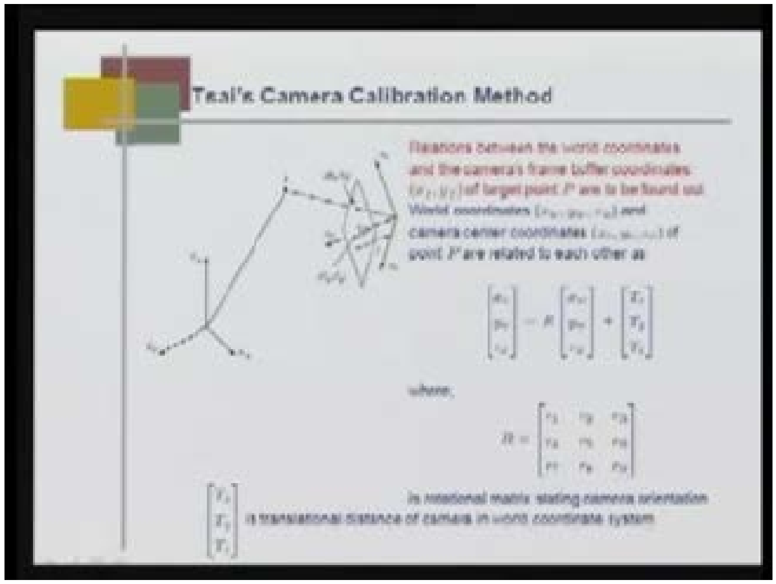
Now, before I go to calibration, I will explain world coordinate and image coordinate. World coordinate is you see that, this is my main reference plane and this is my p my target point or end-effector point. My end-effector is here or my target is here, where the end-effector is to go? The world coordinate is the coordinate spatial coordinate of this point with respect to this frame. Camera is also located in a specific frame which is x_c y_c z_c that is camera frame. If I put camera as the reference frame that is x_c y_c z_c the world coordinate is the coordinate of the target point relative to the main reference coordinate system. From the figure, the world coordinate of point is x_w y_w z_w. The image coordinate is this camera which is placed in this reference plane it has a focal plane. In this focal plane the object has a position and that position is x_u and y_u image coordinate is a coordinate of the target point relative to the image plane of the camera system. This is my image plane, this image plane center; central point is c x and c y, this is origin of image plane. From the figure the image coordinate of the point p which is here, this p is x_u y_u.

(Refer Slide Time: 18:42)



In camera calibration images of a target, known geometry is taken. For example: a cubic object that is, here (Refer Slide Time: 18:54) I place in this reference coordinate an object; this is a cubic object. With respect to this, I know what is given this point, this point, this point and this point. I know actually spatially known points. I generate in x y z coordinate that is, the calibration because meaning of calibration is, once I know given this point what is x_u y_u I should be able to predict. Given this point what should be x_u and y_u - this is called calibration. Correspondence between the world coordinate of a target point and its image coordinates are obtained. Camera calibration is done using Tsai's algorithm which you can find in IEEE journal of Robotics and Automation in 1987 which consist of two parts: First it estimates the camera parameters by using linear least square method. Then, it uses non-linear optimization method to minimize the error among the coordinates of target image points and predicted image points.

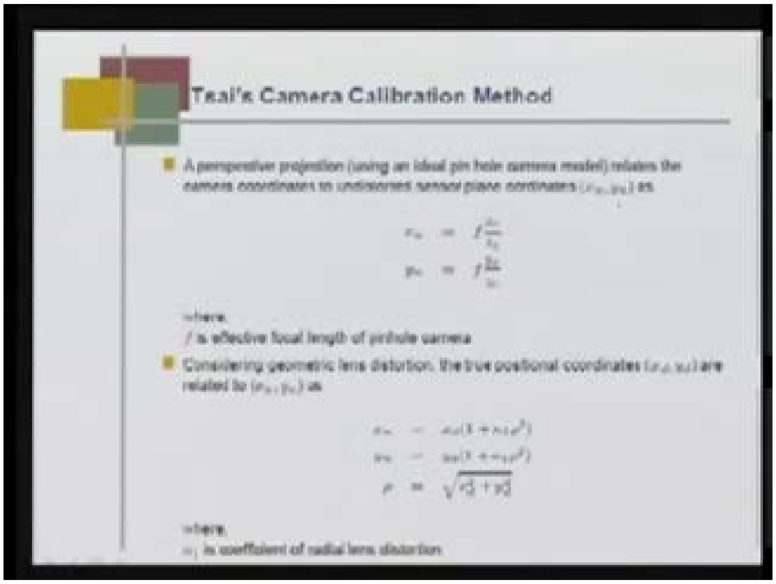
(Refer Slide Time: 20:28)



I will just explain very briefly; you can actually go to this paper that I just referred to understand how the calibration is done. There are other calibration methods but, I am just focusing size calibration method. This is my world coordinate, this is my camera coordinate. First, I express the camera coordinate, this is camera coordinate which is x_c y_c z_c. The relation between world coordinate and camera frame buffer coordinates of target point p are to be found out. The world coordinate x_w y_w z_w; this is the world coordinate of the end point, point p are related to each other. This p can be expressed either in this reference plane or in this reference plane. In this reference plane this is x_c y_c z_c and in this reference plane this is x_w y_w z_w. The point p here is either x_w y_w z_w in world coordinate or x_c y_c z_c in camera coordinate. If I assume that wherever the camera is spatially located that to be the origin, but in camera coordinate x_c y_c z_c can be written very simply.

That is, by rotational matrix r which is r_1 r_2 $r_{3,}$ r_4 r_5 $r_{6,}$ r_7 r_8 r_9; because this is a three link manipulator. Plus the translational term T_x T_y T_z where r is the rotational matrix taking camera orientation and T_x T_y T_z is translational distance of the camera in world coordinates. By using this, how many parameters will I need to compute x_c y_c z_c x_w y_w z_w? Actually this is 9 plus 3, 12 parameters I need to convert from x_w y_w z_w to x_c y_c z_c, from world coordinate, the representation of p to the camera frame coordinate of p.

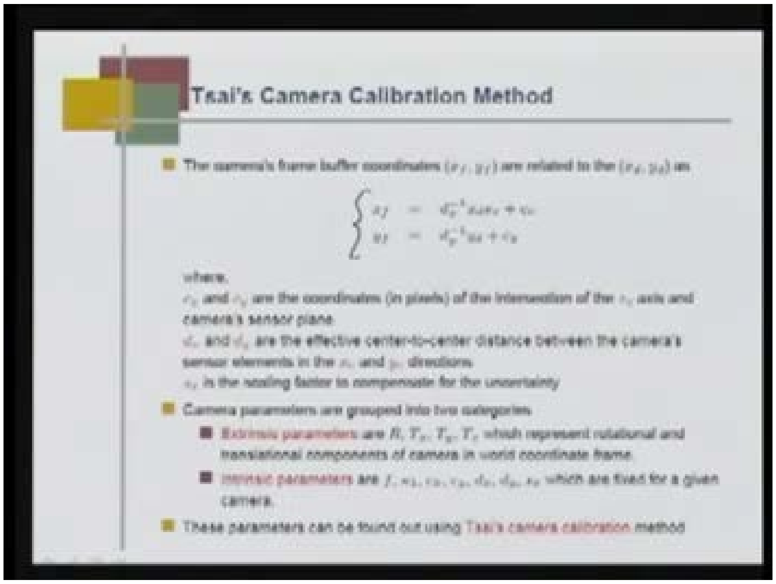
(Refer Slide Time: 23:07)



Now once I have this p (Refer Slide Time: 23:12) represented in x_c y_c z_c then, there is a very easy relationship between how this p will be mapped to the focal plane of the camera. That is given by: x_u y_u is f is the effective focal length of the pinhole camera into x_c by z_c is the x_u and y_u is f upon f into y_c upon z_c. The perspective projection relates the camera coordinates to undistorted sensor plane coordinates as in this formula that is a simple formula. Normally, we do not observe x_u and y_u what we observe is x d and y d due to distortion. This distortion is due to geometric lens distortion.

Instead of xu yu, we get x d and y d that includes the distortion. Given x d and y d, I must know what is xu and yu and that xu and yu are given by this relationship where k_1 is the coefficient of radial lens distortion.

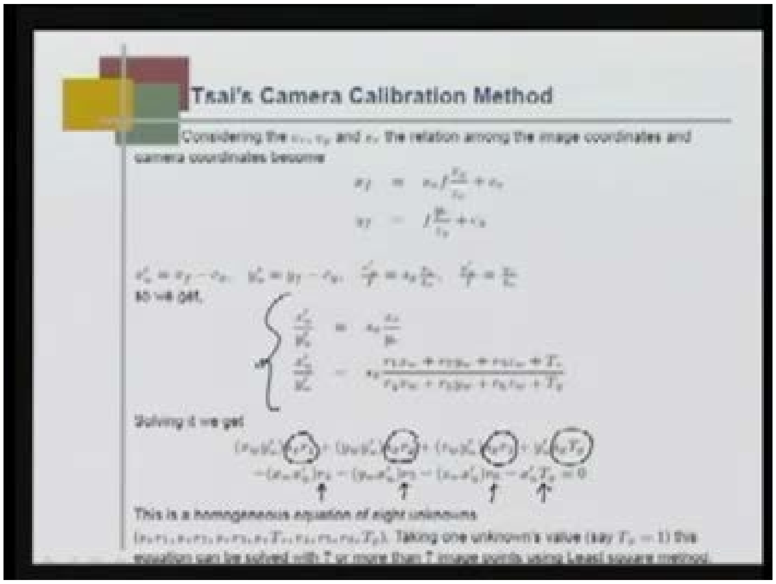
(Refer Slide Time: 24:33)



Now, once x d and y d is given that I am observing the camera frame buffer coordinates which is, x f and y f. This I actually get from a computer image x f and y f. Given an object, what is the position of object is denoted by x f and y f which is, written by d x inverse x d s x plus c x d y inverse y d plus c y where c x c y are the coordinates in pixels of the intersection of z axis and camera sensor plane d x d y are the effective center to center distance between the camera sensor element in the x c and y c direction and s x is the scaling factor to compensate for the uncertainties. By this method finally, we get x f and y f. The camera parameters are grouped into two categories. One is extrinsic parameter which is r T_x T_y and T_z that was we said (Refer Slide Time: 25:45) in this r T_x T_y T_z and r is this matrix. Intrinsic parameters are f k_1 c x c y c d x d y and f x. You saw here, we have the parameters associated with f. These are all computed already; this k_1 and rho is you know x d square plus y d square is the rho; so that is already known.

In this k_1 and f is there and in this is d x d y s x and also c x c y which we said, c x and c y and you know that, this point is c x and c y. These parameters can be found out using Tsai's camera calibration method.

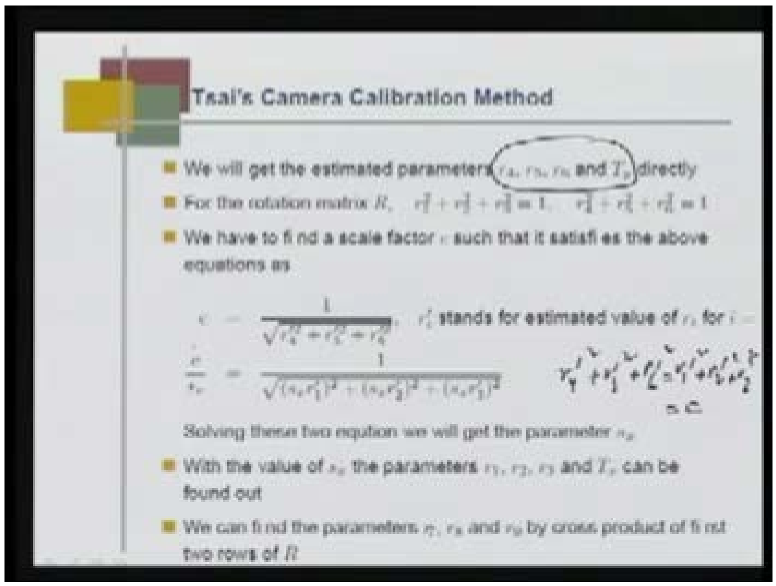
(Refer Slide Time: 26:50)



This is a long list I will not give details. What I am trying to tell you is that, simplification we can do to write x f is s x f x_c z_c plus c x and y f is f y_c z_c plus u y ignoring the distortion, I can write this. Once I write this, we can now represent x u dash is x f minus c x, y u dash as y f minus c x. You know that, I know y f similarly x u dash upon f, f is s x upon x_c upon z_c and y_u dash upon f is y_c upon z_c that is from this relationship. Finally, we get these two equations.

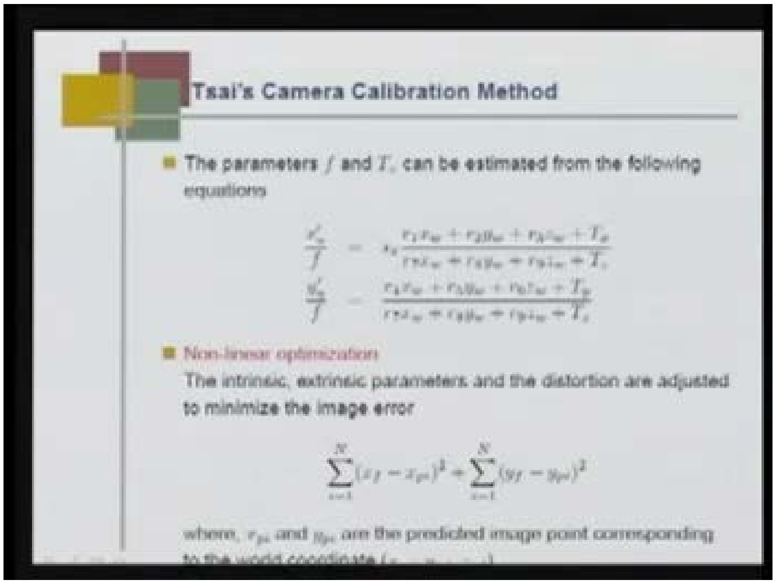
If I solve these two equations, I get this equation and you see that this is simply the objective is that, how I identify the camera model parameters that we started with world coordinate went to the camera coordinates. From camera coordinates, camera image plane; from camera image plane to the actual position of the target point in the camera image plane, which is x f and y f. Given x f and y f and given actual x y z in the world coordinate, how do I compute these parameters which is s x r_1, s x r_2, s x r_3, s x t x? Then r_4 r_5 r_6 m t y, these are the parameters. I already know x w y u des y w because, x f is known c x and c y are given x u this is known from t. So, these x_w y_w z_w are all given. Now, we can collect this data point 1,2,3,4,5,6,7 and 8. Eight data points, eight unknown are here of course, here you see that, 2 unknowns are clubbed together. But, these 8 unknowns and these are known if I have many points I can always solve this using (29:39) technique. That is a homogeneous equation of 8 unknown s x r_1, s x r_2, s x r_3, s x t x, r_4, r_5 and r_6 and t y they can be easily estimated. Taking one unknown value say t y equal to 1; this equation can be solved with 7 or more than 7 image points using (30:01) methods. Now, the other points given these combined values, how do I estimate s x first? Once I estimate s x then, I can easily estimate what is r1 r_2 r_3.

(Refer Slide Time: 30:18)



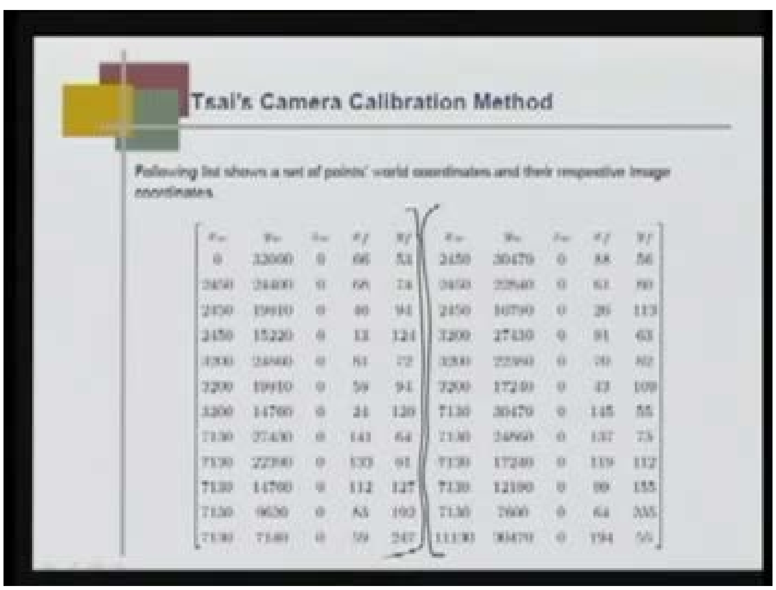
This is a simple method. Out of the eight unknowns we directly get r_4 r_5 r_6 and t y. We have the information r_1 square r_2 square plus r_3 square is 1 also r_4 r square plus r_5 square plus r_6 square is also 1. When we estimate using least square we do not get exact values. So, assuming one upon r_4 square plus r_5 square plus r_6 square to be c this root over and also c upon s x is this quantity which you can easily see. Assuming that, this quantity root over r_4 square plus r_5 square plus r_6 square I estimate is equal to r_1 square plus r_2 square plus r_3 square. Here the assumption is, r_4 del square, r_5 del square, plus r_6 del square is r_1 del square, r_2 del square, r_3 del square is c. By assuming that, we know the estimate of this so from there we can compute c. We also have estimate of this; we know this quantity, so we can easily find out s x. With the value of s x the parameter r_1 r_2 r_3 and t x can be found out. Then, the next is r_7 r_8 r_9 they are found out because, their cross product of the first two rows of r which is r_1 r_2 r_3 and r_4 r_5 r_6.

(Refer Slide Time: 32:14)



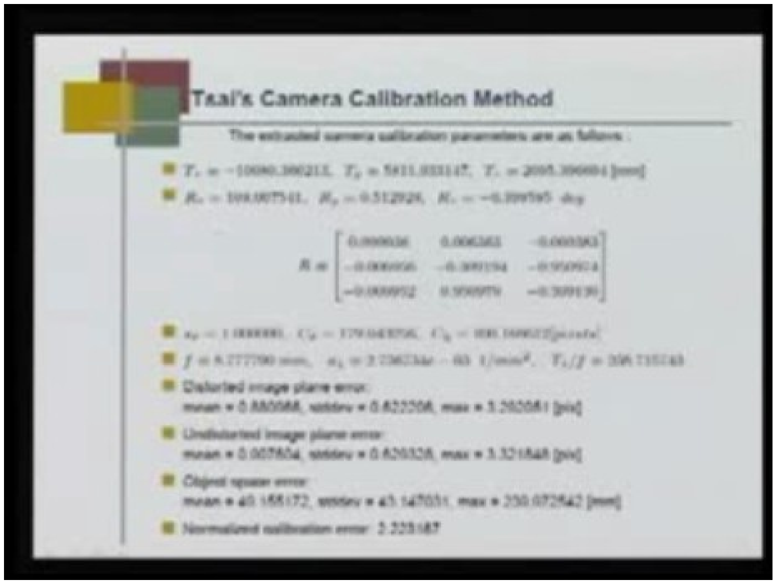
Finally, the non-optimization used to fine tune whatever the estimation has been done because, the estimation has certain approximation because, and distortion has been omitted during the estimation. For that a nonlinear optimization method is done.

(Refer Slide Time: 32:33)



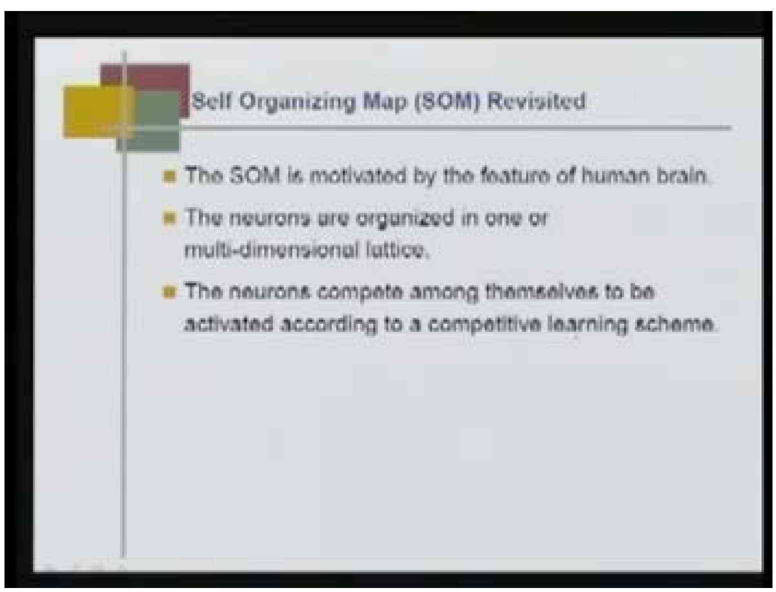
You see that, the object the approach is x_w y_w z_w. So, these are my actual points in the space coordinate and this is my actual image coordinate u x f and y f this I collect from the actual system.

(Refer Slide Time: 32:57)



Then using the algorithm you can find out what is T_x T_y T_z, R_x R_y R_z and r is this rotational matrix s x c x c y f kappa$_1$ t f by f t z by f.

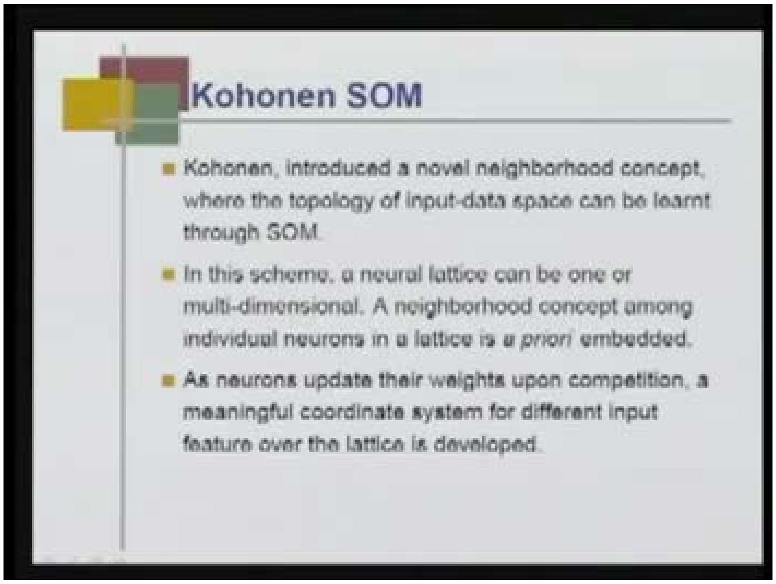
(Refer Slide Time: 33:20)



All the parameters of the camera can be computed using this algorithm, what I just told you that, camera calibration helps us to find out the camera model. Given x y z in the actual world coordinate, camera model helps me to find out what should be x f and y f or which we say u_1 and u_2. Once I know this camera calibration the advantage is that, I really do not have to do experiment to compute. Once I do the camera calibration, I can take this model incorporate in my robot manipulator or combine with the robot manipulator model I can generate data that is given x y z. I can randomly generate theta_1 theta_2 theta_3, it will take two specific x y x- x z and from that, x y z, I can convert what should be my camera plane coordinates.

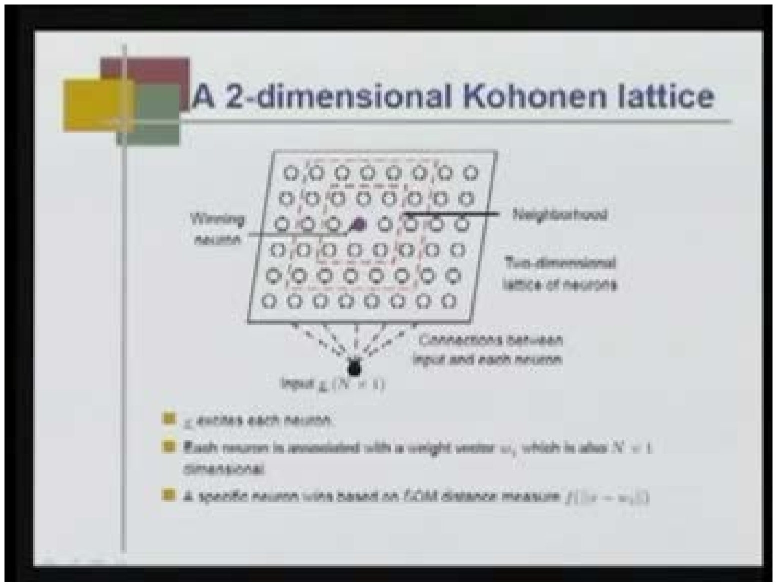
Now, we will talk about self organizing map that, we have already studied in the last class. The SOM is motivated by the feature of human brain. Neurons are organized in one or multi-dimensional lattice. The neurons compete among themselves to be activated according to a competitive learning scheme.

(Refer Slide Time: 34:56)



In Kohonen, we introduced a novel neighborhood concept where, the topology of input data space can be learnt through self organizing map. In this scheme a neural lattice can be one or multi-dimensional a neighborhood concept among individual neurons in a lattice is prior embedded as neurons update their weights upon competition. A meaningful coordinate system for different input features over the lattice is developed.

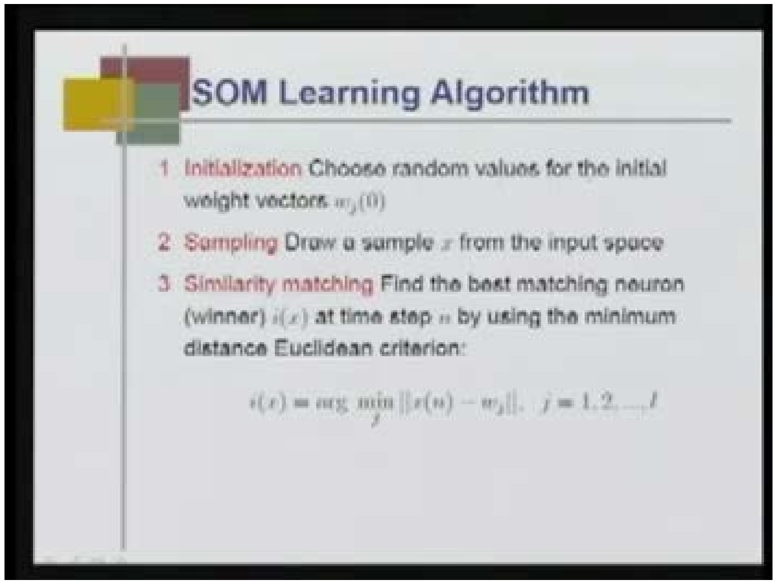
(Refer Slide Time: 35:25)



Given a neural lattice like this is a 2 dimensional lattice and given the input space like for example: if my input data also is coming from a specific 2-D geometry then, the reference the coordinate points of this neurons would be such that, they would represent almost they will learn the actual topology of the input space. If the input space is 3-D they will also learn the 3-D topology.

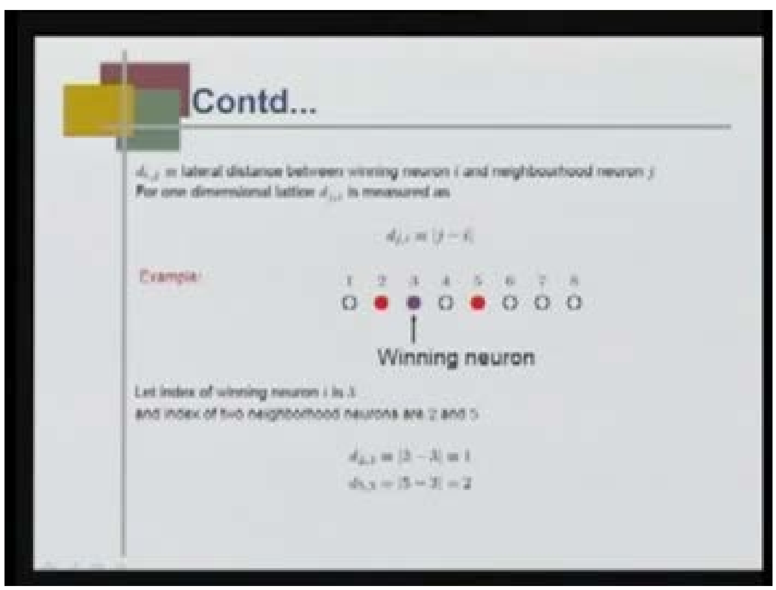
If input space is 1-D they will also learn 1-D topology. The objective of a 2-D Kohonen lattice is my input data that excites all the neurons, one of the neurons becomes the winner. The winner is decided by the equilibrium distance between the neuron reference vector and the input vector. Each neuron is associated with weight vector w_i…a specific neuron wins best; on some distance measure this normally is an equilibrium distance measure.

(Refer Slide Time: 36:38)



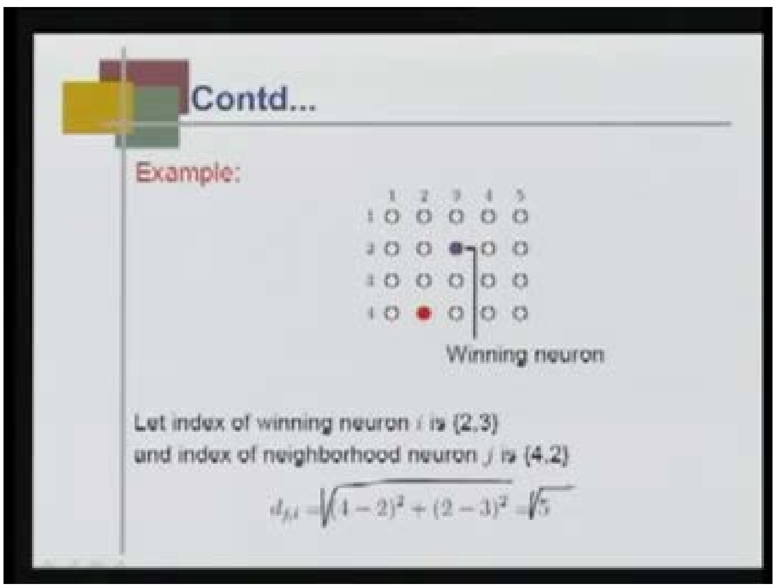
The SOM learning algorithm is choose a random value for the initial weight vector w_j. Draw a sample x from the input space, find the best matching neuron at the time step by using the minimum Euclidean distance criterion.

(Refer Slide Time: 36:57)



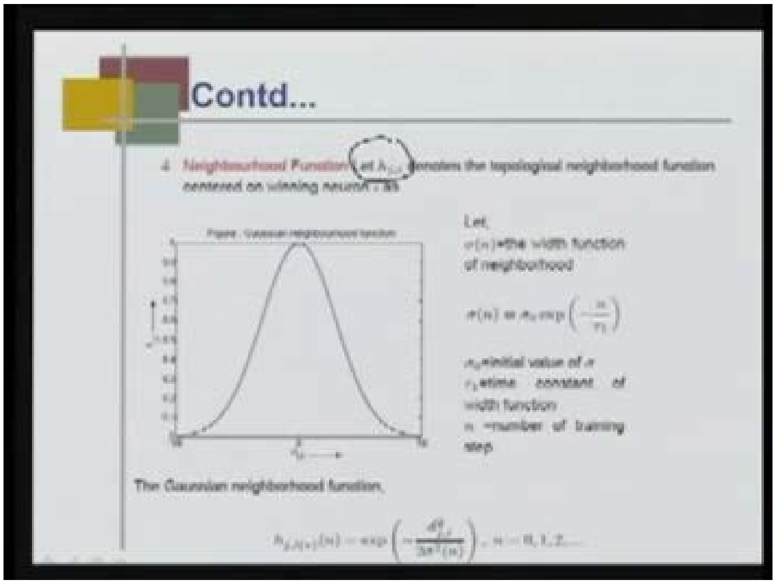
Like for example: this is a 1-D lattice. This is my winning neuron and the neighborhood neurons like the number, this is my neighborhood neuron the distance is 1 and this neuron the distance from winning neuron is 2. While I find (Refer Slide Time: 37:27) the winning neuron I also find the distance of each neuron from the winning neuron. How far they are from these winning neurons? Given a neural lattice, we find first of all winning neuron based on the equilibrium measure distance. The neighborhood, the distance of the neighboring neuron from the winning neuron is computed. In this case, this neuron and this neuron has a distance then, we can say this is 1 and from here to here this is 2, from 8 to 3 of course, it is 5. If I think that 1 and 8 are connected then, you have to see that this distance the lesser distance is to be taken.

(Refer Slide Time: 38:25)



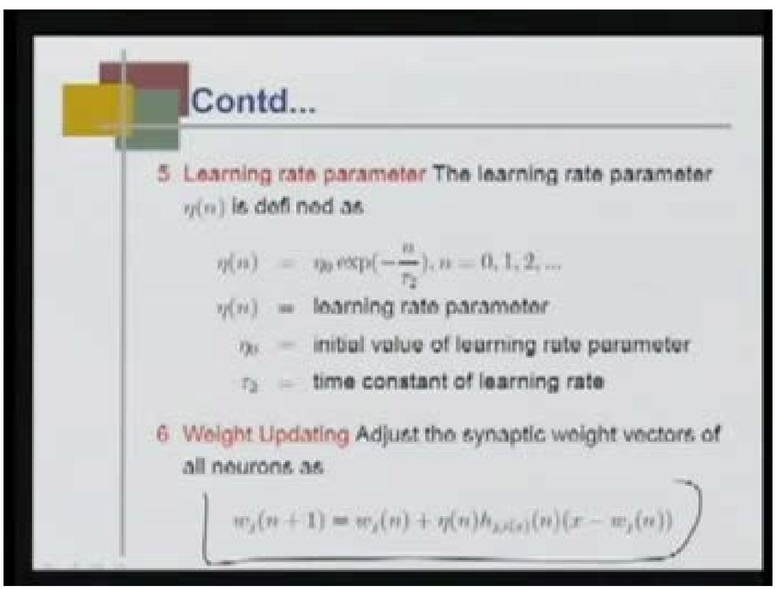
If I have 2-D lattice if this is my winning neuron then this neuron has a distance. You can easily see this neuron has a position 4 and 2. The winning neuron position is 2 and 3. The distance is 4 minus 2 whole square and 2 minus 3 whole square which is 5. Of course, you can also take the square root that is alright whatever the distance you say.

(Refer Slide Time: 38:58)



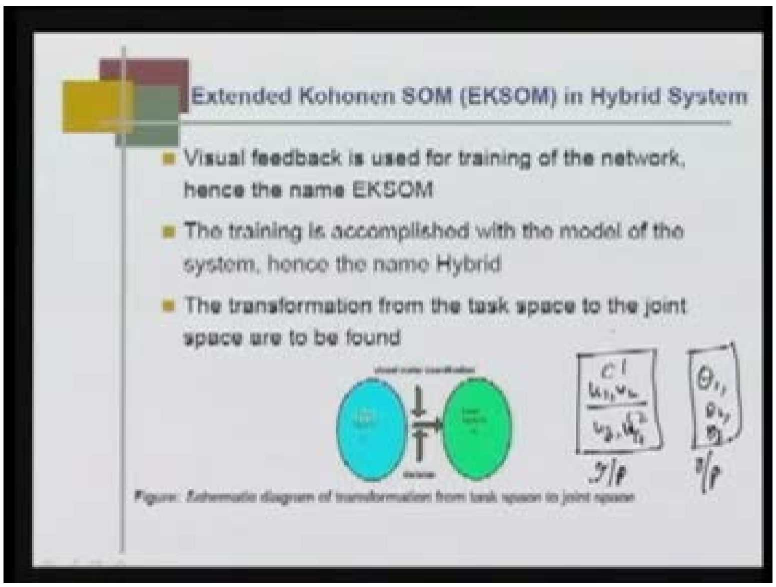
Given the distance, we define a neighborhood function $h_{jI;}$ h_{jI} is a neighborhood function. T-hat means, it says, how close a specific neuron to a winning neuron that is, e to the power minus the distance square upon 2 sigma square. So, it is better that we have written distance like this. e to the power minus the distance square upon 2 sigma square.

(Refer Slide Time: 39:43)



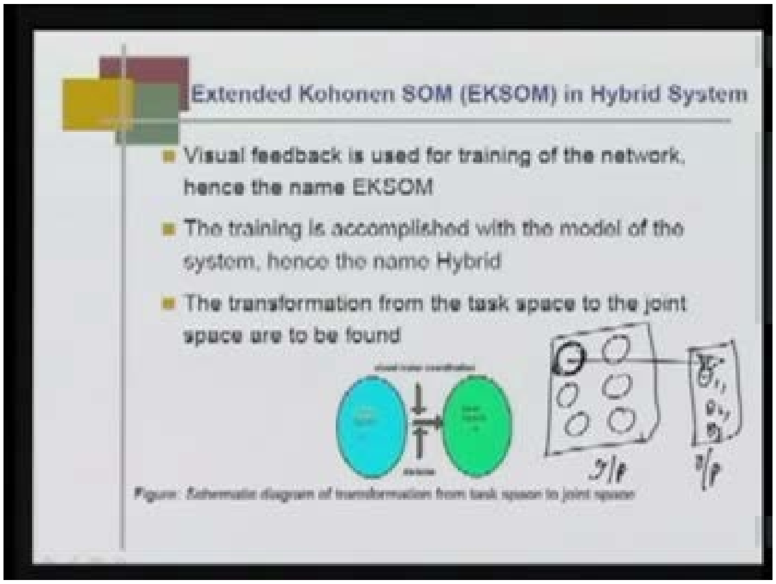
So, once I find out the winning neuron and their neighboring neurons with their neighborhood function then, the weights of each neuron is computed by this. This is my Kohonen algorithm that is, each neuronal index is updated based on its old reference vector; this is the learning rate; this is the neighborhood function. You know that neighbor function will be one for the winning neuron and will gradually decrease for the neurons that are farther away from winning neuron into this is my input vector that excites the neuron minus the reference vector of the neuron.

(Refer Slide Time: 40:40)



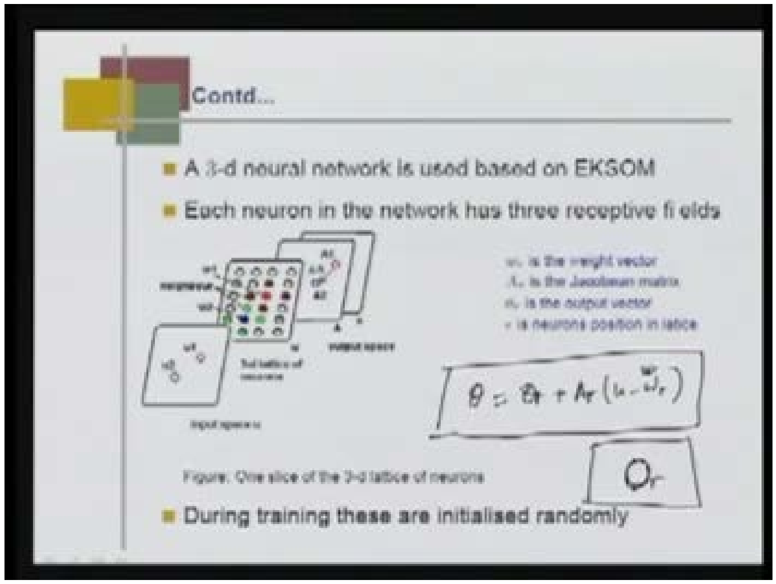
That is, Kohonen lattice it is all (40:45). Now we will be talking about extended Kohonen SOM: the visual feedback is used for training of the network. Here, the concept is little different. We try to correlate between input space and output space. In the visual motor coordinate my input space is u_1 u_2 camera 1 coordinate and u_3 u_4 is the camera 2 coordinate. So, this is camera 1 and this is camera 2. In that space objects are located. I get those point this is my input space and then my output space is theta$_1$, theta$_2$ and theta$_3$. So, one is that, we do the clustering of the input space. Meaning of clustering of the input space is that, I represent my input space using finite number of representative points which I said that is what the Kohonen lattice does.

(Refer Slide Time: 42:16)



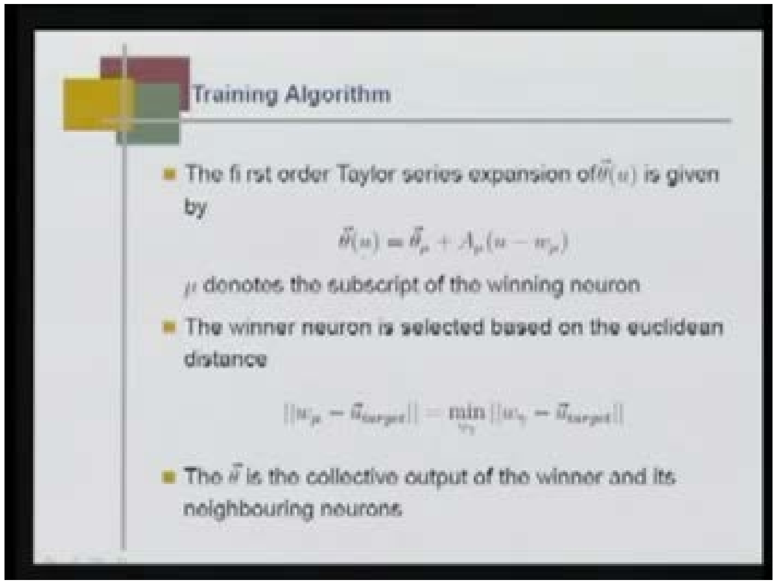
So, in a sense what I do is in input space, I represent some finite cells. These are finite cells which we say here, in this case Kohonen neuron. Within each neuron, we try to establish a relationship between the input space to the output space; that is the object. This is why, we see only Kohonen self organizing map is simply clustering. Extended Kohonen SOM is connecting this clustering to the output space. How do we do it is you know that, the relationship between the input space. The camera image plane coordinate to theta is actually nonlinear but, this we can assume to be linear in the discretized cell. That is the objective.

(Refer Slide Time: 43:11)



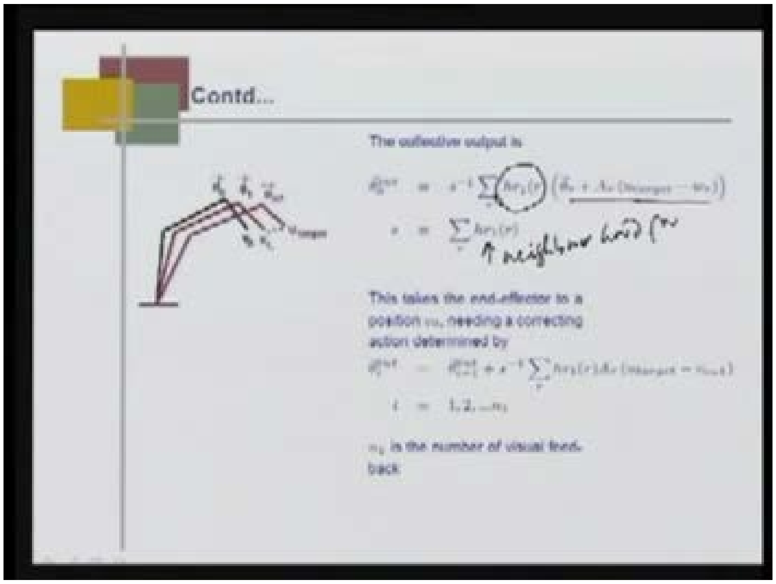
This is what is done here; this is my 3-D lattice this is my u_1 camera 1 coordinate, camera 2 coordinate, each coordinate gives you x f and y f. You have this vector is 4-dimensional vector obviously; the w_y is of these neurons. This is the 3-D lattice and each neuron will have a reference vector which has also the dimension 4 by 1. Then, we transfer or we map the input space to output space through a Jacobian matrix A. By this relationship using Taylor series first order expansion that is, theta equal to theta_r plus A_r u minus w_r; w_r is associated reference vector A_r is the Jacobian. That means what I am saying is that, if this is my input space and this is my rth discrete cell, for each discrete cell associated, the discrete cell is actually a Kohonen lattice neuron. This neuron is associated with weight vector W_r and Jacobian vector A_r as well as a reference theta vector theta. Theta given any other point u which is close to W_r then, the corresponding theta is computed by this linear equation. But unfortunately, we do not have the information what is theta_r what is A_r and W_r, we have learnt using the Kohonen. We know only u; we have to compute what is theta. We do not know what is theta_r and A_r, that has to be learnt. So, this is learnt using some feedback mechanism.

(Refer Slide Time: 45: 21)



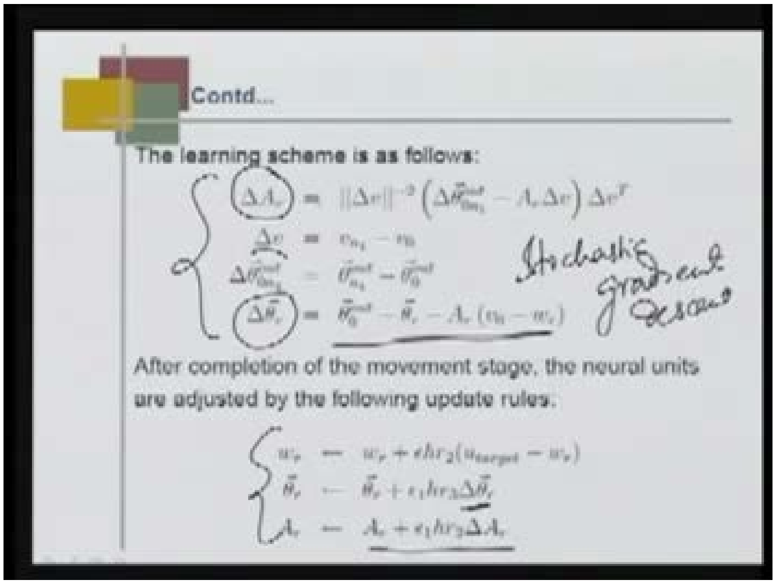
This I have already told you, how theta is related to parameter w_{mu} and $theta_{mu}$. The winner neuron is selected based on the Euclidean distance first. Theta is the collective output of the winner and its neighboring neurons. So, what is final theta (Refer Slide Time: 45:42)? Each neuron will predict given u_1 u_2 winning neuron will predict some theta, so also the neighboring neuron. The final theta is the overall decision making. But, the overall decision making is made by giving less importance to those neuronal decision which are further from the winning neuron.

(Refer Slide Time: 46:13)



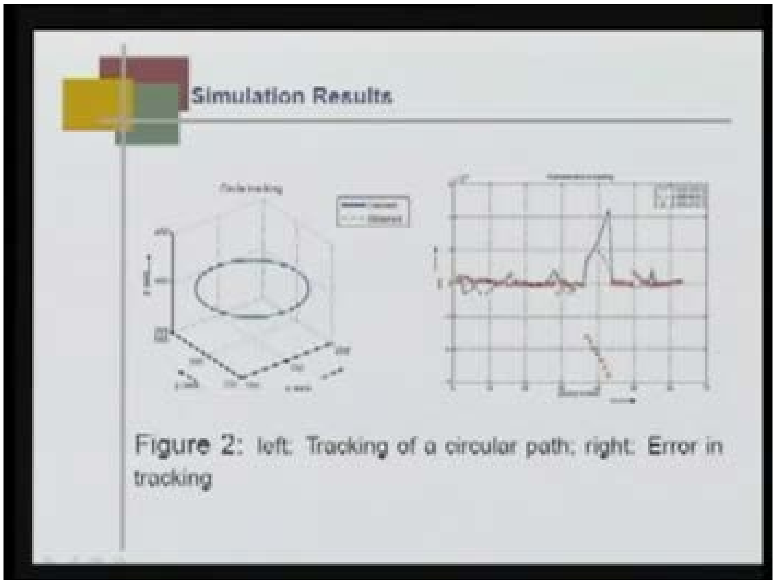
This is my theta output the collective output. This is my individual output. This is my neighborhood function of that individual neuron and I sum that with over r, r is representing neuron. So, if my neural lattice has over hundred neurons, I compute all the responses and s inverse is simply the summation of the distance function. This is my distance function or neighborhood function and this neighborhood function is the highest value is maximum value is 1 and all of other values are less than 1. So, the objective is that, given this target in the beginning some kind of random theta$_1$ theta$_2$ theta$_3$ are generated because, we start from no knowledge. Then, slowly the objective of learning is such that, the end-effector should go to positions like from v_0 to v_1 until it reaches u target. That is the purpose of learning. I just learnt. This never happens systematically or randomly. I will show you a movie that would say how this actually happens.

(Refer Slide Time: 47:47)



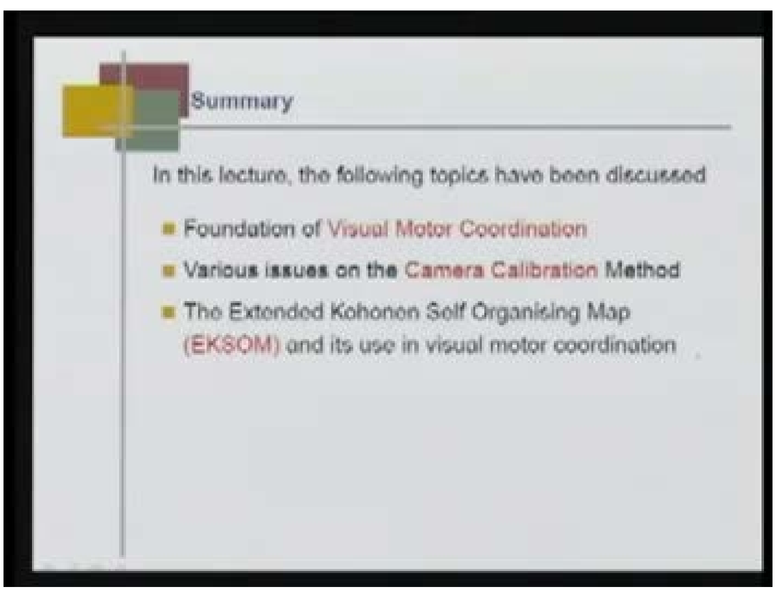
This is the learning scheme that has been using stochastic. This learning scheme has been derived by stochastic gradient descent that minimizes the error between the target point and the end point. So, this is how I learn the Jacobian matrix. How I learn my theta$_r$? This is my final learning algorithm. My reference vector is learnt by this function that is, you can easily see this is my Kohonen learning algorithm theta$_r$ by this expression where delta theta$_r$ is given by this expression and A_r is given by this expression where, delta A_r is given by this expression. Where, delta v is given v n_1 by v_0, v_0 is give initial theta$_1$ theta$_2$ theta$_3$ to the camera point then v_0. Like that, in n steps whatever is the output is v n_1 in n_1 states that is my delta v.

(Refer Slide Time: 49:18)



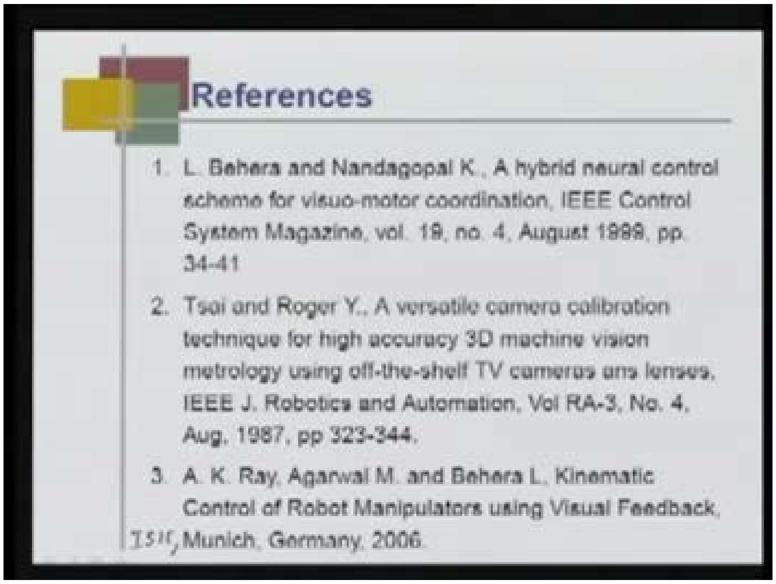
Given this algorithm, now we try to learn a circle, we made the robot to track a circle. So you see that this is circle and the desired one is in blue and the obtained actual points that were covered by the robot manipulator is the green and you can easily see that almost very closely followed. Error point you see in this domain is obtained to the power minus 4. The error is very small tracking of a circular path and in this figure error in tracking.

(Refer Slide Time: 50:05)



In this lecture the following topics have been discussed: Foundation of visual motor coordination, various issues on the camera calibration, Extended Kohonen self organizing map and its use in the visual motor coordination.

(Refer Slide Time: 50:20)



The references are: one of our papers in IEEE Control System Magazine in 1999. For camera calibration you can follow Tsai's paper in 1987 IEEE journal Robotics and Automation. Also we have another paper regarding this work on international symposium on Intelligent Control, Munich, Germany 2006. Finally, I end this lecture showing you two movie files that will show you how the self organizing of the input space takes place and finally how robot really learns through steps reaching the target point.

Thank you.