Intelligent Systems and Control

Prof. Laxmidhar Behera

Department of Electrical Engineering

Indian Institute of Technology, Kanpur

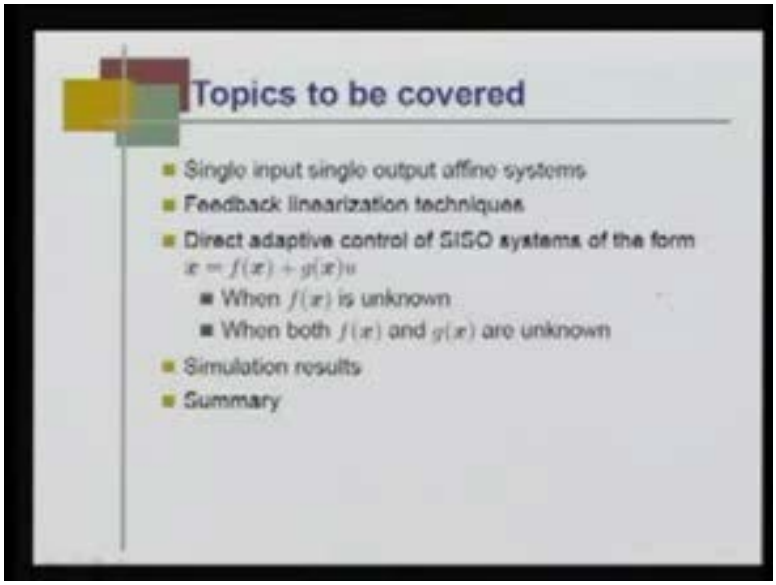
Module - 3 Lecture – 5

Adaptive Neural Control for Affine Systems (SISO)

This is the lecture five of module three on neural control. The topic today is adaptive neural control for affine systems. This lecture would be little more involved mathematically as the neural control is always there and deals with mathematics, because of stability analysis hence forth.

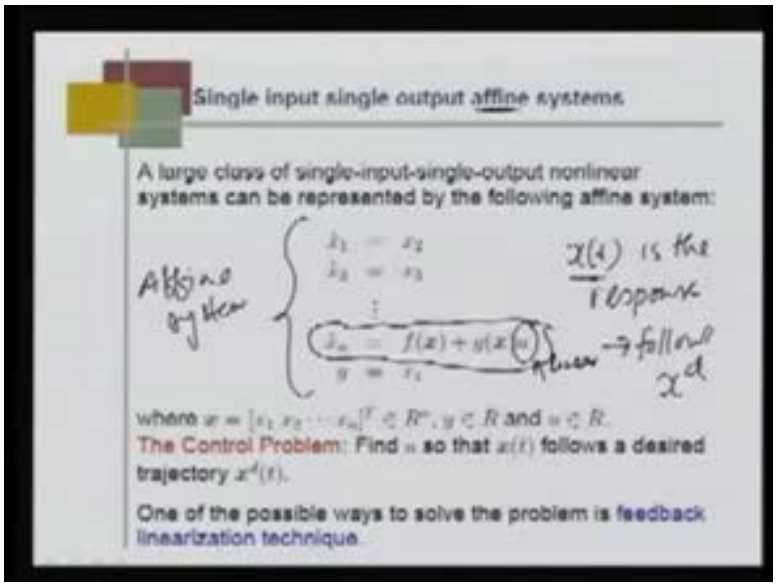
Last three classes, we focused on indirect adaptive control scheme, but this class, today will be on direct adaptive control scheme using neural network. The earlier indirect adaptive control scheme was generic, can be applied to any nonlinear system. But today, we will be confining only to a specific class of nonlinear system which is affine system; adaptive neural control for affine system. Also today, we will be only focusing on single input single output system.

(Refer Slide Time: 01:50)



Topics to be covered are single input single output affine system, feedback linearization technique. What is the meaning of feedback linearization? It is direct adaptive control of single input single output system, of the form x dot equal to f x plus g x u. This is actually the affine system and for controlling this, we can always assume g x to be known and f x is unknown; this is the first case. Second case is, when both f x and g x are unknown. This is obviously more difficult to solve and simulation results and summary. Before we assume f x and g x to be unknown, let us assume what is f x and g x. If f x and g x both are known, then this control scheme is solved using feedback linearization.

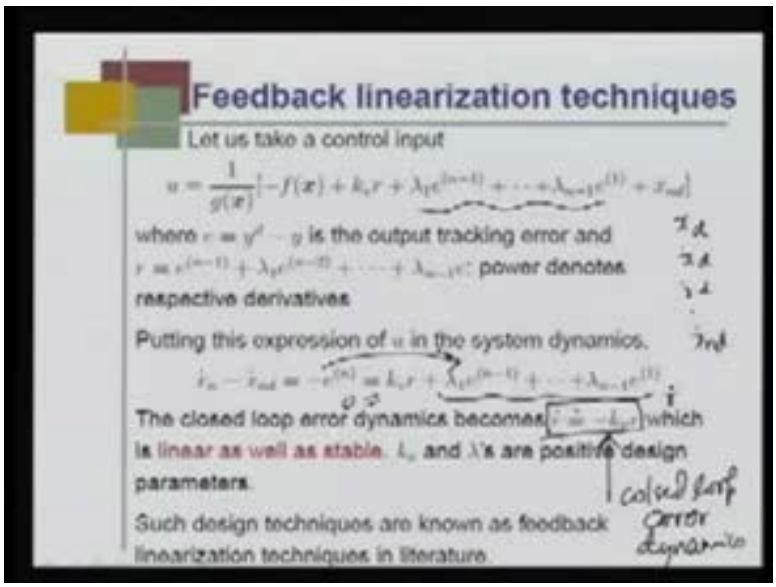
(Refer Slide Time: 02:55)



This is the first part we will be talking. So, this is the single input single output affine system. The structure is here; given a large class of single input single output. Nonlinear system can be represented by following affine system, where x_1 dot is x_2, x_2 dot is x_3 and so on until x_n dot is f of x. f is a scalar but x is a vector of x_1 until x_n. Similarly, g again is a scalar, but x here is a vector and that is x_1 to x_n u and y is x_1. If any differential equation model of a dynamical system, if you represent in this form this is called affine system. This is nonlinear because here is nonlinearity. So, where x is a vector x_1 x_2 until x_n and we have single input and single output, the control problem, find u such that x_t follows a desired trajectory x_t, the response of the system; of course, this is n dimensional system the response is…. so, find the control u such that, the x t follows x d desired

trajectory. One of the possible ways to solve the problem is feedback linearization technique. What is feedback linearization? In feedback linearization, what we normally do, you see that this is u. So, linearization means u should be such that this term will become linear if I find out the overall system. Here, this would turn out to be linear.

(Refer Slide Time: 05:31)



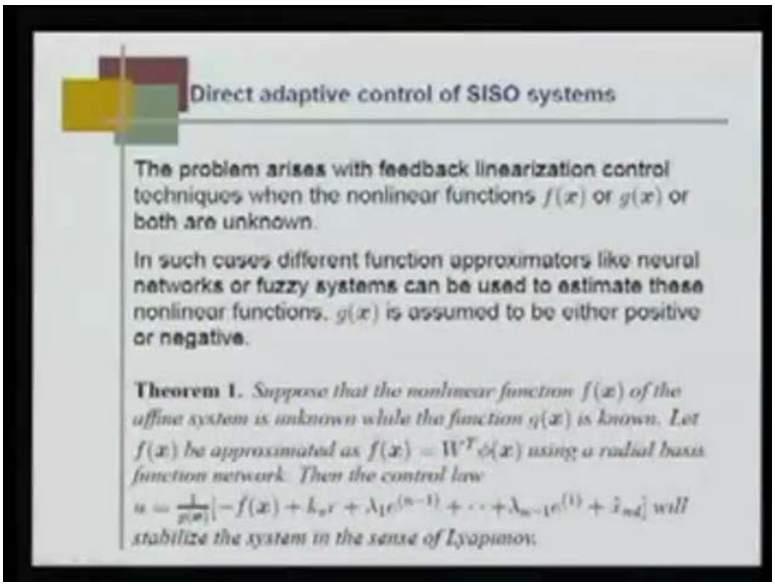
Now, let us take a control input u equal to 1 upon g x minus f x plus kv is some constant into r is the reference trajectory plus lambda 1 e n minus 1th derivative until lambda n minus 1 e first derivative that means e dot plus x n d dot so this is what is x n d dot. If I know x desired, so I always know what is x dot d x double dot d and so on, until the last one we write x n dot d, where e is y d minus y. This e the error is tracking error y d minus y is the output tracking error and r is e to the power n minus 1 plus lambda 1 e to the power n minus 2 plus so on lambda n minus 1 e.

So you see that if I differentiate this r dot I get e n plus this term, this whole term plus e n e nth derivative is the r dot; so the power denotes respective derivative. Putting this expression of u in the system dynamics so if I place this u here {referring fig., 05:18}so I bring the control law that I proposed and put it here and simplify this expression I get x_n dot is on the right hand side in the left hand side is the x_n d dot. I bring it here this side minus x_n d dot which is obviously minus e to the e nth derivative of the tracking error,

equal to this side you see that g x minus f x is there. With u, you have g x so cancels out minus f x and plus f x cancels, so then you are only left with k v r plus these terms here lambda 1 e to the e n minus one derivative until e first derivative. The closed loop error dynamics becomes if you see here this one, minus e n if I bring to this side then this becomes e minus e nth derivative; because plus e nth derivative and this quantity and I said these quantities are dot, so k v r plus r dot is 0 or r dot is minus k v r. So bringing this term to this side then coupling this you get r dot, so r dot k v r is to be 0 and from here we get this term r dot is equal to minus k v r so which is linear as well as stable. So this is r dot is minus k v r is a linear in r and also stable because if I assume k v to be positive then it is stable. So k v m lambda are positive design parameter such design techniques are known as feedback linearization techniques in literature. Because by selecting such a technique, what we try to feedback although the system dynamics is nonlinear by feedback the closed loop error dynamics becoming linear closed loop, this is called closed loop error dynamics.

This closed loop error dynamics is linear, so we understood by now what is affine non linear systems single input single output case. Now we are talking about that what is also we understood what is feedback linearization. Now we will be talking about the controlling such systems when the functionality that is, f x and g x these functions are unknown.

(Refer Slide Time: 11:00)



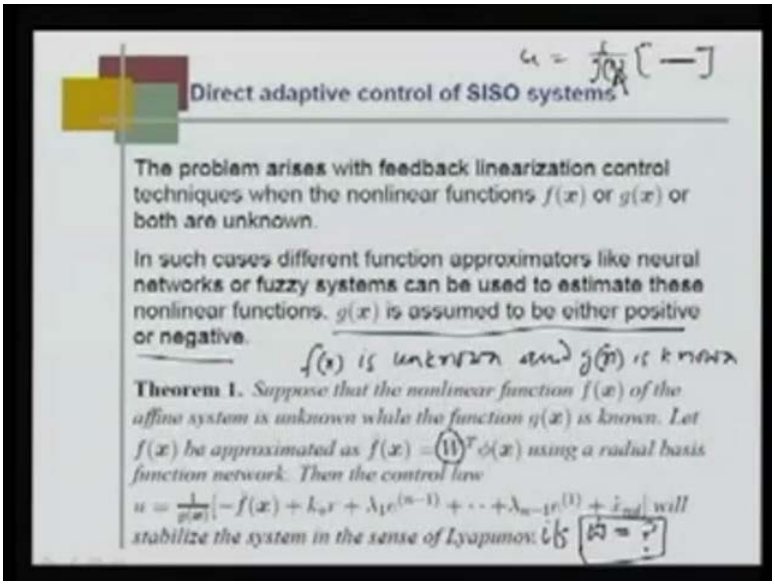
The problem arises with feedback linearization control technique when nonlinear function f x or g x or both are unknown either f x is unknown or both f x and g x are unknown. In such cases different function approximates like neural network or fuzzy systems can be used to estimate these nonlinear functions. g x is assumed to be either positive or negative this is very important.

That is, we do not want the g x value associated with u to cross 0 value. So it goes either it is negative or it is positive it is not both. That is because in otherwise you cannot apply a feedback linearization which means there will be a term u, u is one upon g x and some term here. If g x at some point of time is 0 then control input control is because this is the structure of feedback linearization so g x cannot be 0; so either it is positive or negative also it as a finite lower bound.

Now theorem 1 for direct adaptive control; what is direct adaptive control? I have to explain to you direct adaptive control. In this case we do not identify the system as a whole we do not identify the system rather the objective is how do I select a control law parameterized control law and the parameters of the controller are tuned directly as a function of tracking error. As long as such an adaptive mechanism ensures the stability of

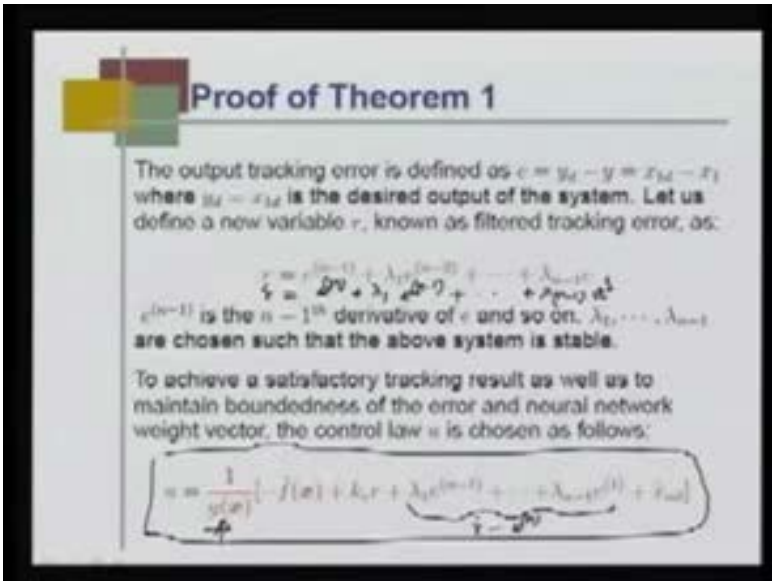
the entire closed loop system, we are satisfied and such a system is direct adaptive control means controller is directly tuned as a function of tracking error.

(Refer Slide Time: 13:26)



This is the direct adaptive control scheme the first theorem we assume in this first one f x is unknown and g x is known. When g x is known and f x is known this theorem tells a control law that will stabilize the system. Suppose that the nonlinear function f x of the affine system is unknown while the function g x is known let f x be approximated as f hat x is W hat transpose phi x using radial basis function network. Then the control law u equal to 1 upon g x minus f x f hat x plus k v r plus lambda 1 n minus 1th derivative of error plus so on; lambda n minus 1 first derivative of error plus x_n d dot will stabilize the system in the sense of Lyapunov. If this is my control law, we can show that the system will be stabilized.

(Refer Slide Time: 15:19)

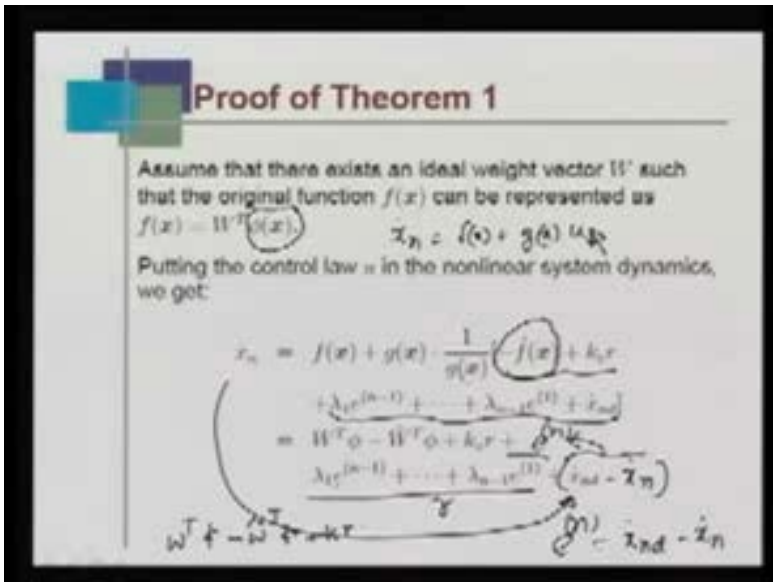


Proof of the theorem 1: the output tracking error is defined as e is y d minus y which is also same as x_1 d minus x_1 where y d is x_1 d the desired output of the system. Let us define a new variable r known as filtered tracking error which is r is e to the power n minus 1 plus lambda 1 e to the power n minus 2 plus and so on lambda n minus 1 e so this bracketed terms are the derivative term so this is n minus 1th derivative of error n minus 2 nth derivative of error. You can easily see if I write r dot then this will be e n plus lambda 1 n minus 1th derivative; lambda n minus 1 e post derivative. So e to e bracketed n minus 1 is the n minus 1th derivative of e and so on, lambda 1 until lambda n minus 1 are chosen such that the above system is stable.

To achieve a satisfactory tracking result as well as to maintain boundedness of the error and neural network weight vector, the control law u is chosen as follows: according to the theorem this is our control law you can check here in this control law this as a feedback linearization structure that is 1 upon g x. In the first case, we have assumed g x to be known and also we assume g x is not 0; that is either negative or positive and it as a finite lower bound. Now you see that this is minus f hat x; this is the estimation neural estimation of the f x using a radial basis function network and here if you look at in this particular term this is my k v r.

Where r is defined like this and this shown you can see this term is r dot minus e n; this term you can easily see; this is r dot minus nth derivative of e and plus x n d dot.

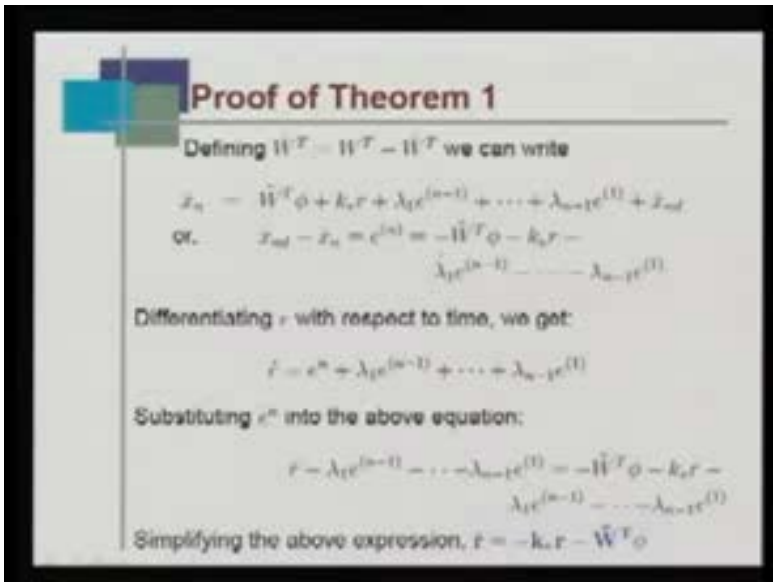
(Refer Slide Time: 17:45)



Assume that there exists an ideal weight vector W such that original function f x can be represented as f x W transpose phi x. Putting the control law u in the nonlinear system dynamics, what is our nonlinear system dynamics? That is, finally x n dot is f x plus g x u. Now you replace this u here so what you get x n dot. Then f x is here plus g x and u is 1 upon g x and these terms that was inside. Doing that what you get f x which is W transpose phi x this is assumed that there exists some basis function this is some basis function it is a vector into W transpose this weight vector. If we sum that we can always approximate any nonlinear function; that is the assumption. W transpose phi minus W hat transpose phi this comes where you see that g x g x cancels out, so remains minus f x hat so this is minus and f x hat is W hat transpose phi where we are trying to update what is this W hat. We have to find out a law by which we have to update this W hat plus k v r plus lambda 1 e n minus 1 plus dot dot dot lambda n minus 1 e ; first derivative which is already here x n d dot. If I now bring this x n to this side so x n d dot minus x n will be here we can write this is e to the power n so e to the power nth derivative of e is x n desired minus x n dot x n desired dot minus x n dot is nth derivative of e you can see that and then what is happening? Here, minus x n dot; so this is this e n so e nth derivative of

e plus lambda 1 e n minus 1th derivative and so on. This total quantity is actually r. What you have now W transpose phi minus W hat transpose phi plus k v r dot r dot is 0. This is what we will find now.

(Refer Slide Time: 21:05)



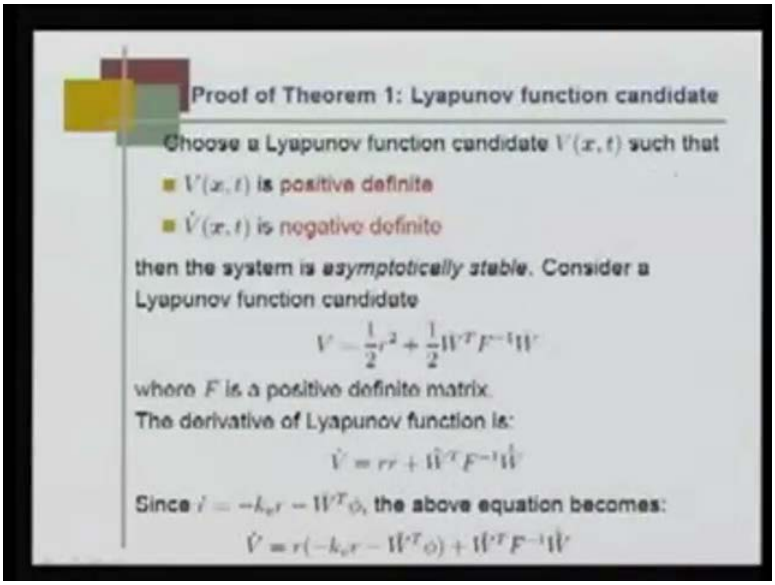
Defining W hat W tilde transpose is W transpose minus W hat transpose we can write the same expression which is r dot. You can easily see that earlier we saw that this is k r plus this is k v r plus r dot and this is W tilde transpose phi so that is exactly here. r dot is minus k v r minus W tilde transpose phi.

Whatever we did earlier, just to summarize we define this - x_n dot becomes W tilde transpose phi k v r into this you bring in x_n dot to this side or you can take to the other side also this side and then you can write that this is e to the power nth derivative of error is minus W tilde transpose phi this k v r will go to the other side minus k v r and this all this terms differentiating r with respect to time you get r dot is this quantity substituting e_n with into the above equation you get this entire quantity and finally r dot is minus k v r minus W tilde transpose phi; but it can be simply also done.

Proof of theorem: now how do we show that it is stable? You see that problem is here this we have to find out a law that stabilizes the system. So let f x then the control law will stabilize the system in the sense of stabilize if W dot we have to find out actually what

should be the W hat this function is approximated by radial basis function network and then weight has to be updated. What is that update law for the weights such that this control law will stabilize? This is the question.

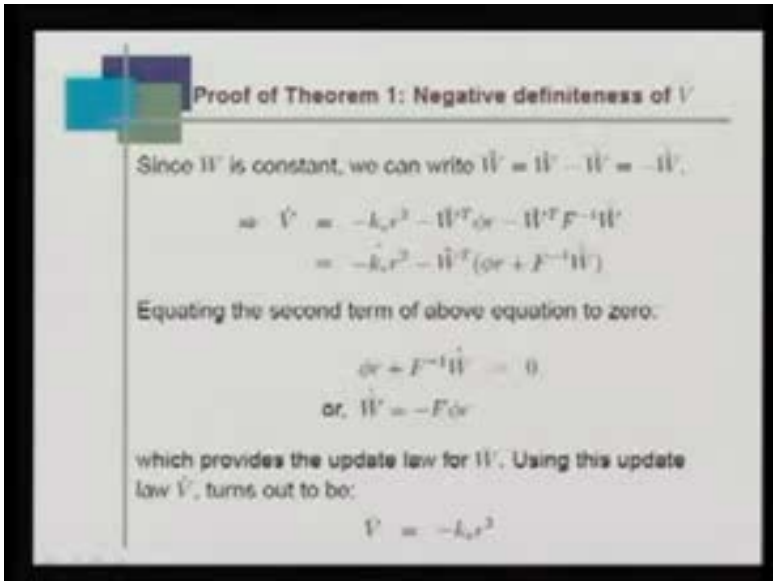
(Refer Slide Time: 23:48)



Choose a Lyapunov function candidate V x t such that V x t is positive definite V dot x t is negative definite then the system is asymptotically stable. Now consider a Lyapunov function candidate for this system is half r square; this is filtered tracking error r is filtered tracking error plus half W tilde transpose F inverse W tilde where F is a positive definite matrix.

A derivative of the Lyapunov function is V dot equal to r r dot plus you can easily see 2 cancels out r r dot rate derivative of this function my Lyapunov function. I am trying to differentiate this with respect to time. The first term is r r dot plus W tilde transpose F inverse W tilde dot. So this F is some positive definite matrix this F, you can take any positive derivative simply even the identity matrix will do. Since r dot is minus k v r minus W tilde transpose phi which we have already derived the above equation becomes V dot is r minus k v r minus W tilde transpose phi. So this r dot is multiplied here and then this term.

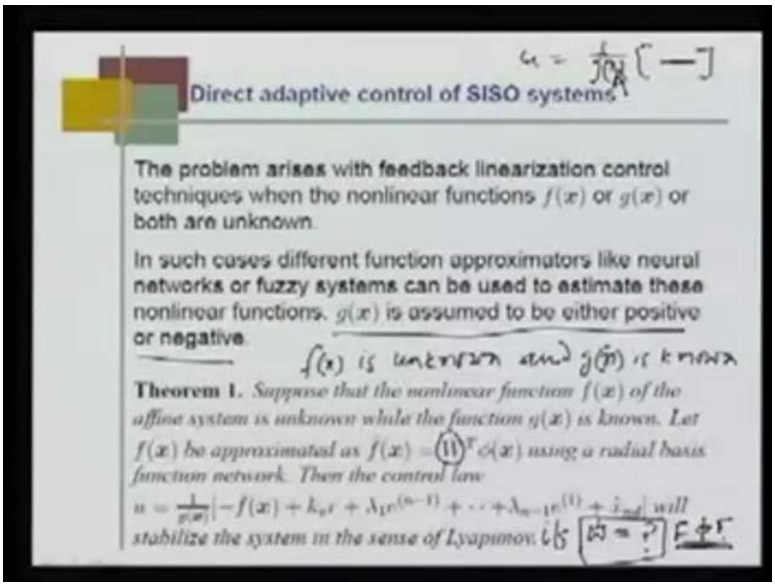
(Refer Slide Time: 25:24)



W is constant you can write W tilde dot is simply minus W hat dot. This W hat is actually the actual weights of the radial basis function network that is supposed to be approximating what is f x. V dot is minus k v r square minus this is your minus k v r square and minus W tilde transpose phi r so this is what is the minus k v r square minus W tilde transpose phi r and the last term which is W tilde transpose F minus earlier it was W tilde dot plus so W tilde dot plus, plus W tilde dot is minus W hat dot.

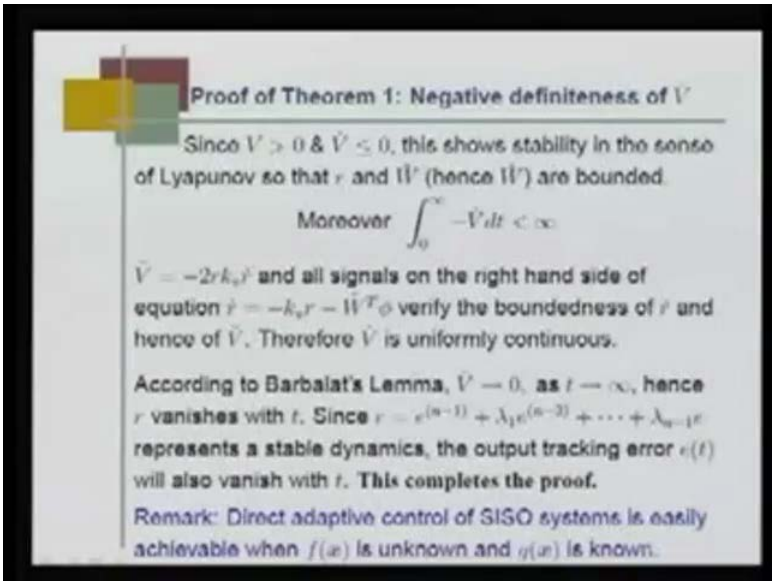
That is why this minus is coming and this is W hat dot. So I can club these two terms as W tilde transpose phi r plus F hat W hat dot. What we need is that the system will be stable if the rate derivative of the Lyapunov function is negative definite. For that if I can cancel this term or if I can make this term to be 0, then V dot can be minus k v r square. For that to happen this term should become 0. From here W hat dot is minus F phi r which provides the update law for W hat using this update law V dot it turns out to be V dot is minus k v r square so this gives us stability. So this stability comes when W hat dot the rate derivative of the W or the weight update law for the radial basis function network; that is, approximating the f x the unknown f x should be F phi into r.

(Refer Slide Time: 27:47)



So if you go back I ask this question that will stabilize the system in the sense of Lyapunov if W hat dot is F phi r; so this negative sign is also there so this is minus F phi r. We completed the proof of the first part now we will go to the second part.

(Refer Slide Time: 28:27)

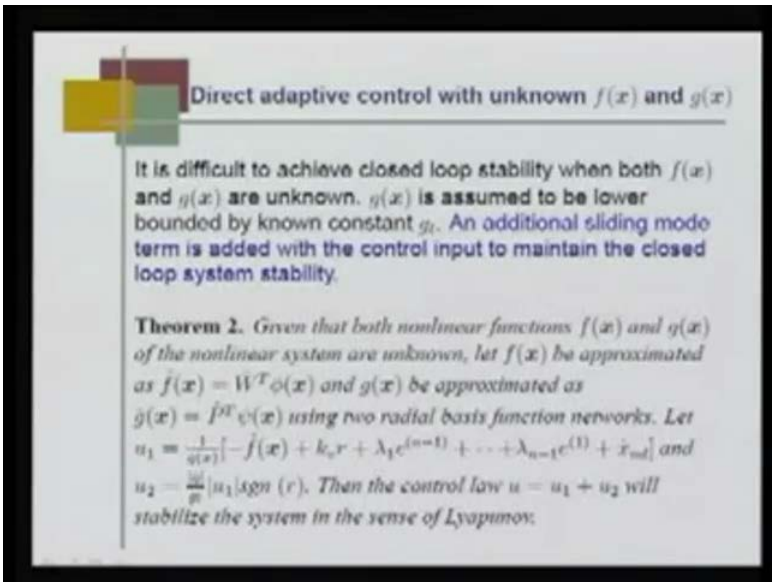


There is also Barbalat's Lemma: you can apply V dot tends to 0 as t tends to infinity then r vanishes with t; since r is this function represents a stable dynamics given lambda 1 and

lambda n minus 1, they are all positive quantities the output tracking error e t will also vanish with t; this completes the proof.

Remark: Direct adaptive control of single input single output systems is easily achievable when f x is unknown and g x is known.

(Refer Slide Time: 29:08)



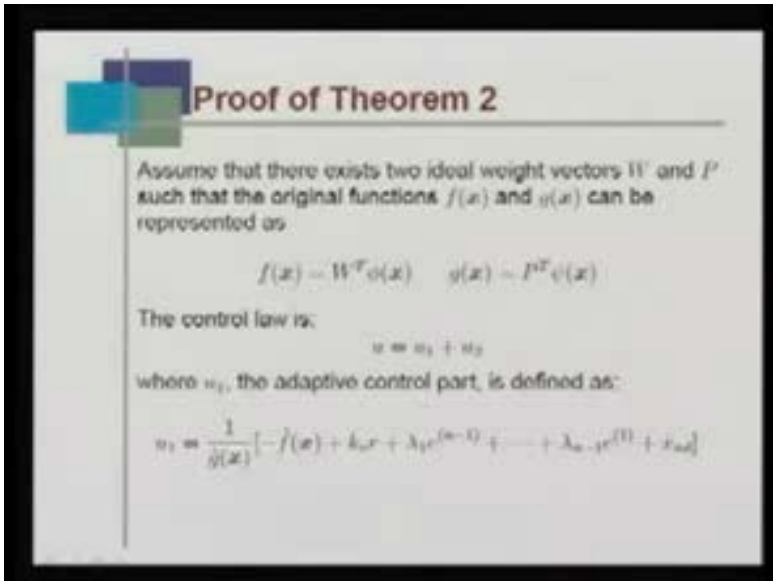
Now, direct adaptive control with unknown f x as well as unknown g x: it is difficult to achieve close loop stability when both f x and g x are unknown. g x is assumed to be lower bounded by known constant g_l. An additional sliding mode term is added with the control input to maintain the closed loop system stability. Theorem 2: given that both nonlinear functions f x and g x of the nonlinear systems are unknown. Let f x be approximated as f hat x is W hat transpose phi x; that means, f x has been approximated by some neural network; radial basis function network. This is different than system identification, because here, we assume that a specific neural network is approximated, but we are not identifying what is f x and g x be approximated as g hat x is P hat transpose phi x. This is again, the weights of another radial basis function network represented by P using two radial basis function network.

Let u_1 be a control law like earlier first part that we showed for unknown f x but known g x, where 1 upon g hat x minus f hat x plus k v r; plus this tracking error terms plus x n d desired trajectory x n dot desired. u_2 is a sliding mode term which is given as g hat absolute term by g l is the lower bound of g u_1 we assume that g_l is also known. u_1 is the magnitude of absolute magnitude of u_1 sine function of r. then the control law u equal to u_1 plus u_2 will stabilize the system in the sense of Lyapunov, provided if the weight update law for this for the approximating f x is W hat dot is minus F phi r and P hat dot is minus g psi u_1. So, this part is missing in this slide. Please note kindly try to understand what this theorem means. We are given an affine system both f x and g x are unknown f x is approximated by 1 radial basis function network; not in terms of system identification.

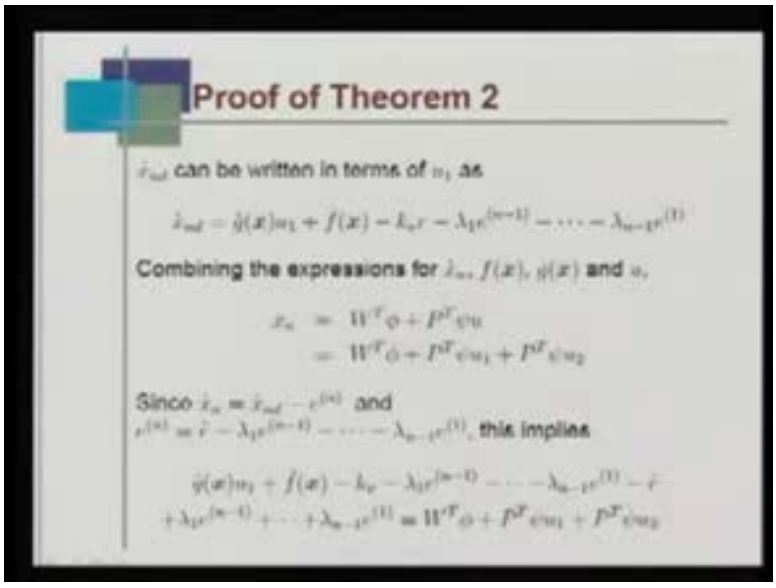
We simply say that this is representing that and g x by another radial basis function network. Then the control law u_1 and u_2 which are defined here will stabilize this affine system. What is this affine system? If W hat dot is minus F phi r where F is a positive definite matrix and P hat dot is minus g psi u_1 r, where g is again another positive definite matrix. phi is the basis function for approximating f x and phi is the basis function for approximating g x.

Proof of the theorem 2; it is similar. Again assume that there exist some optimal weight W as well as P such that f x can be exactly written as W transpose phi x and g x also can be exactly written as P transpose phi x; a control law is u equal to u_1 plus u_2.

(Refer Slide Time: 33:48)



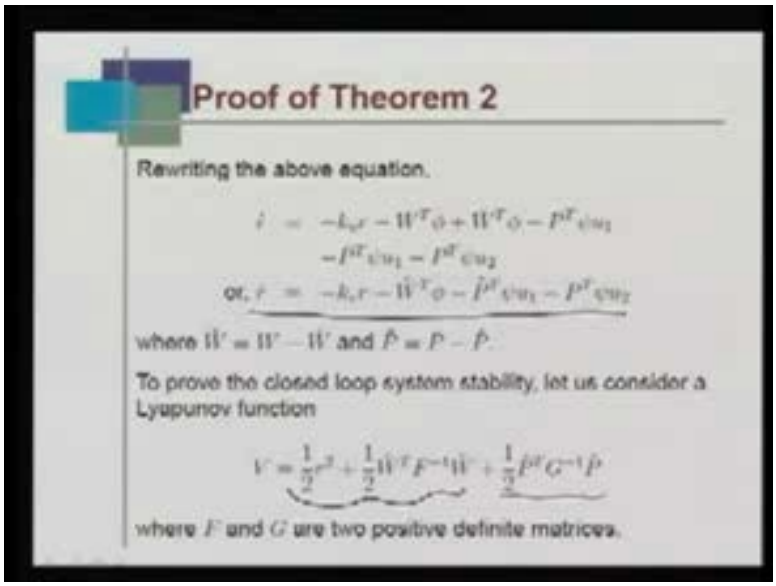
(Refer Slide Time: 39:18)



Where u_1 is the adaptive control part and defined by this particular term, adaptive because here, f x and g x are updated. The weights are this. f x is approximated by W. so when I put this hat means I have a update law for W. Hence these are all varying from, so this that is why this is an adaptive control law because this controller parameters are adaptive so as to always ensure the tracking error converges to 0. Again we can do the same take the control law, put in the affine system, find out the closed loop error

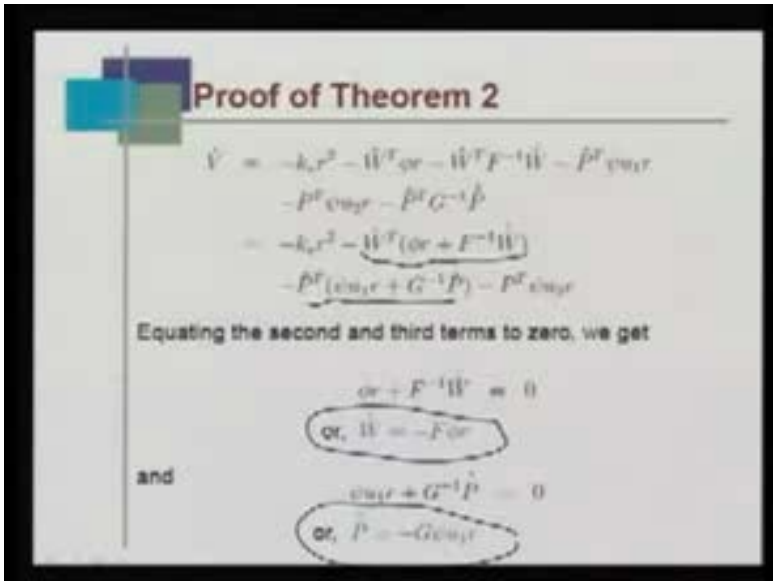
dynamics. If you do, this is your x_n d dot; this one can be written in terms of u_1 and this one combining the expression for x_n dot f x g x and u. You get this particular term since x_n dot is x_n d dot minus nth derivative of e. I have already shown you that nth derivative of e is r dot and this quantity actually r dot is e n plus. This implies that from here, you get this particular expression here. Further simplifying, rewriting the above equation you get r dot if this particular expression minus k v r minus W tilde transpose phi minus P tilde transpose psi u_1 minus P transpose psi u_2; where W tilde is W minus W hat P tilde P minus P hat the way we did last time. The proof to prove the closed loop system stability.

(Refer Slide Time: 35:48)



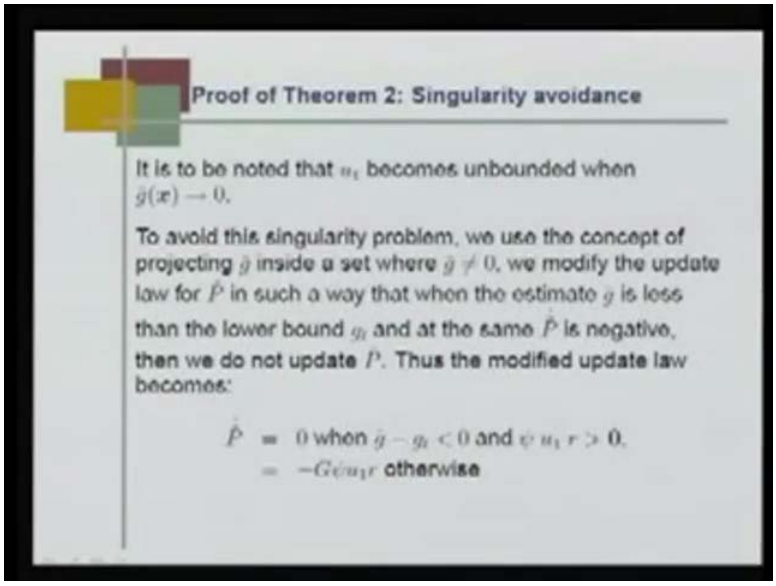
Let us consider a Lyapunov function of this style, where earlier we considered this to be the Lyapunov function and because we have again another unknown g x so that is again added to the Lyapunov function. You take the rate derivative of the Lyapunov function you get r r dot plus W tilde transpose F inverse W tilde dot P tilde transpose G inverse P tilde dot. Substituting r dot into the above equation you get V dot is r. This is your r dot. This r dot we found out earlier in the closed loop dynamics, so this particular expression if you place here in this, then rate derivative of V dot becomes this. Since W and P are constant you can always write W tilde dot is simply minus W hat dot and P tilde dot is minus P hat dot. So, now recombining, this is your V dot.

(Refer Slide Time: 37:15)



If you re-combine rate derivative of Lyapunov function, you get one combination here. One combination; this is intelligently done so that we can find an update law for W hat dot. Another update law for P hat dot and we can make these two terms 0o and that is done by this update term and this update term. So, this is there already in the theorem if W hat dot is minus F phi r and P hat dot is minus G phi u_1 r then this is Lyapunov stable. But we have not actually proved, because even if we do that V dot is minus k v r square minus P transpose phi u_2 r. There is an extra term here. So, we found the condition for weight update for that part of neural network which is representing f x and the other part of the neural network that is representing g x. We have two controllers. This additional control law which is $u_{2;}$ due to that here…

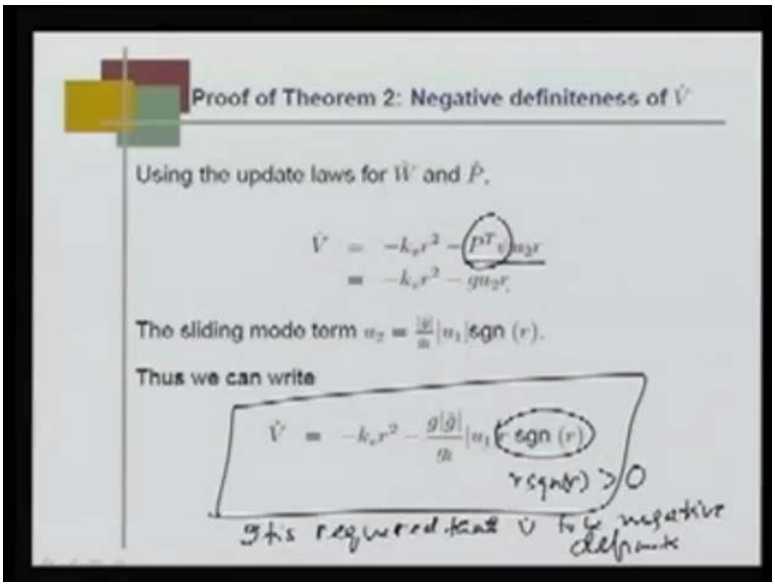
(Refer Slide Time: 38:45)



Before I discuss that here, it is a singularity avoidance. It is to be noted that u_1 becomes unbounded when g hat x tends to 0 which is very obvious you can easily see that u_1...

This u_1 will be unbounded if g hat x tends to 0. So, just to avoid this, because if it becomes unbounded, a control input cannot actuate to any physical system; any unbounded input has to be finite and bounded to ensure that what we do, to avoid the singularity problem. We use the concept of projecting g hat inside a set where g hat is not equal to zero. We modify the update law for p hat in such a way that when the estimate g hat is less than the lower bound g_l and at the same time at the same time P hat dot is negative then we do not update P hat thus the modified update law is P hat dot is 0 when g hat minus g l is less than 0 and psi u_1 r is greater than 0. Otherwise it is because this u_1 is a very large quantity. Such an update would destabilize the system.

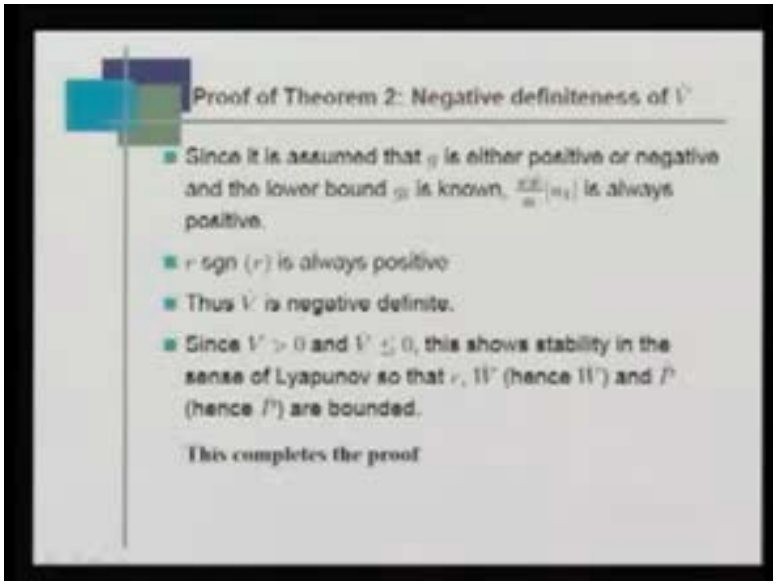
(Refer Slide Time: 40:41)



Using the update law for W hat and P hat, we finally get the rate derivative of the Lyapunov function to be minus k v r square and this extra term which is P transpose psi u_2 r. This P transpose is actually the actual g which is unknown into u_2 r. How do I ensure that V dot is actually negative definite? In this, the u_2 is the sliding mode term written by g hat by g_1 into u_1 r given that this is my final expression for V dot and it is required that V dot to be negative definite.

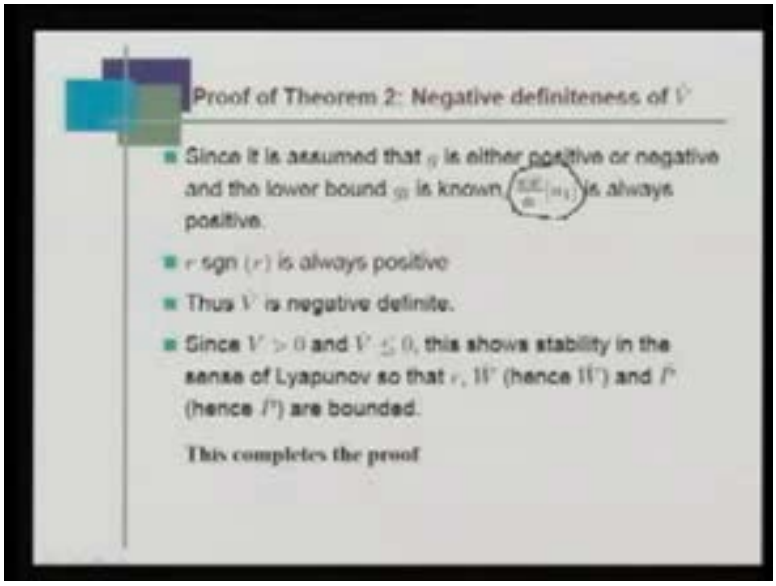
How do I show it? You can easily see r sign r this term is always positive if r is negative this is negative so positive and r is positive the sign also positive so it is positive; now we have to show that g hat u_1 upon g l is also positive that argument is here.

(Refer Slide Time: 42:18)



Since it is assumed that g is either positive or negative and the lower bound g_l is known. So g_l will have same sign as of g; this is always a positive quantity if g is positive g_l is also positive if g is negative g_l is also negative. Hence this quantity is always positive; this is an assumption again because we cannot solve for a generic case. The most important assumption here is that g is either positive or negative that is the only assumption. That is, g should not have value both positive and negative. This part is positive r sign r is always positive hence V dot is negative definite. Since V is greater than 0 and V dot is negative definite, this shows stability in the sense of Lyapunov so that r W tilde and P tilde are bounded.

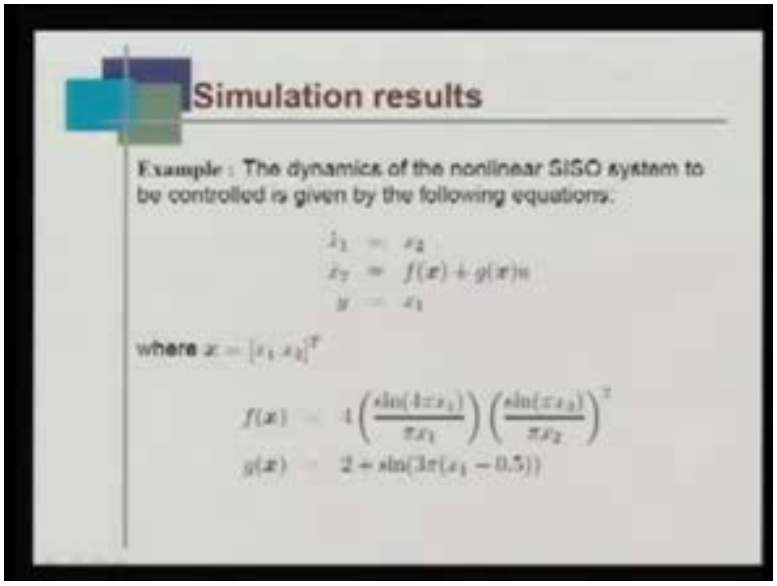
(Refer Slide Time: 43:28)



This completes the proof. What we proved actually? Again I go back; I showed to you what we actually proved just now (Refer Slide Time: 43:42). This is the proof that given an affine system which is x_1 dot is x_2 and so on until x_n dot is f x plus g x u, given this affine system. This u is consisting of 2 terms: u_1 plus u_2 where u_1 is given by this linear feedback linearization term structure u_1 has a feedback linearization controller structure. But this is adaptive because it consists of 2 neural networks that are approximating those 2 networks. One is approximating f x another is approximating g x and the other control signal which we say sliding mode term is u_2 which is this particular term. If we have this u the control law summation of these two control law and the adaptive control law has parameters W here and P here and the adaptive law is W hat dot is minus F phi r P hat dot is minus G psi u_1 r.

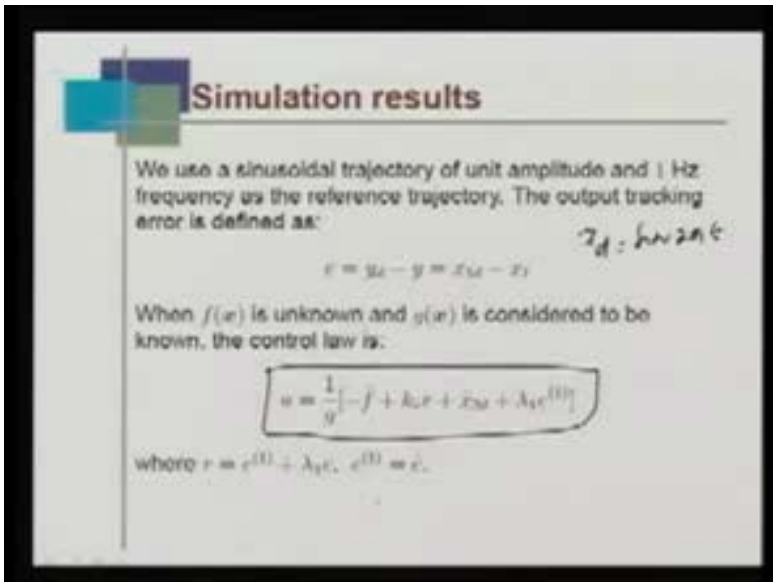
If the parameters of these 2 neural networks were updated according to this law then, this control law u equal to u_1 plus u_2 will stabilize the system. That is this will be Lyapunov stable. This is very important. What we discussed today is a very generic understanding of a direct adaptive control of an affine system. Now we will validate this controller simulation theoretical validation is already there; now we will do the simulation validation before we go for experimentation.

(Refer Slide Time: 46:00)



For simulation validation we take a single input single output system, where x_1 dot is x_2 this is a second order nonlinear system. x_2 dot is f x plus g x u y equal to x_1 where f x is given by this nonlinearity which is 4 sine 4 phi x_1 by phi x_1 into sine phi x_2 by phi x_2 whole square and g x is given by this term. You can easily see here, g x is a positive quantity, it can never become negative.

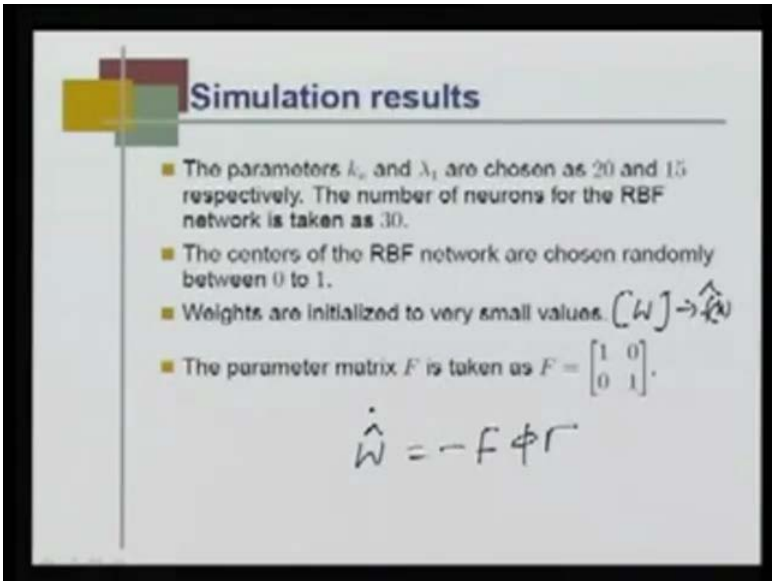
(Refer Slide Time: 46:34)



We use a sinusoid trajectory of unit amplitude and 1 hertz frequency as the reference trajectory. The reference trajectory x_1 d is a sinusoid trajectory with frequency 1 hertz; that is x d is sine 2 phi t, this is the my reference trajectory and e is y d minus y which is x_1 d minus x_1 in state space. When f x is unknown and g x is considered to be known as per our theorem your e is 1 upon g minus f hat plus k v r x_2 dot d lambda 1 e 1.

This is our usual control law that we have already defined and r is first derivative of e plus lambda 1 e the c parameters k v and lambda 1. If we look here in this controller, the 2 external parameter 1 is k v another is lambda 1.

(Refer Slide Time: 48:00)

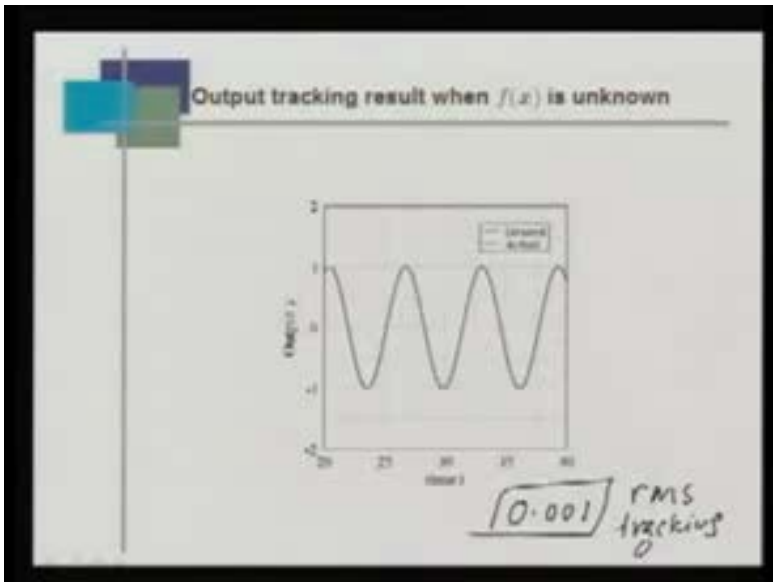


That is 20 and 15 the number of neurons in the radial basis function network is 30. The centers of the radial basis function network are chosen randomly between 0 to 1. You see that now what we are trying to do here, we are you can easily see that my f x is a function of x_1 and x_2 and g x is a function of only x_1. My radial basis function network as 2 inputs x_1 and x_2 because I do not know what is g x so still I will keep my radial basis function network 2 inputs to be x_1 and x_2. And the center has dimension d2 by 1 and each of this value is randomly assigned between 0 to 1.

The weights are initialized to very small values. That is W to approximate f x and the parameter matrix F is taken as identity matrix. As you know that our W hat dot the

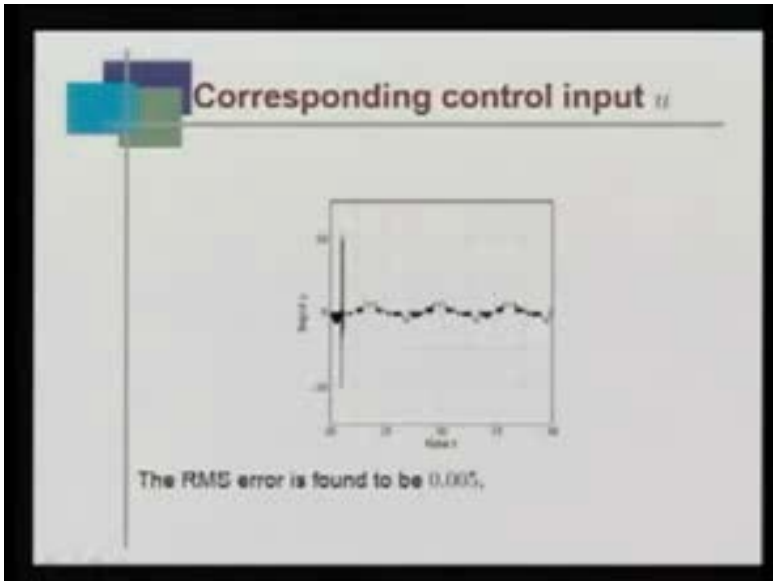
weight update law for this approximating effect x we found out to be minus F phi r. And we also took this phi function to be Gaussian function.

Refer Slide Time: 49:39)



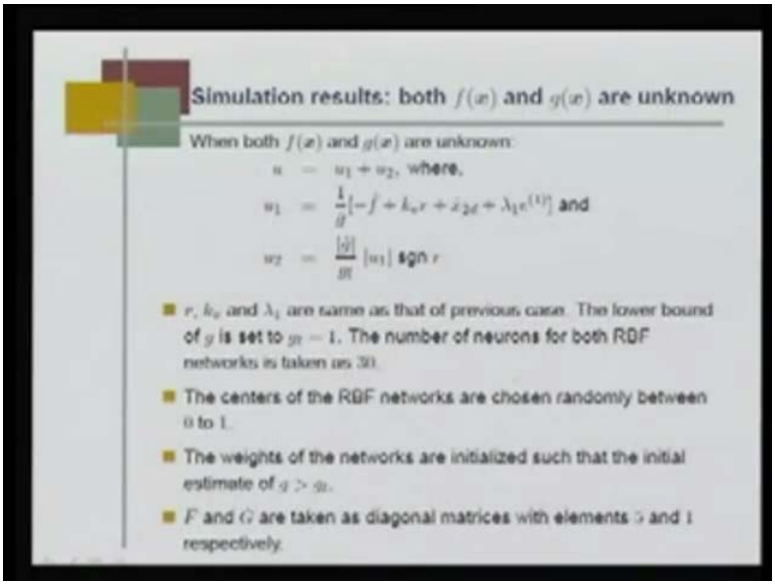
Taking that you see that the tracking is very exact, the desired and actual they almost match. In fact, the tracking error the R M S tracking error here is 0 point 001.

(Refer Slide Time: 50:02)



This is the control input. That is you can easily see how a controller is very… because a nonlinear system sure the controller looks like (50:19) highly nonlinear term.

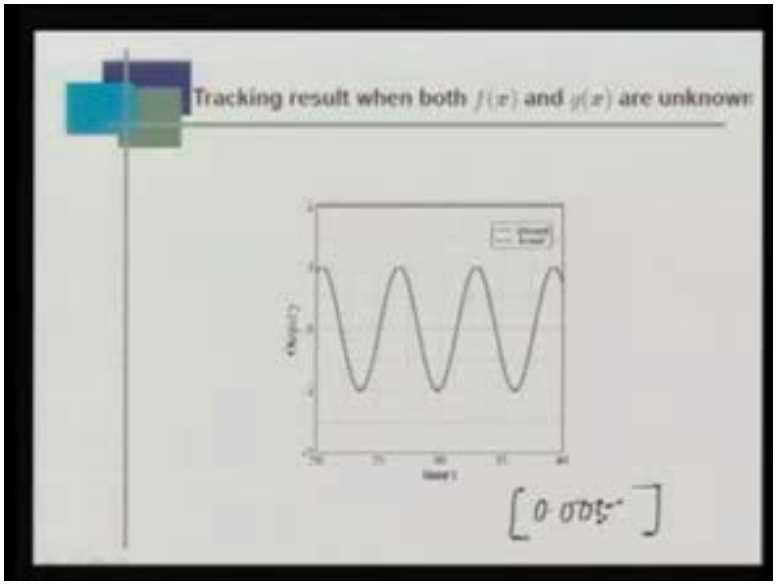
(Refer Slide Time: 50:25)



That is when f x is unknown; we assume that g x to be known. Now we assume both f x and g x are unknown.

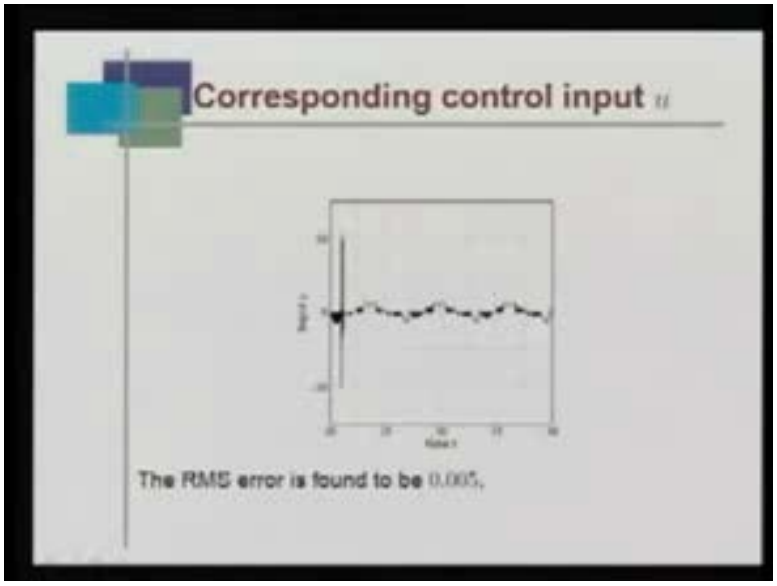
There in the first part, we put radial basis function network for only f x, g x was known. Now we have to put both f x and g x to radial basis function network and our control law u is u_1 plus u_2 which is where u_1 is this 1 and u_2 is given. This is the sliding mode term r k v and d- lambda 1 are same as that of previous case, lower bound of g is set to g_l equal to 1, the number of neurons for both R B F network is taken as 30. The centers of the R B F network are chosen randomly between 0 and 1. The weights of the network are initialized such that the initial estimate of g is greater than g_l. F and G are taken as diagonal matrices with elements phi and 1 respectively.

(Refer Slide Time: 51:44)



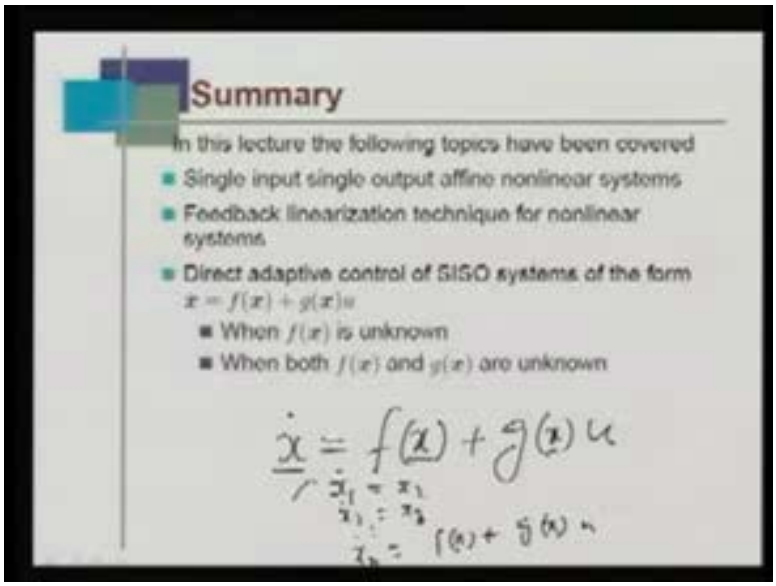
After that when we do the simulation, this is desired and actual and the R M S tracking error has slightly increased in this case to be 0 point 005, but you can easily see that in a macro scale you cannot see that there is a difference between the desired and actual trajectory. So, the desired trajectory sinusoid and actual trajectory is also sinusoid.

(Refer Slide Time: 51:55)



Here you see that the controller is highly fluctuating and also a lot of oscillation. This is happening, because it is expected we have assumed that g x is unknown.

(Refer Slide Time: 52:18)



Summary: In this lecture, the following topics have been covered; single input single output affine nonlinear system, feedback linearization technique for nonlinear system, direct adaptive control of single input, single output system of the form x dot equal to f x plus g x u when f x is unknown, when both f x and g x are unknown. To summarize, you learnt today, affine system which has a form in general and affine system has a form f x plus g x u in general. The most simplified version, if you can still write, here in this case, this is a vector this is a vector and this is a vector, but the easiest way even the most simplified affine form is x_1 dot is x_2, x_2 dot is x_3 and so on. x_n dot is f x plus g x u; this is even simpler than this. So, this form is easier; is easier to analyze than this form.

Thank you very much.