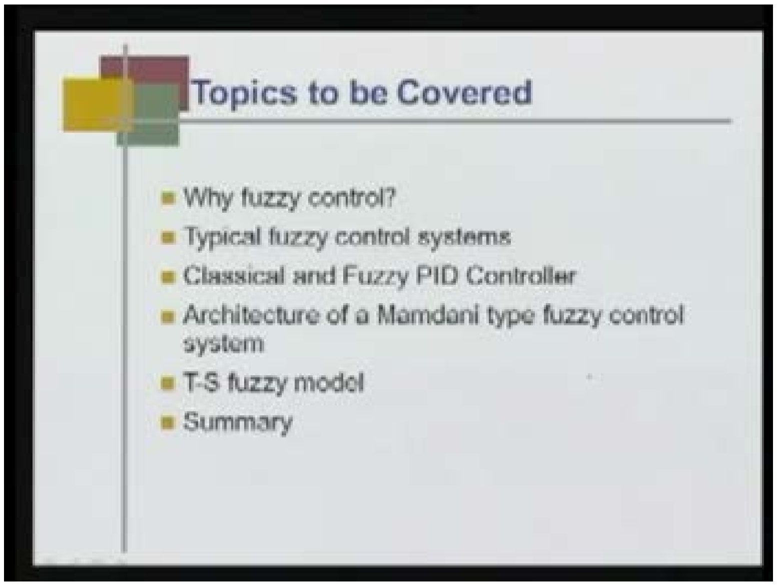
Intelligent Systems and Control

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

Module – 2 Lecture – 4
Introduction to Fuzzy Logic Control

In this lecture today, we will be discussing fuzzy logic control. It is the fourth lecture in this module. Earlier, we discussed fuzzy sets, fuzzy relation, and then using relation matrix and the rule of composition, compositional rule of inference, how do we infer about the consequence in a rule base. Last class, we learnt about fuzzy rule base and the inference mechanism using approximate reasoning. We are now ready to understand the concept of fuzzy logic controller.

(Refer Slide Time: 01:13)

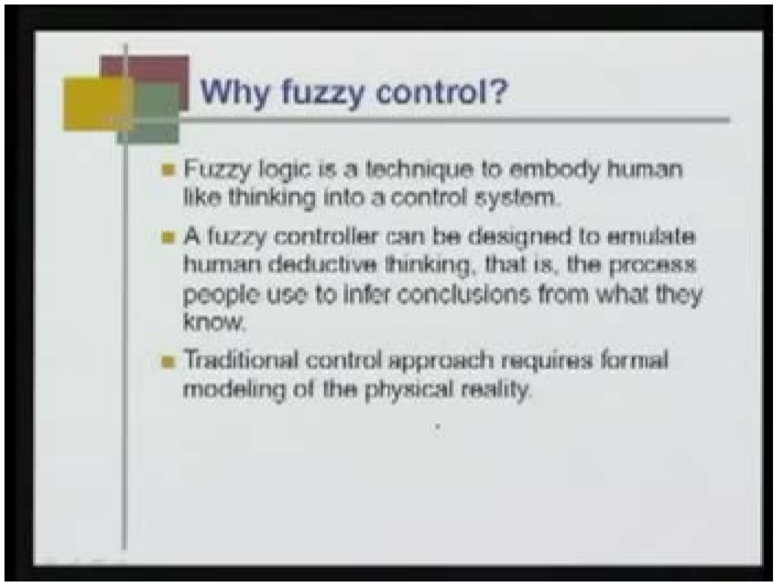


Topics to be Covered

- Why fuzzy control?
- Typical fuzzy control systems
- Classical and Fuzzy PID Controller
- Architecture of a Mamdani type fuzzy control system
- T-S fuzzy model
- Summary

In the lecture today, we will be covering why are we after fuzzy logic controller, why we are interested in fuzzy logic controller or why it is interesting; typical fuzzy control systems, classical and fuzzy PID controller, and architecture of a Mamdani type of fuzzy control system. In this lecture, we will be basically dealing with two kinds of architecture: Mamdani type fuzzy control system and T-S fuzzy model (these two have become very popular in the fuzzy control literature) and then a summary of what we will teach today.

(Refer Slide Time: 01:56)



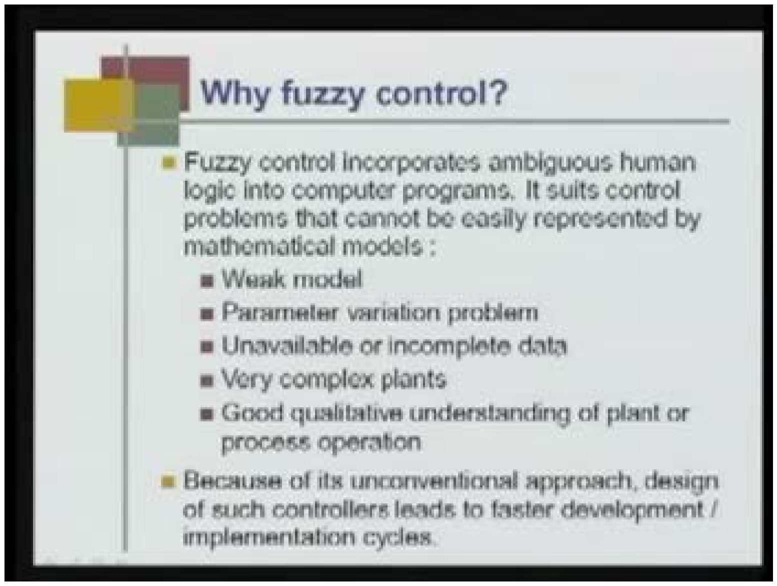
Why fuzzy control?

- Fuzzy logic is a technique to embody human like thinking into a control system.
- A fuzzy controller can be designed to emulate human deductive thinking, that is, the process people use to infer conclusions from what they know.
- Traditional control approach requires formal modeling of the physical reality.

Why fuzzy control? Fuzzy logic is a technique to embody human-like thinking into a control system. A fuzzy controller can be designed to emulate human deductive thinking, that is, the process people use to infer conclusions from what they know. The traditional control approach requires formal modeling of the physical reality. We know that there are many systems in which human operators work. When they work, they operate the plant based on their experience; better the experience, better the control mechanism.

Can we utilize such principles of human ways of decision-making into designing control systems? Or can our control system behave the way human operators control certain physical devices based on their experiences? Although there are various reasons why we talk about fuzzy control, here are some of the important parameters why scientifically fuzzy logic control has been accepted in control literature.

(Refer Slide Time: 03:35)



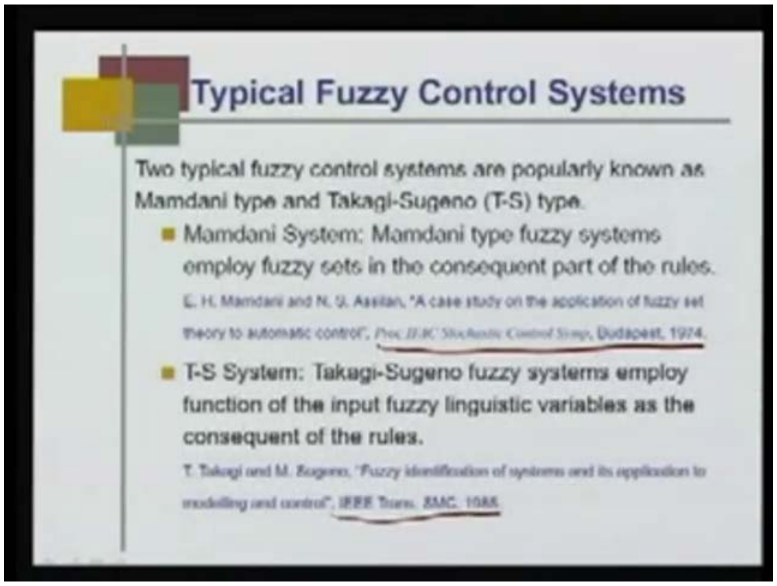
Why fuzzy control?

- Fuzzy control incorporates ambiguous human logic into computer programs. It suits control problems that cannot be easily represented by mathematical models :
  - Weak model
  - Parameter variation problem
  - Unavailable or incomplete data
  - Very complex plants
  - Good qualitative understanding of plant or process operation
- Because of its unconventional approach, design of such controllers leads to faster development / implementation cycles.

Fuzzy control incorporates ambiguous human logic into computer programs. It suits control problems that cannot be easily represented by mathematical models. For example, there are systems for which we have very weak models. That means the physics of the systems are very poorly understood. Then, there are physical systems where the parameter variations are arbitrary and cannot be known a priori, or there are systems where the data obtained from the systems are incomplete or unavailable.

Sometimes, the plant may be very complex in the sense even if I know the physics of the system, if I want to build the mathematical model for this complex plant, the model would be very complex. Another reason could be for fuzzy logic control is good qualitative understanding of plant or process operation. Because of its unconventional approach, design of such controllers leads to faster development or implementation cycles.

(Refer Slide Time: 05:05)



Typical Fuzzy Control Systems

Two typical fuzzy control systems are popularly known as Mamdani type and Takagi-Sugeno (T-S) type.

- Mamdani System: Mamdani type fuzzy systems employ fuzzy sets in the consequent part of the rules.

  E. H. Mamdani and N. S. Assilan, "A case study on the application of fuzzy set theory to automatic control", *Proc. IFAC Stochastic Control Symp.*, Budapest, 1974.

- T-S System: Takagi-Sugeno fuzzy systems employ function of the input fuzzy linguistic variables as the consequent of the rules.

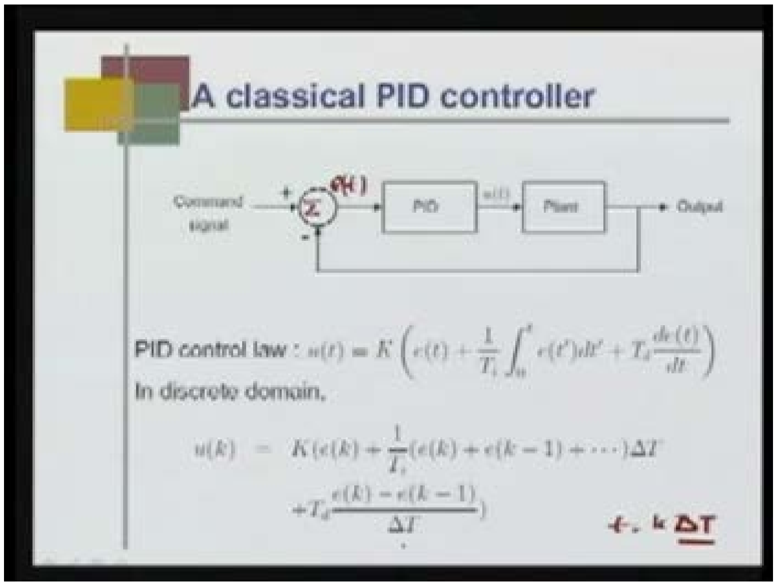
  T. Takagi and M. Sugeno, "Fuzzy identification of systems and its application to modelling and control", IEEE Trans. SMC, 1985.

Today, we will be talking about two categories of fuzzy control systems. These two typical fuzzy control systems are popularly known in control literature as Mamdani type and Takagi–Sugeno type (T-S for short form). Mamdani type fuzzy systems employ fuzzy sets in the consequent part of the rules. This is a fuzzy logic controller, the Mamdani type fuzzy logic controller. What they do is that the consequent part itself takes the control action; the incremental control action is described in the consequent part of each rule.

For those who are more interested to know why Mamdani type of system, we say because it is their first paper – Mamdani and Asslan, ‘A case study on the application of fuzzy set theory to automatic control’. This is the first paper on fuzzy control that has been recorded. This is the first one that has been recorded way back in 1974. Most of the fuzzy logic controllers follow a very close format as that of the Mamdani FLC presented in this paper. Then, those who are interested can look at this reference to ‘A case study on the application of fuzzy set theory to automatic control’ (Refer Slide Time: 06:49). That is why these are Mamdani type controllers.

The next is T-S fuzzy system, Takagi–Sugeno fuzzy systems. They employ function of the input fuzzy linguistic variable as the consequent of the rules. What happens here is that a fuzzy dynamic model is expressed in the form of local rules. Local rule means if there are two variables x_1 = a and x_2 = b, then the plant dynamics can be represented either as a linear dynamical system or a nonlinear dynamical system, as a known dynamical system. Those who are interested to know the origin of Takagi–Sugeno model can follow this particular link, that is, IEEE Transaction SMC, 1985 (Refer Slide Time: 07:59). Takagi and Sugeno published a paper on ‘Fuzzy identification of systems and its application to modeling and control’.

(Refer Slide Time: 08:09)



A classical PID controller

PID control law : $u(t) = K\left(e(t) + \frac{1}{T_i}\int_{0}^{t} e(t')dt' + T_d\frac{de(t)}{dt}\right)$

In discrete domain,

$$u(k) = K(e(k) + \frac{1}{T_i}(e(k) + e(k-1) + \cdots)\Delta T + T_d\frac{e(k) - e(k-1)}{\Delta T})$$

$t = k\underline{\Delta T}$

Now, I will introduce you to how we convert or how we can design a Mamdani type of fuzzy logic controller. The first part we will be discussing is Mamdani type of fuzzy logic controller. You know that fuzzy logic controller means we are designing the controller for a nonlinear system, the plant is unknown and cannot be modeled properly. There are various other constraints about the system dynamics.

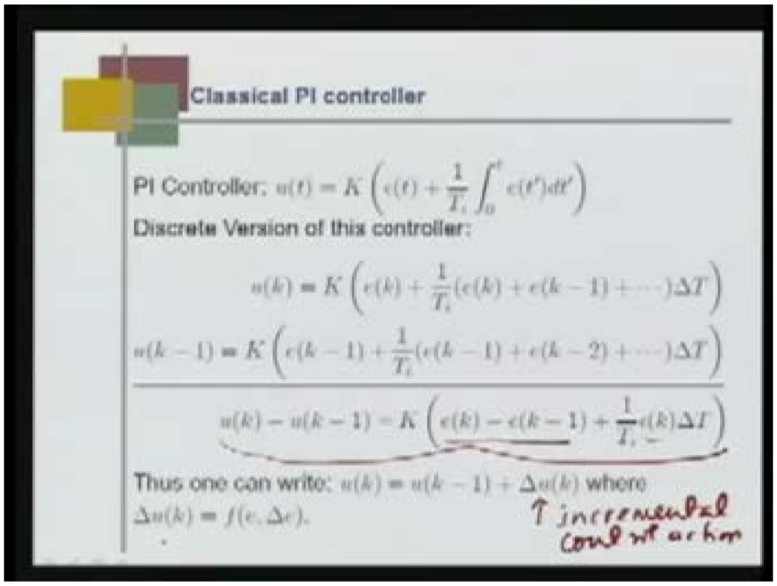
What you are seeing here is a simple feedback control system where you are seeing a plant, a PID controller, u t is the control action, this is the output, this is the unity feedback and this is the command signal. The objective of this control system is that the output should follow the command signal. Whatever is the command signal, the plant output should follow that as accurately as possible. What is the PID controller? The PID controller u t is proportional gain K into e t. This is e t (Refer Slide Time: 09:29). So e t plus 1 upon T_i 0 to t e t dash d t dash. This is an integral action.

You can easily see that the error is directly multiplied with K and here, the error has been integrated over time and there is a time constant associated with it, T_i. Similarly, this error is differentiated in this term d e t by dt and T_d. This is your PID controller; this is very well known. You will be surprised to know that in spite of varieties and very novel schemes that have appeared in the literature and have been implemented in real time, still, most of the systems, most of the industrial processes employ PID controllers.

If I write this PID controller in the discrete domain, you have an expression like this. Discrete domain means here t is represented as k delta T. delta T is the sampling time interval, k is the sampling instant and so, t is k delta T. For the sake of convenience, we just remove delta T and then, we write the discrete form of the equation.

The k th sampling instant means at t equal to k delta T, u k is K e k into 1 upon T_i. If I want to integrate this e t dash d t dash, how do I write? e K delta T plus e K minus 1 delta T plus e K minus 2 delta T until e 0, the initial value of error, into delta T plus this is T_d into this d et by dt, if I want to differentiate using a backward difference model, then it will be e K minus e K minus 1 upon delta T. This is using the difference method or Euler's difference method. Using the difference method or Euler's difference method, this is the difference d e t upon dt. This is the recursive form of u k. Why am I doing it? To show you that this PID controller has a very interesting structure. Just wait for a moment. Just see here.

(Refer Slide Time: 12:29)



Classical PI controller

PI Controller: $u(t) = K\left(e(t) + \frac{1}{T_i}\int_0^t e(t')dt'\right)$

Discrete Version of this controller:

$$u(k) = K\left(e(k) + \frac{1}{T_i}(e(k) + e(k-1) + \cdots)\Delta T\right)$$

$$u(k-1) = K\left(e(k-1) + \frac{1}{T_i}(e(k-1) + e(k-2) + \cdots)\Delta T\right)$$

$$\underbrace{u(k) - u(k-1) = K\left(\underline{e(k) - e(k-1)} + \frac{1}{T_i}e(k)\Delta T\right)}$$

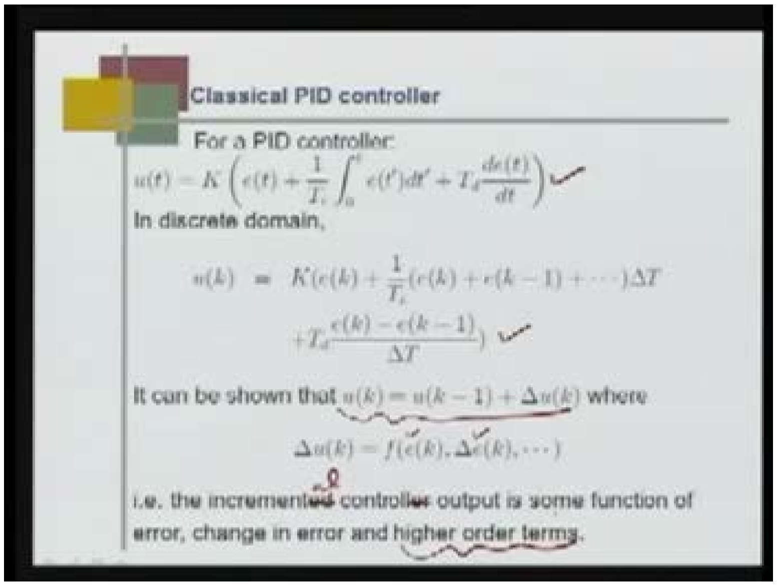
Thus one can write: $u(k) = u(k-1) + \Delta u(k)$ where $\Delta u(k) = f(e, \Delta e)$.

$\uparrow$ incremental control action

I made a little simplification in the controller structure removing the derivative part. So, I have now the proportional part and the integral part. This is a PI controller. The discrete version of this at the k th instant, I would write u k equal to K e k plus 1 upon T_i e k plus e k minus 1 and so on into delta T. Similarly, I can write this equation at the k minus 1 th instant to be u k minus 1 equal to K e k minus 1 plus 1 upon T_i e k minus 1 plus e k minus 2 and so on until delta T.

Now, if I subtract this equation from this equation, what do I get? u k minus u k minus 1 is K (this is the proportional gain) into e k minus e k minus 1. You can easily see e k minus e k minus 1 and here, e k minus 1, this will be 0 and all these terms will be cancelled, only one term will appear and that is e k by T_i into delta T. So what you are finding is that if I now write this expression (Refer Slide Time: 13:44), I will write that u k equal to u k minus 1, that is, the present control action is equal to previous control action plus the incremental control action. I would say this is the incremental control action, where you can easily see that this delta u k is some function of e and the change in error, because you know that these are all constants; K, T_i and delta T are constants and so, you can easily say delta u k, the incremental control action, is a function of error and change in error. This is the very basis or this is the very foundation on which… this specific understanding led to innovation of fuzzy logic control rule base and how the fuzzy logic control rule base is structured. I will explain that.

(Refer Slide Time: 15:06)



Classical PID controller

For a PID controller:

$$u(t) = K\left(e(t) + \frac{1}{T_i}\int_0^t e(t')dt' + T_d\frac{de(t)}{dt}\right) ✓$$

In discrete domain,

$$u(k) = K(e(k) + \frac{1}{T_i}(e(k) + e(k-1) + \cdots)\Delta T + T_d\frac{e(k) - e(k-1)}{\Delta T}) ✓$$

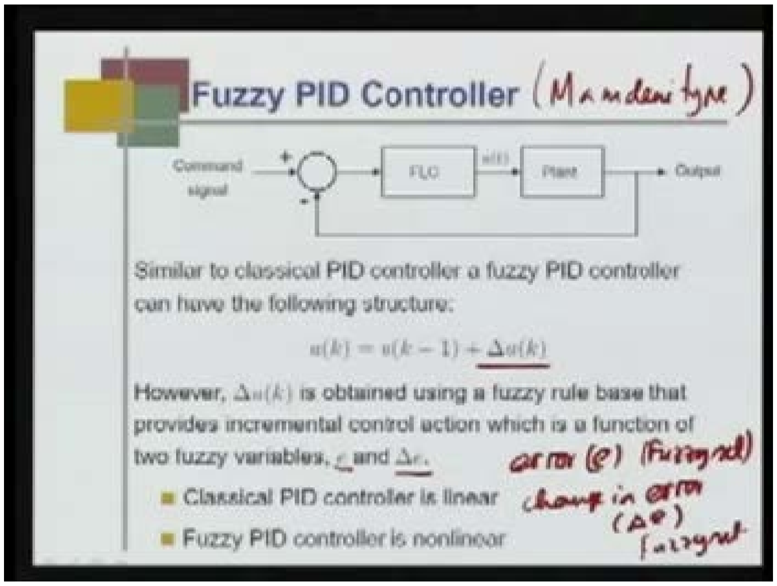
It can be shown that $u(k) = u(k-1) + \Delta u(k)$ where

$$\Delta u(k) = f(\overset{✓}{e}(k), \Delta \overset{✓}{e}(k), \cdots)$$

i.e. the increment~~ed~~al control~~ler~~ output is some function of error, change in error and higher order terms.

But first, let us go back to the previous example, that is, a classical PID controller. In a classical PID controller, this is my control action (Refer Slide Time: 15:12): u t equal to K e t plus e t dt integration and d et upon dt into some T_d. In discrete domain, I showed you that this can be written in this particular form. We can write for u k minus 1 the corresponding discrete form and subtract. If you do that, again u k can be written as present control action is equal to previous control action plus incremental control action, where this incremental control action is a function of error, change in error and higher order terms. The incremental control output is some function of error. This is delta u k, which we call as incremental control action. It is some function of error, change in error and higher order terms. Now, we will understand….

(Refer Slide Time: 16:34)


Command signal
+
-
FLC
u(t)
Plant
Output

Fuzzy PID Controller (Mamdani type)

Similar to classical PID controller a fuzzy PID controller can have the following structure:

$$u(k) = u(k-1) + \underline{\Delta u(k)}$$

However, $\Delta u(k)$ is obtained using a fuzzy rule base that provides incremental control action which is a function of two fuzzy variables, $\underline{e}$ and $\underline{\Delta e}$. error (e) (Fuzzy set) change in error (Δe) fuzzy set

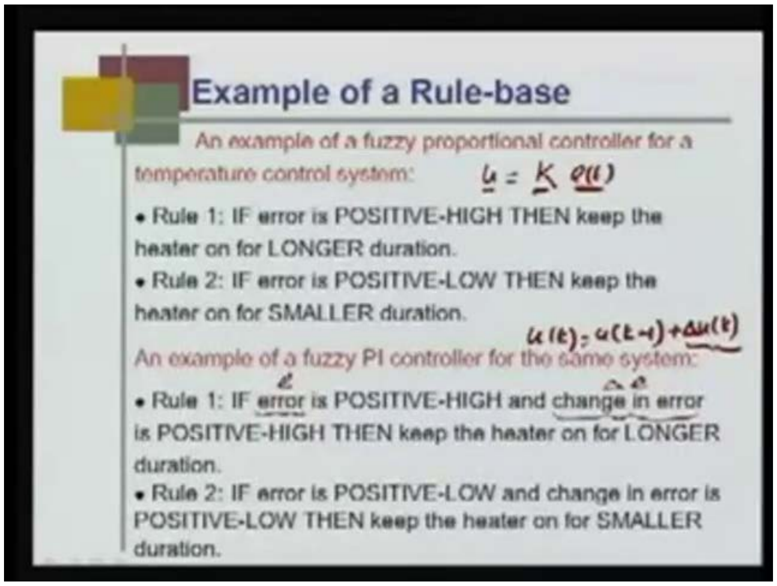
- Classical PID controller is linear
- Fuzzy PID controller is nonlinear

This is Mamdani type. This is typically Mamdani type controller, where you see that instead of a PID, we have put a fuzzy logic controller there and the rest is same. Similar to classical PID controller, a fuzzy PID controller can have the following structure. It is not that this is the only structure but there are various other structures also. I am just presenting one of the structures that is most commonly followed. In that structure, u k is u k minus 1 plus delta u k. We realized that whether it is a PI or PID, it will have this particular form u k equal to u k minus 1 plus delta u k.

This delta u k, in case of fuzzy logic controller, is obtained using a fuzzy rule base that provides incremental control action, which is a function of two fuzzy variables, error and change in error. I would say this is error e and change in error, which is delta e. This is a fuzzy set (Refer Slide Time: 17:58). We build another fuzzy set delta e and the delta u k, which is another fuzzy variable, incremental control action, is a function of two fuzzy variables e and delta e.

Why are we doing fuzzy PID controller instead of classical PID? In classical PID controller, normally, it is a design for linear system. There are also PID controllers that can be designed for nonlinear systems, where the PID gains are varying. They are trajectory dependent and as the system moves from one part of the trajectory to another part of the trajectory, the gains do vary. That is possible also using classical PID, but fuzzy PID controller is nonlinear, that is, this particular controller is always preferred for nonlinear system, where the classical PID controller design is either difficult or is not very accurate – even if we design, it is not very efficient. Now, we want to compute what is the incremental control action delta u in a fuzzy logic controller of Mamdani type and to compute this incremental control action, we have a rule base. How do we construct a rule base? Think that the fuzzy logic controller is implementing simply a proportional controller, a proportional controller for nonlinear a system. What is happening? According to the situation error….

(Refer Slide Time: 20:04)



Example of a Rule-base

An example of a fuzzy proportional controller for a temperature control system: $\underline{u} = \underline{K}\,\underline{e(t)}$

- Rule 1: IF error is POSITIVE-HIGH THEN keep the heater on for LONGER duration.
- Rule 2: IF error is POSITIVE-LOW THEN keep the heater on for SMALLER duration.

$u(t) = u(t-1) + \underline{\Delta u(t)}$

An example of a fuzzy PI controller for the same system:

- Rule 1: IF $\overset{e}{\underline{\text{error}}}$ is POSITIVE-HIGH and $\overset{\Delta e}{\underline{\text{change in error}}}$ is POSITIVE-HIGH THEN keep the heater on for LONGER duration.
- Rule 2: IF error is POSITIVE-LOW and change in error is POSITIVE-LOW THEN keep the heater on for SMALLER duration.

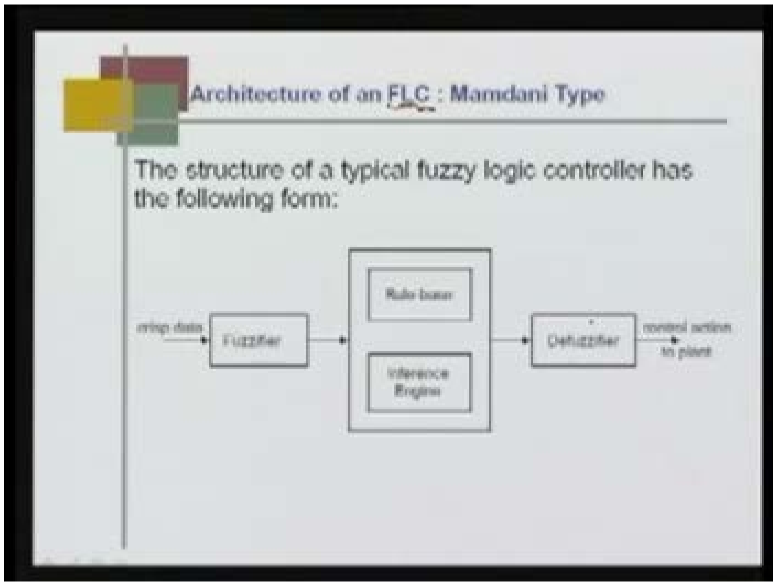
You know the proportional controller u is simply K e t. This is my error, this is my gain, and this is my direct control action actually. Naturally, when I have a proportional controller, my fuzzy rules look like… if error is positive-high, then keep the heater on for longer duration. This is the temperature control system. If error is positive-low, then keep the heater on for smaller duration. We will see how the rule works.

In this rule, you see that the antecedent part has only one fuzzy variable, that is, error, because, in proportional controller, u is directly a function of e t and the control action is not an incremental control action but is a direct control action u equal to K e t, but in case of a fuzzy PI controller, this is the incremental control action, that is, for PI controller, we realized that u K should u K minus 1. The present control action is equal to previous control action plus delta u K (Refer Slide Time: 21:29). The fuzzy logic rule base or fuzzy logic controller will compute what is delta u K. You already know what the previous input is and that has to be added.

This incremental addition is computed and you know that this incremental addition is a function of error and change in error. Hence, the rule should look like this: if error is positive-high and change in error is positive-high, then keep the heater on for longer duration. Instead of that, probably you should understand that the incremental control action here is…. The rule states that I am presenting this to you to let you know how we form the rule.

In case of fuzzy proportional controller, my fuzzy variables, independent variables is only one and that is error. In case of a fuzzy PI controller, my independent variables are error and change in error. This is e and this is delta e and based on that, the control action is computed. For example, here, we have written that if error is positive-high and change in error is positive-high, then keep the heater on for longer duration. Rule 2: if error is positive-low and change in error is positive-low, then keep the heater on for smaller duration.

(Refer Slide Time: 23:08)


Architecture of an FLC : Mamdani Type

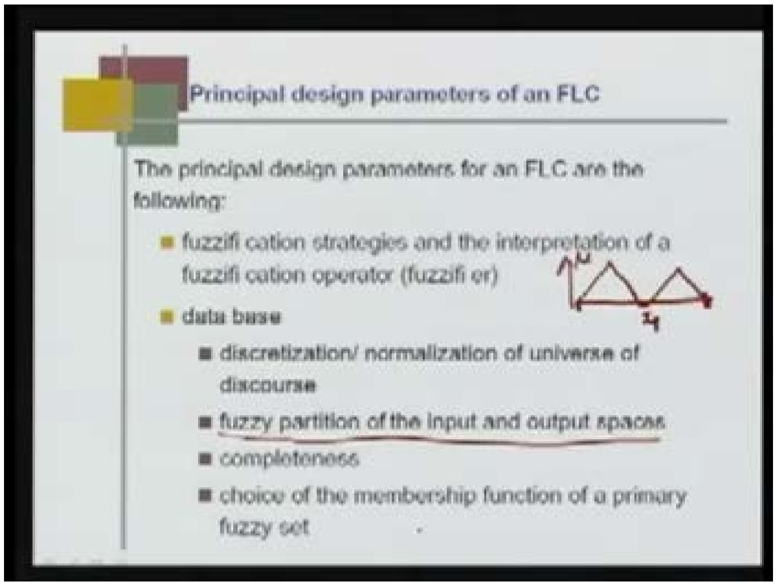
The structure of a typical fuzzy logic controller has the following form:

This is the architecture of a fuzzy logic controller. This is the short form for fuzzy logic controller (Refer Slide Time: 23:16), Mamdani type. This is the Mamdani type controller, where the actual data that the controller is receiving is crisp data or classical data that has a definite value. That crisp data goes to the fuzzy logic controller and it has these four components that you can see: fuzzifier, rule base, inference engine and defuzzifier.

Fuzzifier. In a fuzzy logic controller, the computation is through linguistic values, not through exact computation. Naturally, the fuzzifier would fuzzify the crisp data. In case of temperature, I can say it is hot, medium-hot, cold, medium-cold, very hot and normal. These are the fuzzifier. That means given a crisp data or the value of temperature say 40 degrees, then I have to now convert to various linguistic values and each linguistic value will be associated with a specific membership function. That is fuzzifier.

Once the data has been fuzzified, then it goes to the rule base and using an inference mechanism…. The inference is taking place in fuzzy term, not in classical term and after a fuzzy inference takes place about the decision or about the control action, we place a defuzzifier. What this defuzzifier does is it converts the fuzzy control action to a crisp control action.

(Refer Slide Time: 25:12)


Principal design parameters of an FLC

The principal design parameters for an FLC are the following:

- fuzzifi cation strategies and the interpretation of a fuzzifi cation operator (fuzzifi er)
- data base
  - discretization/ normalization of universe of discourse
  - fuzzy partition of the input and output spaces
  - completeness
  - choice of the membership function of a primary fuzzy set

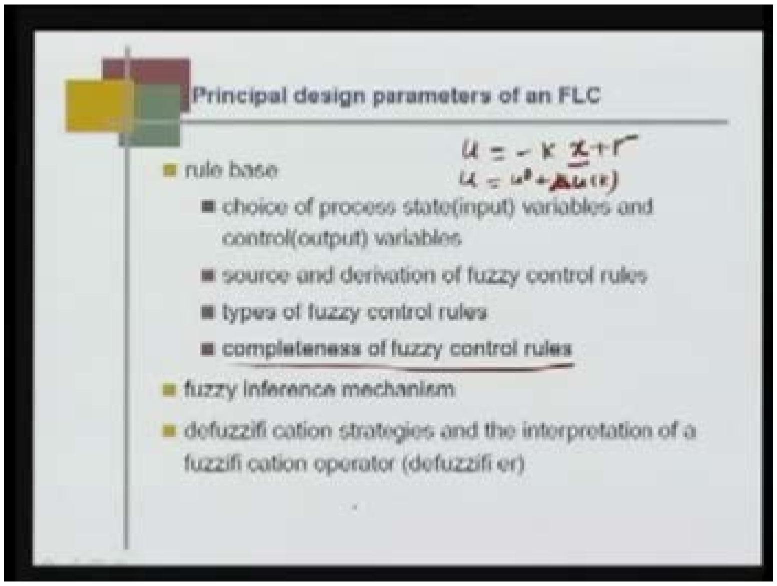
In general, what we can say is the principal design parameters of a fuzzy logic controller are the following: fuzzification strategies and interpretation of a fuzzification operator. How do we fuzzify a crisp data? In the database, the discretization or normalization of universe of discourse is done, because we must know the range of data one will encounter in an actual plant. Accordingly, the normalization must be done so that we are taking into account all possible values of data that one may encounter in a physical plant.

Fuzzy partition of the input and output spaces. If I know the dynamic range of an input to the controller and the input to the plant (input to the plant is actually output to the controller)… if I know the dynamic range, then in that dynamic range, I must learn how to do fuzzy partition of the input and output space and this fuzzification suits such the process should be complete in the sense.… You see that I am drawing a universe of discourse here. This is the real value for a specific variable x_1. If I have defined a fuzzy set like this and like this, you can easily see that this part of the data is not associated with any fuzzy membership. This is mu and this is x_1 and unfortunately, this part is not associated with any membership.

This fuzzification process is not complete. That means the entire universe of discourse in a specific domain, wherever there are control systems…. There are various kinds of control systems: process control, robot control, and aircraft control. Every control system is associated with some input data and some output data. All possible input data and all possible output data should be associated with a specific linguistic value as well as a membership function.

Choice of the membership function of a primary fuzzy set. We must be very clear what category and what kind of…. We have already discussed in the first lecture the varieties of membership function. We talked about triangular membership function, trapezoidal membership, pi function, and so on. We have to be very clear about what kind of membership function we would like to employ.

(Refer Slide Time: 28:30)



Principal design parameters of an FLC

$u = -K \underline{x} + r$

$u = u^0 + \Delta u(k)$

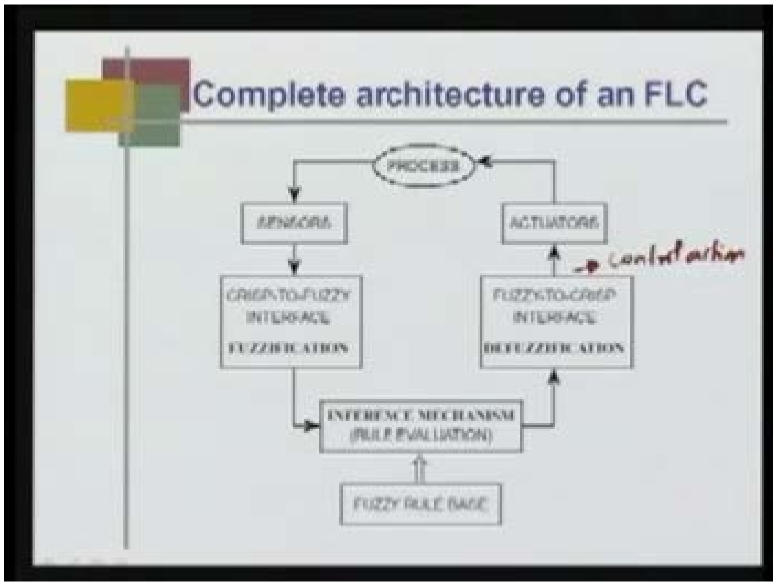
- rule base
  - choice of process state(input) variables and control(output) variables
  - source and derivation of fuzzy control rules
  - types of fuzzy control rules
  - $\underline{\text{completeness of fuzzy control rules}}$
- fuzzy inference mechanism
- defuzzifi cation strategies and the interpretation of a fuzzifi cation operator (defuzzifi er)

Now, rule base. Once fuzzification is done, how do we create a rule base? As I said, typically, in the rule base, the two variables that are most important are error and change in error and we also showed why it is so. Rule base. Choice of process state input variables and control variables. You know that if I am implementing a fuzzy state feedback controller, then, a fuzzy state feedback controller u would be minus K x. So, x is the states of the system, whereas if I am implementing a fuzzy PID controller, then it will be u old plus K delta u k. Here, this delta u k is a function of error and change in error, whereas, in a state feedback controller, this is a common signal r and so, the control action is dependent on state x_1, x_2, and x_n.

Source and derivation of fuzzy control rules. How do I derive these rules? What is the basis? Types of fuzzy control rules. A type of fuzzy control rule means whether it is a PID controller, fuzzy PID controller or it is a fuzzy state feedback controller. Similarly, completeness of fuzzy control rules means given any crisp data in the domain of input space as well as output space, do I have in my rule base a specific rule associated with this data? If I do not have any rule for this data, then the FLC will fail. That is meaning of completeness of fuzzy control rules.

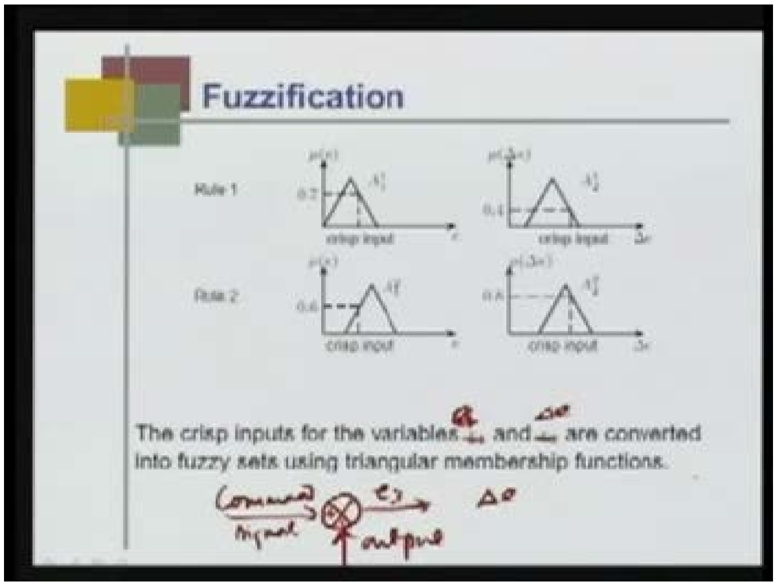
Then, fuzzy inference mechanism. We have already talked about what is fuzzy inference mechanism. Given multiple rules, how do we infer the consequence part? Defuzzification strategies and the interpretation of fuzzification operator. Once the fuzzy inference is done, from the fuzzy inference, how do I get a crisp value or a crisp control action? This is called defuzzification.

(Refer Slide Time: 31:18)



Now, I hope you are very clear about the complete architecture of a fuzzy logic controller. This is your process or a plant. You have sensors. They provide the plant state data and then, you have this fuzzification, crisp-to-fuzzy interface and then, inference mechanism. This inference mechanism takes the help of the fuzzy rule base and given a specific sensor data, it evaluates what should be the fuzzy control action. From fuzzy control action, we go to defuzzification, which we call fuzzy-to-crisp interface, and finally, the control action. This is your control action here in this domain (Refer Slide Time: 32:05). This control action is actuated through actuator to the (Refer Slide Time: 32:09) process. This is the overall structure of a fuzzy logic controller.

(Refer Slide Time: 32:42)



Fuzzification

The crisp inputs for the variables e and Δe are converted into fuzzy sets using triangular membership functions.

Now, I will be a little precise in terms of some of the concepts that Mamdani type fuzzy logic controller is involved with. In a Mamdani type fuzzy logic controller, the fuzzification processes…. Not only Mamdani but this is the normal fuzzification process. Let us say I have rule 1 and this is rule 2. Of course, all my rules…. Normally, a fuzzy PID controller has two fuzzy variables: one is error and another is change in error.

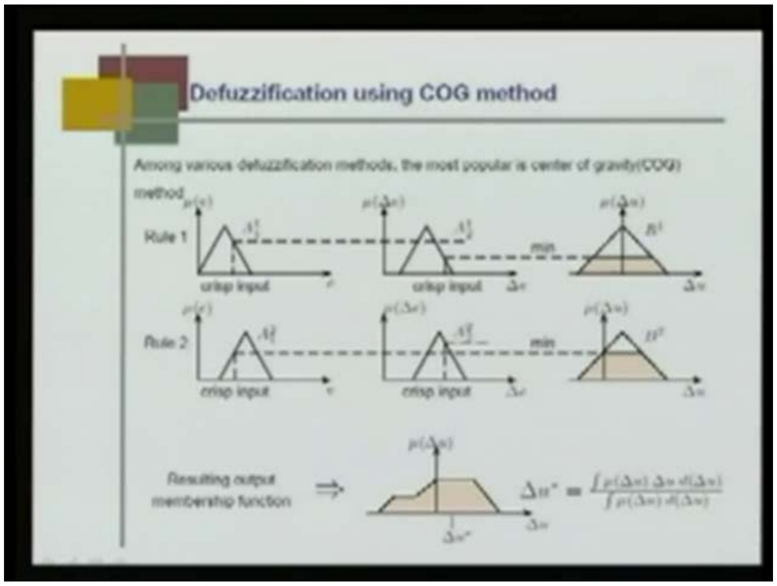
because I know what my error at the command signal is and I know what my output from the system is. I compare it with the command signal and then, I know what the error is. Given a command signal and output, I know what the error is. I also know what is the change in error, because I know what was the previous error and what is the error now. By subtracting the previous error from the current error, I know what my delta e is. I know these things already. That means the fuzzy logic controller has the information of error and change in error. If you know the change in error and what is the crisp value, then the corresponding fuzzy membership is computed according to the rule.

The rule defines that if my error is A_1 1 where A_1 1 is a specific fuzzy set (in this case, it is triangular shaped, triangular membership function), you can easily see… now, this particular crisp input has a fuzzy membership 0.7 in the fuzzy set A_1 1. Similarly, the second fuzzy variable, that is, change in error whose crisp input is this value, this specific value (Refer Slide Time: 34:36), has a fuzzy membership function 0.4 in fuzzy membership function A_2 1.

The error is A_1 1 and change in error is A_2 1. Now, given the crisp data, which is error and change in error, we have a corresponding membership function for rule 1. Rule 1 shows that the error has membership function 0.7 and the change in error has 0.4. In the second case, in rule 2, the same crisp input gives you 0.6 membership function and the other crisp input gives you 0.8 membership function.

This is how we fuzzify a crisp data to fuzzy data or we make them fuzzy, that is, the crisp input for variable x_1 and x_2…. Actually, this is not x_1 and x_2 but e and delta e are converted to fuzzy sets using triangular membership functions. It is not always triangular, it can be anything, but normally in control literature, most of these membership functions are triangular functions.

(Refer Slide Time: 36:21)



Defuzzification using COG method

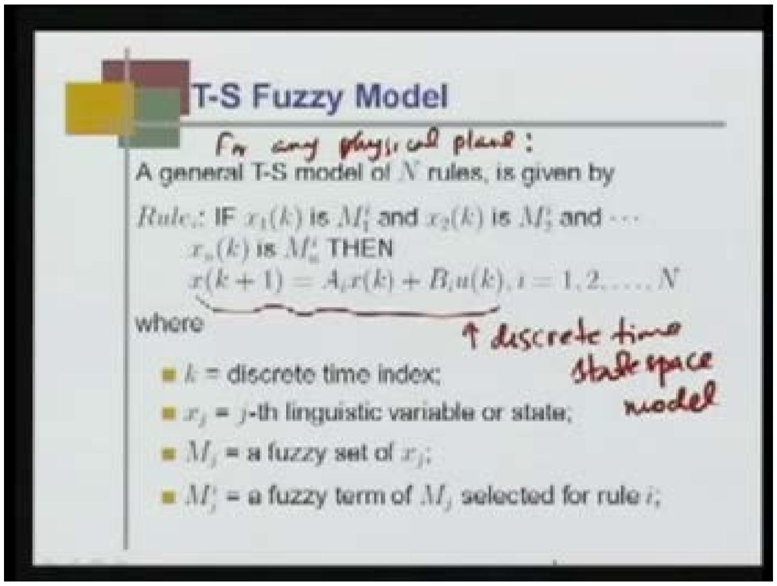
Among various defuzzification methods, the most popular is center of gravity(COG) method

$$\Delta u^* = \frac{\int \mu(\Delta u)\, \Delta u\, d(\Delta u)}{\int \mu(\Delta u)\, d(\Delta u)}$$

Defuzzification. Once I know how to do the fuzzification, defuzzification is explained in the following diagram. You see that among various defuzzification methods, the most popular is center of gravity method. How do I do it? Crisp input is given at any situation, any k th sampling instant and the fuzzy logic controller gets the command signal, gets the actual output of the plant, computes the error, computes the change in error and then, those crisp values are fed into the fuzzification layer. Then, you have the membership function. You pass on those fuzzy data to the rule base and then, for a specific rule base… You see that in rule 1, you see that if you compare the membership mu A_1 1 and mu A_2 1, mu A_2 1 is the minimum and correspondingly, you shade the zone of action. This is delta u. How much should be the incremental control action? This is my shaded portion or shaded portion of my control action.

Now I take a second one, second rule and there again, I evaluate the fuzzy membership function A_1 2 and A_2 2. You see that the membership function in A_1 2 is less. Corresponding to that, we shade the incremental control action. Now, you see that if I take the maximum of these two shaded zones, I get this (Refer Slide Time: 38:03), maximum of this. After I get, this is the fuzzy decision, this is the fuzzy incremental control action, but how do I convert this fuzzy incremental control action to a crisp action? That is by the center of gravity method. In the center of gravity method, I integrate mu delta u d delta u upon integration of mu d delta u. If I integrate this function, I get somewhere here to be the center of gravity. delta mu star is this value, which is graphically shown here. We discussed about Mamdani type fuzzy logic controller.

(Refer Slide Time: 39:04)



T-S Fuzzy Model

For any physical plant :

A general T-S model of N rules, is given by

$Rule_i$: IF $x_1(k)$ is M_1^i and $x_2(k)$ is M_2^i and $\cdots$
$x_n(k)$ is M_n^i THEN

$$\underbrace{x(k+1) = A_i x(k) + B_i u(k)}, \; i = 1, 2, \ldots, N$$

↑ discrete time state space model

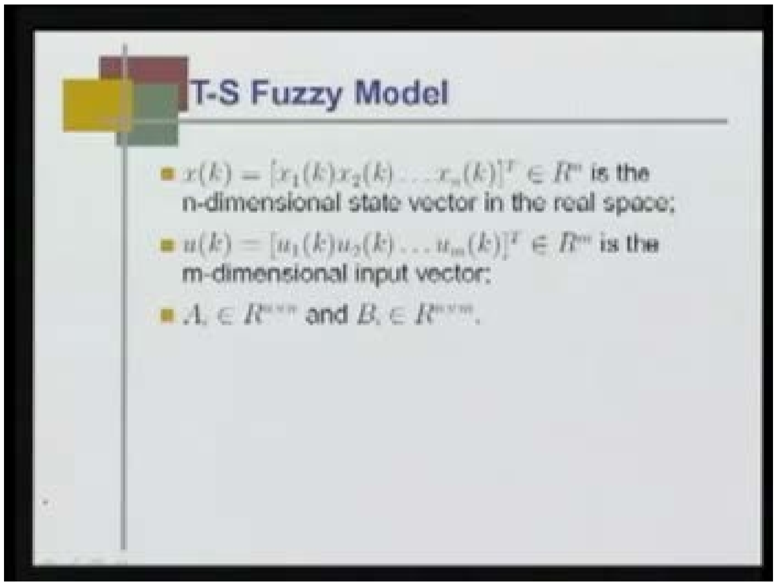
where

- k = discrete time index;
- x_j = j-th linguistic variable or state;
- M_j = a fuzzy set of x_j;
- M_j^i = a fuzzy term of M_j selected for rule i;

Now, in the second part of this lecture, we will be discussing about the Takagi–Sugeno fuzzy model. We will not discuss about how we design the control system but we will just present how an actual physical system is modeled using Takagi–Sugeno (T-S) fuzzy model. A general Takagi–Sugeno model of N rules for any physical plant, a general T–S model of N rules is given by Rule_i. This is the i th rule. If x_1 k is a specific fuzzy set M_1 i and x_2 k is another specific fuzzy set M_2 i and so on until x_n k is another fuzzy set M_n i, then the system dynamics locally is described as x k plus 1 is A_i x k plus B_i u k, where i equal to 1, 2 until N, because there are N rules.

The slide we represented here is the discrete time. I hope that all of you know that this is a discrete time state-space model (Refer Slide Time: 40:59). This k refers to a discrete time index or sampling instant, x_j is the j th linguistic variable or state, M_j is a fuzzy set of x_j, and M_j i is a fuzzy term of M_j selected for rule i. If two state variables in a plant are temperature and pressure, then M_1 would stand for temperature and M_2 will stand for pressure. Similarly, if some physical system has two state variables like position and velocity, then we can say M_1 is the fuzzy set associated with position and M_2 is the fuzzy set associated with velocity. Then, M_1 i and M_2 i are the various fuzzy terms within each fuzzy set. If I am looking at the fuzzy set for position, then, I can describe position to be positive big, positive small, positive medium, negative small, or negative medium. So, this is a specific term. These are M_j i. Similarly, if it is temperature, if my state variable is a temperature, then M_j would be a temperature fuzzy set and M_j i would be a term like hot, cold, and medium. This is how this T-S fuzzy model is described.

(Refer Slide Time: 43:06)

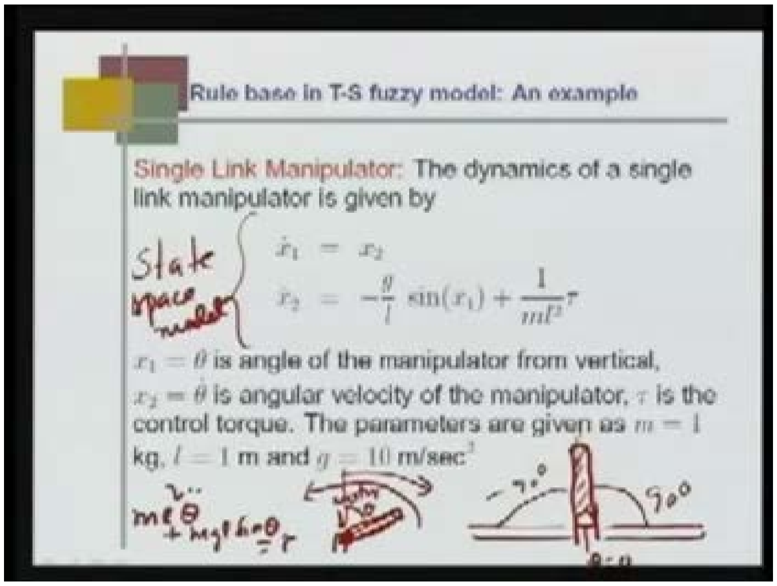


T-S Fuzzy Model

- $x(k) = [x_1(k) x_2(k) \ldots x_n(k)]^T \in R^n$ is the n-dimensional state vector in the real space;
- $u(k) = [u_1(k) u_2(k) \ldots u_m(k)]^T \in R^m$ is the m-dimensional input vector;
- $A_i \in R^{n \times n}$ and $B_i \in R^{n \times m}$.

We are assuming that my physical system is described by n state space – n states, which are x_1 k, x_2 k until x_n k and the inputs are m-dimensional input vector, that is, it is a multi-input system. I have some a priori understanding about the system.

(Refer Slide Time: 43:35)



Rule base in T-S fuzzy model: An example

Single Link Manipulator: The dynamics of a single link manipulator is given by

State space model
$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -\frac{g}{l}\sin(x_1) + \frac{1}{ml^2}\tau \end{cases}$$

$x_1 = \theta$ is angle of the manipulator from vertical, $x_2 = \dot{\theta}$ is angular velocity of the manipulator, τ is the control torque. The parameters are given as $m = 1$ kg, $l = 1$ m and $g = 10$ m/sec^2

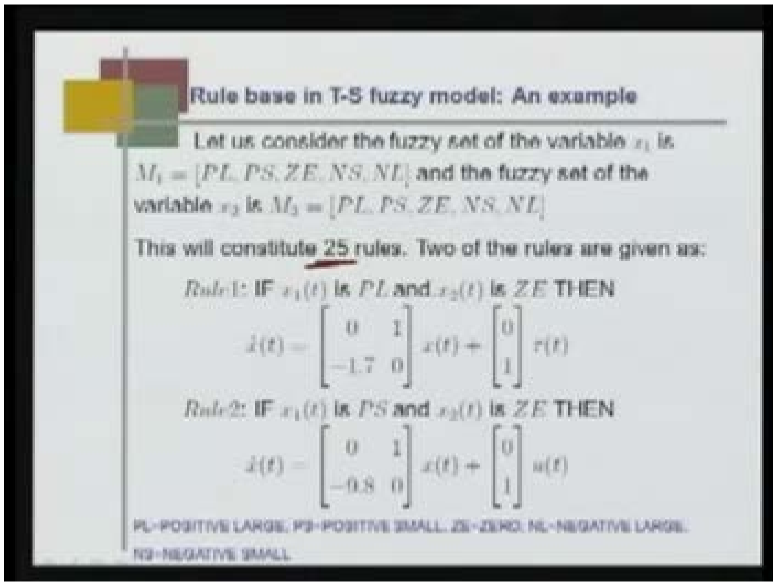
$m l^2 \ddot{\theta} + mgl\sin\theta = \tau$

Now we will go to a simple system. You know a single link manipulator. What is a single link manipulator? You know that there is a hinge and to the hinge is attached one manipulator or one link and there is a motor here (Refer Slide Time: 43:45). This is your motor and this manipulator can go in this direction, it can rotate vertically in the vertical plane. It can rotate either anticlockwise or clockwise.

Normally, if I assume this is my hinge, then this is my link. If I assume this to be my neutral position, that is, theta = 0. When this link comes to this position.… I can say this is positive 90 degrees (Refer Slide Time: 44:43) and this position is negative 90 degrees, the moment it takes. (Refer Slide Time: 44:52) you define. For such a single link manipulator, you can easily derive the dynamics. If this is my theta with respect to my vertical axis.… This is my vertical axis (Refer Slide Time: 45:13) and this is my theta. So, the torque will be m l square theta double dot plus m g l sine theta. Given the amount of torque that the motor would apply at this hinge, the position and velocity of the link would vary according to this dynamics.

That is m l square theta double dot, this is the acceleration plus m g l sine theta is tau. You can easily derive this. Now, if I take x_1 = theta, the angle of the manipulator from the (Refer Slide Time: 46:06), and x_2 equal to theta dot, which is the angular velocity of the manipulator, and tau is the control torque, then I can write this equation in the state-space format. This is the state-space model (Refer Slide Time: 46:31), where you can easily see x_1 dot is theta dot, which is x_2, and x_2 dot, which is theta double dot is equal to, you can see, minus m g l sine x_1 divided by ml square which is minus g by l sine x_1 plus 1 upon m l square into tau, the control action. You have a system single link manipulator that has two states. We would like to represent this dynamics as a fuzzy system. How do we do it?

(Refer Slide Time: 47:20)



Rule base in T-S fuzzy model: An example

Let us consider the fuzzy set of the variable x_1 is $M_1 = [PL, PS, ZE, NS, NL]$ and the fuzzy set of the variable x_2 is $M_2 = [PL, PS, ZE, NS, NL]$

This will constitute 25 rules. Two of the rules are given as:

Rule1: IF $x_1(t)$ is PL and $x_2(t)$ is ZE THEN

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ -1.7 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tau(t)$$

Rule2: IF $x_1(t)$ is PS and $x_2(t)$ is ZE THEN

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ -0.8 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$

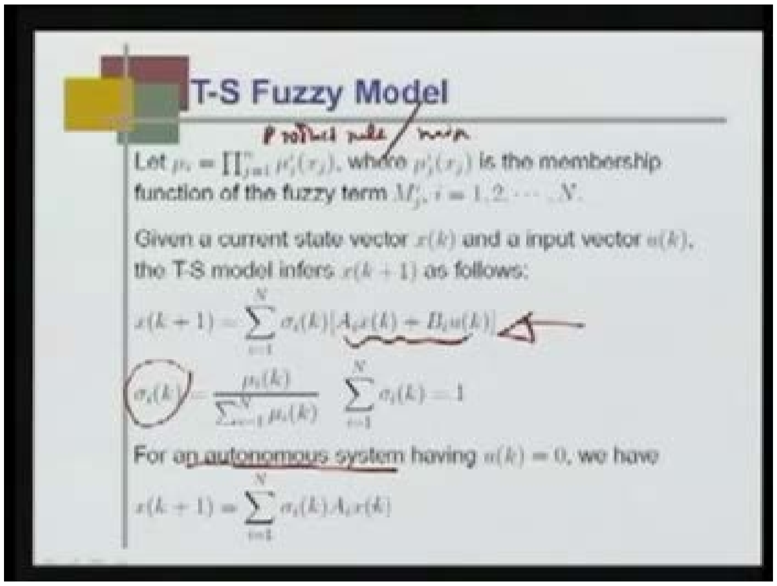
PL-POSITIVE LARGE, PS-POSITIVE SMALL, ZE-ZERO, NL-NEGATIVE LARGE, NS-NEGATIVE SMALL

Let us consider the fuzzy set variable x_1 is M_1, which is positive large, positive small, zero, negative small, and negative large, which is written here. Similarly, the fuzzy set associated with the variable x_2…. x_1 is angular position and x_2 is angular velocity and for that, the fuzzy set is M_2 and the fuzzy terms are positive large, positive small, zero, negative small, and negative large. At the moment, we are not bothered….

You can consider these fuzzy terms to be either triangular or pi function or s function or Gaussian function or trapezoidal function. We will not be interested so much on that aspect now, but what is important to learn is that if I have a rule, a system dynamics defined in two fuzzy variables, each fuzzy variable is again partitioned into five fuzzy zones. Then, the total number of rules that are necessary to describe such a dynamical system could be 25. What I have to say now is that if x_1 is positive large and x_2 is positive large, then the system dynamics is this. For example, here, rule 1: if x_1 t is positive large and x_2 t is 0, then x dot t is A matrix into x t plus B matrix into input tau. Similarly, you can derive another rule: if x_1 t is positive small and x_2 t is 0, then my system dynamics can be written in terms of a linear state-space equation, which is x dot t equal to A x t plus B u t.

Just like we have written in terms of continuous linear model, we can also write in terms of a discrete linear model. These expressions also can be written in terms of known simple nonlinear models. It is up to the designer (the control system designer) what kind of rules he needs in the domain of T-S fuzzy model. What we are talking about is given a dynamical system, how do we derive the T-S fuzzy model of a dynamical system so that we can design a specific controller for the system?

(Refer Slide Time: 50:11)



T-S Fuzzy Model

product rule / norm

Let $\mu_i = \prod_{j=1}^{n} \mu_j^i(x_j)$, where $\mu_j^i(x_j)$ is the membership function of the fuzzy term M_j^i, $i = 1, 2, \cdots, N$.

Given a current state vector $x(k)$ and a input vector $u(k)$, the T-S model infers $x(k+1)$ as follows:

$$x(k+1) = \sum_{i=1}^{N} \sigma_i(k)[\underbrace{A_i x(k) + B_i u(k)}] \leftarrow$$

$$\sigma_i(k) = \frac{\mu_i(k)}{\sum_{i=1}^{N} \mu_i(k)} \quad \sum_{i=1}^{N} \sigma_i(k) = 1$$

For an autonomous system having $u(k) = 0$, we have

$$x(k+1) = \sum_{i=1}^{N} \sigma_i(k) A_i x(k)$$

The T-S fuzzy model. Let mu_i be the product of all the fuzzy memberships associated with the fuzzy term in a rule base. Previously, we have two fuzzy terms: positive large and 0 (Refer Slide Time: 50:32). Given a crisp data, that is, x_1 and x_2, you have a specific membership function from positive large and another specific membership from 0. You multiply them. Sometimes, you multiply and sometimes, you also take the minimum. This is the product rule that we have taken. You can also take mean.

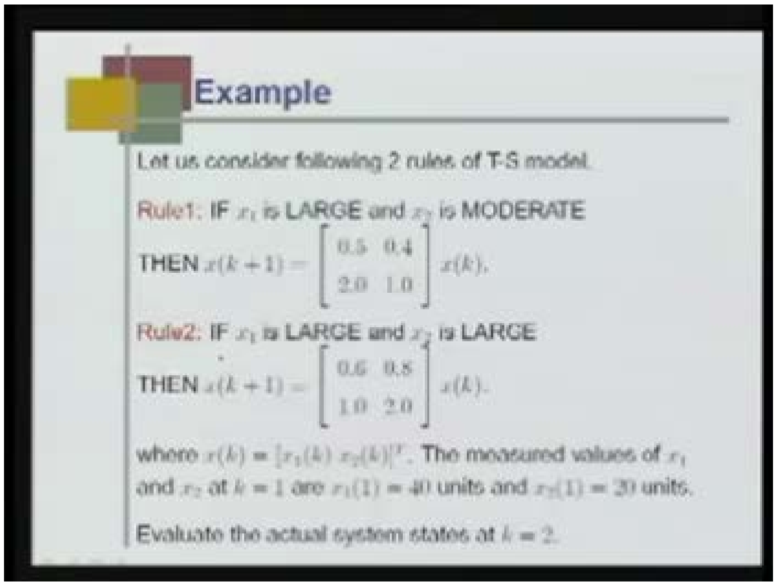
For this individual membership function, I can take a mean and that membership function is associated with this dynamics (Refer Slide Time: 51:25). Given a crisp data, we compute the membership function for each fuzzy term and the product of these fuzzy terms or the mean of this fuzzy terms is associated with the dynamical equation in that fuzzy zone. Then, the overall system dynamics is expressed as x k plus 1 is sigma, is summation….

If I have N rules, then I have N such linear systems and each linear system is associated with a fuzzy membership function, which is mu_i k and that is normalized further, because each linear system is associated with mu_i k and we normalize. The normalized membership function is sigma_i k. This is the (Refer Slide Time: 52:40) such that the summation of all sigma_i k is equal to 1.

Try to understand what we are trying to do now. Given a set of rules, for each rule…. What does a set of rule mean? A dynamic system is fuzzy partitioned into 25 different fuzzy zones. That is the meaning and in each fuzzy zone, we have a linear model. Now, we are trying to define what is the global model. The global model is described by this expression (Refer Slide Time: 53:26), where this is the linear model for the i th fuzzy rule.

I have N rules and for each fuzzy rule, the associated fuzzy membership is mu_i, which is computed by either the product rule or mean rule. That is how we compute the global fuzzy dynamics of a system. If the system is autonomous, then u k is 0. We can conveniently write x k plus 1 is sigma summation sigma_i k into A_i x k, where again sigma_i is the normalized membership function associated with the i th linear model in the i th fuzzy zone.

(Refer Slide Time: 54:33)



Example

Let us consider following 2 rules of T-S model.

Rule1: IF x_1 is LARGE and x_2 is MODERATE

THEN $x(k+1) = \begin{bmatrix} 0.5 & 0.4 \\ 2.0 & 1.0 \end{bmatrix} x(k)$.

Rule2: IF x_1 is LARGE and x_2 is LARGE

THEN $x(k+1) = \begin{bmatrix} 0.6 & 0.8 \\ 1.0 & 2.0 \end{bmatrix} x(k)$.

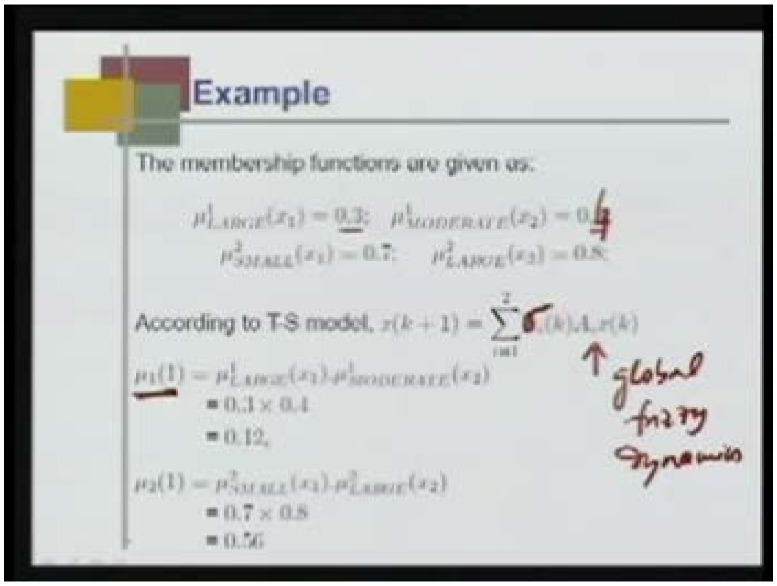
where $x(k) = [x_1(k)\ x_2(k)]^T$. The measured values of x_1 and x_2 at $k = 1$ are $x_1(1) = 40$ units and $x_2(1) = 20$ units.

Evaluate the actual system states at $k = 2$.

Now, we will explain this concept through an example. Let us consider the following two rules of T-S fuzzy model. If x_1 is large and x_2 is moderate, then x k plus 1 is (0.5, 0.4, 2, 1) into x k. If x_1 is large, x_2 is large, then x k plus 1 is (0.6, 0.8, 1, 2) into x k, where x k is…. Of course, this is a two-state dynamical system. The measured values of x_1 and x_2 at k = 1 are 40 units and 20 units, respectively, that is, x_1 1 is 40 and x_2 1 is 20 units.

The question that is asked is evaluate the actual system state at k = 2, looking at this fuzzy model. Given a crisp data, can I… if the crisp data is given at the k th instant…. The state of the actual system is given at the k th instant. Now using the T-S fuzzy model, compute the crisp state of the system.

(Refer Slide Time: 55:56)



Example

The membership functions are given as:

$$\mu^1_{LARGE}(x_1) = \underline{0.3}; \quad \mu^1_{MODERATE}(x_2) = 0.\cancel{4}$$

$$\mu^2_{SMALL}(x_1) = 0.7; \quad \mu^2_{LARGE}(x_2) = 0.8;$$

According to T-S model, $x(k+1) = \sum_{i=1}^{2} \sigma_i(k) A_i x(k)$

global fuzzy dynamics

$\underline{\mu_1(1)} = \mu^1_{LARGE}(x_1)\cdot\mu^1_{MODERATE}(x_2)$

$= 0.3 \times 0.4$

$= 0.12,$

$\mu_2(1) = \mu^2_{SMALL}(x_1)\cdot\mu^2_{LARGE}(x_2)$

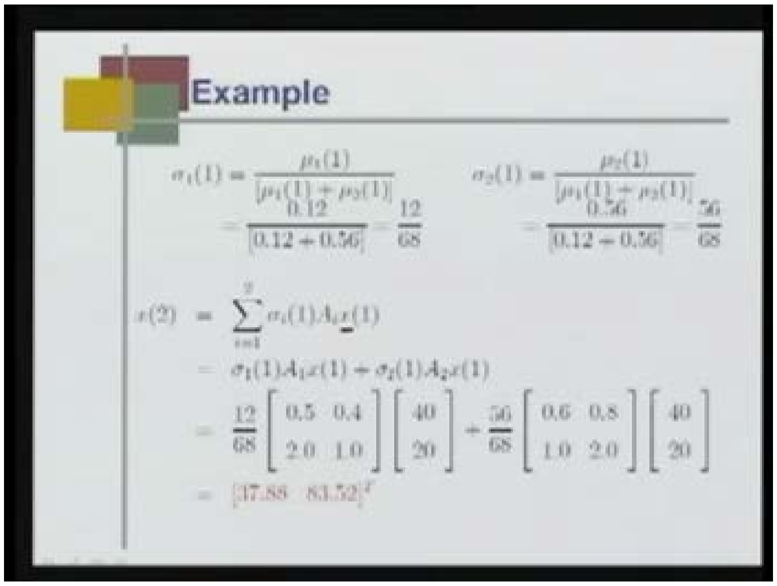
$= 0.7 \times 0.8$

$= 0.56$

In this example, we assume that x_1 is large and x_2 is moderate and here, x_1 is large and x_2 is large. For these four fuzzy terms, their associated membership functions are mu_{LARGE} 1 is 0.3, $\text{mu}_{MODERATE}$ 1 is 0.3, mu_{SMALL} 2 is 0.7 and mu_{LARGE} 2 is 0.8. Given that, according to the T-S fuzzy model, this is my global fuzzy dynamics. This is not h_i, this is sigma_i k (Refer Slide Time: 56:44) A_i x k, where sigma_i k is the associated fuzzy membership.

What do we do? We have two rules. For each rule, I have to compute what is the associated membership value. The associated membership value for the rule 1 would be product of this membership function and this membership function 0.3. So $\text{mu}_{MODERATE}$ 1 is actually not 0.3, this is 0.4 (Refer Slide Time: 57:20); 0.3 into 0.4 is 0.12. Similarly, mu_2 1 is the fuzzy membership associated with the second rule or second linear system, that is, mu_{SMALL} 2, which is 0.7, into mu_{LARGE} 2 is 0.8 is 0.56. The normalized value….

(Refer Slide Time: 57:51)



Example

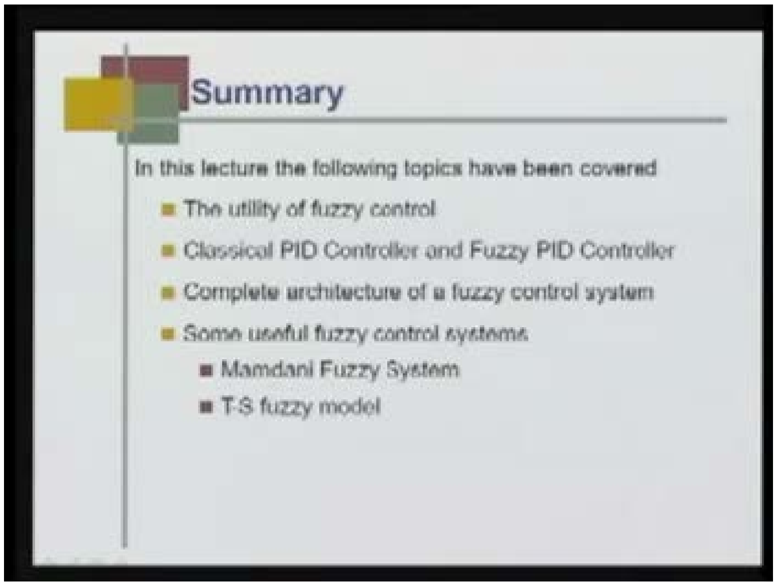
$$\sigma_1(1) = \frac{\mu_1(1)}{[\mu_1(1)+\mu_2(1)]} = \frac{0.12}{[0.12+0.56]} = \frac{12}{68} \qquad \sigma_2(1) = \frac{\mu_2(1)}{[\mu_1(1)+\mu_2(1)]} = \frac{0.56}{[0.12+0.56]} = \frac{56}{68}$$

$$\begin{aligned} x(2) &= \sum_{i=1}^{2} \sigma_i(1) A_i \underline{x}(1) \\ &= \sigma_1(1)A_1x(1) + \sigma_2(1)A_2x(1) \\ &= \frac{12}{68}\begin{bmatrix} 0.5 & 0.4 \\ 2.0 & 1.0 \end{bmatrix}\begin{bmatrix} 40 \\ 20 \end{bmatrix} + \frac{56}{68}\begin{bmatrix} 0.6 & 0.8 \\ 1.0 & 2.0 \end{bmatrix}\begin{bmatrix} 40 \\ 20 \end{bmatrix} \\ &= [37.88 \quad 83.52]^T \end{aligned}$$

mu$_1$ is normalized, then we get sigma$_1$ 1; mu$_2$ 1 is normalized, we get sigma$_2$ 1; sigma$_1$ 1 is 12 upon 68 and sigma$_2$ 1 is 56 upon 68. Finally, we find out the actual crisp value x$_2$ using this dynamics, which is i equal to 1 to 2. We have two rules. sigma$_i$ 1 is the associated normalized fuzzy membership, A$_i$ is the associated system matrix and x is the state vector, which is x$_1$ and x$_2$ and the 1 stands for the sampling instant.

If we compute that, you get 12 upon 68, my A$_1$ matrix is given as this (Refer Slide Time: 58:46), my x$_1$ is (40, 20). Similarly, the second rule 56 upon 68 is my membership function associated, A$_2$ is given as this (Refer Slide Time: 58:57) and at the k th sampling instant, my x$_1$ or the states are 40 and 20. If you compute, this is my answer: 37.88 and 83.52. You got an idea how by using T-S fuzzy model, we can describe the dynamics of an actual plant. We will discuss about the T-S fuzzy model and Mamdani type of fuzzy logic controller in this course in detail and in depth later.

(Refer Slide Time: 59:41)



Summary

In this lecture the following topics have been covered

- The utility of fuzzy control
- Classical PID Controller and Fuzzy PID Controller
- Complete architecture of a fuzzy control system
- Some useful fuzzy control systems
  - Mamdani Fuzzy System
  - T-S fuzzy model

Today, we end this class with the summary. What we learnt today is the utility of fuzzy control, why we employ fuzzy control, classical PID controller and fuzzy PID controller, whether they have a connection or they do not have a connection. We showed the idea that normally in Mamdani type of fuzzy controller, the incremental control action is dependent on two fuzzy variables, that is, error and change in error. It basically comes from the structure of PI controller or PID controller. Then, we learnt complete architecture of fuzzy control system and what are the various components of fuzzy control system. Some useful fuzzy control systems are popularly known as Mamdani type fuzzy logic system and Takagi–Sugeno fuzzy model. Thank you.