INTELLIGENT CONTROL OF ROBOTIC SYSTEMS

Prof. M. Felix Orlando

Department of Electrical Engineering

Indian Institute of Technology Roorkee

Lecture 09: Fuzzy Logic Control-II

Good afternoon, everyone. Today, we are going to see fuzzy logic control part 2. In that, we are going to see about fuzzy PI PD controller coming under Mamdani controllers, and then we are going to see about TS fuzzy controller. Now, the outline is, as I said, first we are going to see fuzzy PI and PD controllers that fall under fuzzy Mamdani controller. Then, we are going to see Takashi Sugeno fuzzy control, that is, TS fuzzy control.

First, let us start with the fuzzy PI controller. You know that the classical PI controller is given by an expression in continuous form $u(t) = K_P e(t) + K_I \int_0^t e(\tau)\, d\tau$ where K_p is the proportional gain, K_i is the integral gain, e(t) is the error, and u(t) is the control signal.

The discrete version of the above PI controller is can be written as

$$u(k) = K_P e(k-1) + K_I\big((e(k-1) + e(k-2) + \cdots)\Delta T\big)$$

Thus, we can say that from the above two equations, we can say

$$u(k) - u(k-1) = K_p\big(e(k) - e(k-1)\big) + K_I e(k)\Delta T$$

And we can say that $u(k) = u(k-1) + \Delta u(k)$,

where $\Delta u(k)$ is a function of error and change in error. And that is given by

$u(k) - u(k-1)$ and it is a function of, as I said, e changing error and the higher-order forms. Now, coming to the fuzzy PI controller, the fuzzy controller will have the control structure given by $u(k) = u(k-1) + \Delta u(k)$ and using the fuzzy rule base, delta u is obtained, which is nothing but an incremental control action which is a function of error and change in error e and delta e, respectively. Now, looking at the general characteristics of a PI controller response, the rule base is formed in fuzzy PI controller.

Now, let us say that this is the response of the PI controller. Then we can say with some nomenclature: PL stands for positive large, PM stands for positive medium, and PS

stands for positive small, ZE stands for zero, NS stands for negative small, and this is basically having the response to the steady-state desired output, which is here 1. So, the error is given by yd minus y, where yd is the desired steady-state output and y is the actual output. From this response, we can derive the rules as such with e and e dot, which is nothing but precisely e. Let us take any one: if the error is 0 and the change in error is also 0,

then the control action, which is an incremental control action delta u, is 0. Let us say that if the error is positive small and the change in error is 0, then your change in control action, the incremental control action, is positive small. Likewise, if we say the error is positive medium and the change in error is positive medium again, then you have to have your delta u coming out to be positive large. Let us say that if the error is positive small and your change in error is negative small, then what will be your delta change?

It is 0 because your change in error is negative, negative small. Okay, so it is going to get reduced slowly. We are here because of the error positive small, and it has to reach here and it has to reach here to the desired value with your change in error also reducing. That means your actual value, as time goes on, is also decreasing towards the desired value. So, in that case, your delta u you do not have to give any delta u. That is why, in that case, it is 0. Okay, so in this way, for a PI fuzzy PI controller, from the response of the controller, we can have our rules formed or rules developed so that they form the rule base. This is called intuition-based rule base. Based on intuitions, we can have our rules developed: rule 1, rule 2, rule 3, rule 4, rule 5, rule 6, rule 7, rule 8, rule 9, rule 10 accordingly. Now, let us move on to a fuzzy PD controller. A classical PD controller is given by the control law $u(t) = K_P e(k) + K_D \frac{(e(k)-e(k-1))}{\Delta T}$. Thus, the control action u is a function of e of k and the change in error.

So, if you want to have the rules, We can have the rules based on three linguistic variables: e, e dot, and u, where u is the output linguistic variable, and e and e dot are input linguistic variables. Now, the fuzzy PD controller can also be expressed as a function of e(k) and delta e(k) to form the rule base, as I told. Now, the rule base, let us take An example, that is, say, two-link Cartesian position control, two-link robot Cartesian control, like you need to control the position of the end effector.

I need to go here. So, this is your x desired, this is your x actual. So, that is my goal. So, if we are going to derive the rules for this control objective to have the tooling planar robot reach a desired position in the Cartesian space, we are having here e, change in

error, and change in error as two input linguistic variables for the fuzzy controller and You being the joint torque controller, being the joint torque, what will happen in this case?

So, rule 1 can be: if e is positive high and delta e is positive high, then the joint torque is positive high because if the error is too large, then your Current value or actual x actual is very less to reach the desired value, and also your change in error is positive high, which implies that at every instant, we have our actual value not moving towards the desired value of the robotic system. The robot has to reach x_d, so Change in error keeps on increasing, which is positive high, which implies that the current robot end effector is not moving towards the desired value x_d. So, joint torque must be positive high to bring it to the desired value x_d. That is what is given in rule 1.

Likewise, let us move on to rule 2. If the error is positive low, that means you are close to the desired point in the Cartesian space, and your change in error delta e is positive low, then the joint torque is also positive low because of the closeness to the desired position in the Cartesian space. Your change in error is also positive low, which means at every instant the actual position of the robot is moving towards the desired position. And hence, the joint torque, which is the control signal given to the robotic system, that is tau 1 and tau 2 given to the robotic system, is also low.

Now, let us see an example with a single-link manipulator of a PD controller. Implementation of a PD controller towards a single-link manipulator control. Let the dynamic model of a single-link manipulator be given by $\ddot{\theta} + 10 \sin\theta = \tau$

where theta is the joint angle from the vertical position and tau is the joint torque. Now, considering x_1 as theta and x_2 as theta dot as the state variables, the control input u as tau is we can have the state space model of this single-link manipulator as $\dot{x}_1 = x_2$

$\dot{x}_2 = -10 \sin x_1 + u$

So, upon doing the state space control of it using a simple PD controller of the state space model of the single-link manipulator. We obtain this joint angular. Result. Where we can see. The input joint arc. Is obtained.

With saturating. Like we are having a maximum. Cut shot of that. Like we are forcefully. Suppressing that.

That if it is more than 25. And less than 20, we are making it to be within that limit, hard limiting the control input to this system. With this hard limitation of the control signal to this system, we get the response as shown in the slide here, which is, we have the

settlement of the system that is coming around 2.5 seconds, or precisely you can say 3 seconds, and then we have The time history of the error and derivative of the error where you can say that there is a steady state error for the joint angular value and hence we can see that the error is not becoming zero in this case for a desired value it is not converging to that Then we need to tune the KP and KD control gain parameters to obtain the steady state error 0.

Now, there exists some steady-state error in this situation. Now, let us design a PD controller for this same problem. So, we need to form the rules. Let us say that The fuzzy sets associated with the input variable or fuzzy linguistic variable B, negative, positive, and 0 for the fuzzy linguistic variable E. Likewise, the other fuzzy input linguistic variable has the fuzzy subsets.

These are the subsets, not the sets. These are subsets. So, that will give you the output based on your value of fuzzy linguistic variable 1, that is E, and fuzzy linguistic variable E dot. Based on that, we are going to have E being negative, positive, or 0. Now, there are two linguistic input fuzzy variables.

With the fuzzy subsets being 3, so the number of rules that is going to be here is r equal to q power n, the two state variables with q being 1, 2, 3, so 3 power 2 equal to 9 rules going to be formed with this combination of negative, positive, and 0 for E and E dot. So, we form those 9 rules so that they are going to be the rule base. Rule 1 says when E is negative and E dot is positive, accordingly, U is 0. Likewise, when E is positive, E dot is positive, then we have your U, that is the control signal, to be negative. Likewise, we form 3.

Power 2, that is 9 total rules for this control problem. Once we form these rules, we will find the membership function values for the corresponding rules. For this membership function value and for this input membership function value, we will perform the AND operation to give the the consequent membership function value. Likewise, we do all. Once we complete all, we get the output distribution by maximizing the consequence. When we maximize all the consequences of all the rules, we get the output distribution.

From the output distribution, we can get the crisp value upon defuzzification, which we can obtain using either the center of gravity approach or the weighted average approach. That crisp value is going to be your u, which is the control signal given to the single-link manipulator. So, the simulation we have done in this problem will be as such: for each instant, you have the output crisp value obtained from the output distribution. Since this

is a regulation problem, your control signal u converges to 0, as you can see once the complete iteration gets over. Once your complete iteration gets over, your crisp output value is zero because this is a regulation problem. These are the results we obtained from this simulation study of a single-link manipulator using a fuzzy PD controller. Finally, we have our

joint angular trajectory, which means the convergence of the joint angle of the single-link manipulator. It started with some value around, say, 23 degrees, and it converged to zero degrees. Hence, it is a regulation problem: you start from some non-zero value and make it converge to zero. And hence, to achieve this, your control signal in this problem converges to 0 as well. Now, let us talk about the Lyapunov-based approach. Here, it implies very explicitly that the Lyapunov-based approach means we are going to use a Lyapunov function candidate, which is a positive definite scalar function candidate. It will be utilized to develop the rules because, so far, we have seen the rules developed based on intuition. Here, we have the Lyapunov stability analysis to form the rules. Let us have a simple example given by the linear model $\dot{x} = Ax + Bu$ given by

$$\dot{x} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} [x] + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} u$$

Here, the inputs are the state variables x1, x2, x3. The fuzzy sets for these state variables are, that is, the fuzzy subsets for these inputs are positive, 0, and negative for each of these. x_1 positive, 0, and negative. Likewise, for x_2 positive, 0, and negative. So, for x_3 again, it is positive, 0, and negative.

So, how many rules are we going to have? Definitely, it is 3 to the power of 3. 3 state variables and 3 subsets associated with each state variable. So, 3 to the power of 3 is 27 rules going to form the rule base, and the fuzzy subsets for output U are large positive, positive, 0, negative, and large negative. Now, from the fuzzy controller, we are extracting the control input u by giving x1, x2, x3, the state variables as input to the fuzzy controller.

Now, the rule base is going to be formed by having 27 rules, out of which we are going to see a few rules because of time brevity. Using the Lyapunov function, we are forming these rules. Let the Lyapunov function candidate be $v = \frac{1}{2}x^T x$

where x is a vector of three elements x_1, x_2, x_3. Expanding that, $\frac{1}{2}(x_1^2$

So, this is V. Now, we want V dot, the time derivative of the Lyapunov function candidate, to be negative definite. So, we take the time derivative of it, which is

$$\dot{v} = x_1\dot{x}_1 + x_2\dot{x}_2 + x_3\dot{x}_3$$

So, from the state space model, x3 dot is nothing but u. So, we have x1 dot, x2 dot, x3 dot on the left-hand side. From that, expanding that, we will have x3 dot equal to the control input u. Thus, it is given expanded as $= x_1x_2 + x_2x_3 + x_3u$ (from the state space model)

So, the objective of this controller is to find a control input u such that the system is stable around $x = 0$ because x equal to 0 is the equilibrium point. And from the Lyapunov theory, we know that $\dot{v} < 0$ that is, negative definite.

Now, using this, we will try forming the rule base for the controller. First, let us consider all three state variables x1, x2, x3 are positive. To make V dot negative, we analyze the following form of the expression for V dot. So, V dot be positive plus positive plus positive into u. Therefore, u should be large negative because we have taken x1, x2, x3, all three state variables, being positive.

So, the first term will be x1, x2, x2, x3. So, positive, positive, and the third term will be x3 and x3 dot. So, x3 dot is u. So, positive into u. In that case, u should be large negative in order to have V dot be negative. Now, accordingly, the rule statement is if x1 is positive p and x2 is positive p,

And x3 is positive p, then u is ln, which is large negative. That is how we are done with rule 1. Similarly, consider the case x1 equal to 0, x2 greater than 0, and x3 less than 0. So, in that case, v dot is going to be 0 plus negative plus negative into u. So, in that case, u is positive or 0 to make V dot negative. Therefore, the rule is if X1 is 0 and X2 is negative and X3 is negative, then u is positive or 0.

Likewise, for different combinations of the inputs, we get a total of 27 rules. And this type of Lyapunov-based stability-oriented fuzzy controller will be helpful in percutaneous needling intervention towards minimally invasive surgery where the control input is the needle spinning velocity and the insertion velocity. Now let us move on to the second portion of today's lecture, which is TS fuzzy control. In 1985, Takaji and Sujino proposed a powerful fuzzy model representation which states that if a non-linear system is represented by a

cluster of locally defined linear systems, then the systematic understanding of the non-linear system becomes stronger as we know the linear systems already. The TS fuzzy control has this flowchart, which is quite similar to the Mamdani controller. It is this one. The arrow mark is wrong, so from the plant, we get the information through the sensor which is a crisp value. Crisp value is converted to fuzzy through fuzzification, and the fuzzification or the fuzzy value enters the fuzzy inference mechanism. Then, the fuzzy inference mechanism does the rule evaluation. From which we get the fuzzy value that gets converted into a crisp value by defuzzification, where the center of gravity or weighted average approach is utilized, and that crisp value goes to the actuator for which the system is actuated. Now, in the fuzzy inference mechanism, the fuzzy rule based on

With output state variables represented by an expression or a model. That is the difference because straight away we would not be forming the output variable. The fuzzy output linguistic variable is obtained through an expression or a model. Precisely here we are going to see an example where we are going to use a linear model in order to obtain the fuzzy output variable.

Considering a general continuous non-linear dynamical system described by x dot equal to f of x comma u, where x and u are the n and m dimensional state and input vectors respectively. On fuzzy merging of equivalent linear systems in different operating domains using the TS fuzzy modeling, the above non-linear system can be effectively modeled. I repeat, by fuzzy merging of equivalent linear systems in different operating domains using the T-S fuzzy model, the above non-linear system can be effectively modeled. In general, a T-S fuzzy model is composed of n rules where the ith rule is given by the following form, rule I.

If x1 is t1 i where i corresponds to this rule i and x2 is t2 i accordingly it goes up to xn is tn i then x dot equal to ai x plus biu where x is an n dimensional state vector and you have your I varying for the system towards S, which means S number of rules. And the number of rules is given by Q power N. Because each rule represents a fuzzy region in the state space of N dimension, thus representing total number of rules equal to Q power N. That is the number of fuzzy regions if each state is falsified into Q fuzzy zones. In continuation of T-S fuzzy control, by providing the current state vector x and an input vector u, the T-S fuzzy model representation of the system dynamics is given by x dot equal to 1 by summation over i equal to 1 to s mu i multiplied by summation of i equal to 1 to s mu i of a i x i plus b i u. where mu i is given by the product over j equal to 1 to n mu i j of x j, where the mu i j is the membership function value of the fuzzy term T j i.

Upon representing the normalized membership function value associated with the ith value by beta i equal to

mu i upon summation over i equal to 1 to s mu i. Then, in the discrete domain, the TS fuzzy model representation of a discrete-time non-linear system is given by an expression x of k plus 1 equal to summation i equal to 1 to s beta i ai x of k. This is the final expression to get the next state variable given the current state variable. Now, take an example to understand this TS fuzzy model. Let us consider a single-link manipulator whose dynamics is given by x1 dot equal to x2, x2 dot equal to minus g by L sine of x1 plus 1 by m l square into tau, that is, tau is the control input u, where x1 equal to theta is the angle of the manipulator from the vertical axis, and x2 equal to theta dot is the angular velocity of the manipulator, and tau is the control signal, precisely the control torque for this manipulator. The parameters are mass m equal to 1 kilogram, length l equal to 1 meter, and g

acceleration due to gravity is 10 meters per second squared. Now, the question asked here is: derive any two rules that describe the TS fuzzy model of the above system. The solution here is: let the fuzzy subset of X BPL PSZE NS NL be obtained as a combination to a set X1. Precisely, the fuzzy set of variable X1 be capital X1, which is given by a set of these subsets or elements.

Accordingly, for X2, it is capital X2 equal to PL, PS, GE, NS, NL. Positive large, positive small, 0, negative small, negative large. These are the five subsets associated with the fuzzy set X2 as well as X1. So, by seeing that, you can say that two state variables x1, x2, and the fuzzy subsets associated with each state variable is 5. So, it is q power n as the number of rules.

So, 5 power 2, 25 rules, total rules. Now, two of these rules are only asked as the question. If x1 is positive large and x2 is is 0, then x dot is given by a1 x plus b1 tau. x1 is basically the joint angle variation, which is positive large. This is the x minus x, and this is basically the single manipulator. And you can have a positive angle.

This is a theta that comes here. So, positive large, let us say 85 degrees being x1, and x2, your angular velocity being 0. So, for this system, you have A and B matrices coming out to be 0, 1, minus G by L sin of X1, 0, B equal to 0, 1 by ML square. Substituting the value of G, L, and M, we get A equal to 0, 1, minus 10 times sin of X1, 0, and B matrix is 0, 1.

Now, coming to the linearized form of A given by A1 equal to The Jacobian matrix, which is given by dou f1 by dou x1, dou f1 by dou x2, and the second row is dou f2 by dou x1 and dou f2 by dou x2. The partial derivative of this matrix, that is say a1, is given now upon linearization as 0, 1, and minus 10 cos of x1, 0, 0, 1, minus 0.87 and 0, and b1 is 0, 1, because here x1 is 85 degrees, we are substituting, and accordingly, we get minus 0.870 in the second row. Therefore, the rule 1 becomes if x1 of t is positive large and x2 of t is 0, then we have x dot equal to ax plus bu, where a is this a1, b is this.

Likewise, rule 2. If x1 is positive small and x2 is 0, then x dot equals a2x plus b2 tau, where a2 is the linearized form of a with the operating point given by x1, which is positive small because x1 is positive small, say 10 degrees, and x2 is 0 again, so it is 0. So, accordingly, we will have the rule coming out to be X dot equals A and B, given by 0 1 minus 9.8 0, and B matrix is 0 1, which is obtained by substituting the value of positive small. That is how we finished our TS Fuzzy model understanding.

Now, coming to the conclusion of this lecture, we have seen today fuzzy PI, fuzzy PD, and Lyapunov function-based formation of rules. Once a rule base is ready, then we can proceed further using defuzzification and get the crisp value, and we can go for controlling the plant. Then, we have seen T-S fuzzy control theory along with a simple single-link manipulator example.

In the next class, we will be seeing fuzzy C-means clustering. Thank you very much for your time.