Design of an Optimized Fuzzy Controller for a 3R Non-planar Robotic Manipulator

Armin Ghanbarzadeh
Faculty of Mechanical Engineering
K.N. Toosi University of Technology
Tehran, Iran
ghanbarzadeh.armin@gmail.com

Esmaeil Najafi
Faculty of Mechanical Engineering
K.N. Toosi University of Technology
Tehran, Iran
najafi.e@kntu.ac.ir

***Abstract*—This paper investigates the control of a 3R robot using three different methods. The designed controllers are PID, Fuzzy logic controller (FLC), and particle swarm optimized Fuzzy logic controller (PSOFLC). The fuzzy controller calculates the required joint torques, based on the fuzzy rules and the given inputs. Using an intelligent optimization method allows adjusting fuzzy membership function parameters. The proposed method shows improved performance and a lower total error rate over its conventional counterpart. The comparative evaluation with respect to each controller is presented to validate the controller design for tracking problems. The robot modeling and control design has been simulated within MATLAB and Simulink software, where the achieved numerical results are the criteria for evaluation.**

***Index Terms*—3R robot, PID control, Fuzzy logic control, Particle Swarm optimization**

I. Introduction

Robots and robotics are stapled members of most manufacturing facilities and factories today. They are used to automate tasks that are too dangerous, repetitive, or difficult for human operators [1]. This characteristic makes them very valuable in modern industry. Implementing robots in a setup comes with its challenges such as control, safety, maintenance, economics, planning, and selection [2], [3]. Smart factories that use robots also need more educated expert personnel to program and operate them [4].

This research focuses on the controlling aspect of industrial robots. There are many ways to categorize different control methods, a popular one being model-based and non model-based control. In this paper, the focus is non model-based methods, namely PID and Fuzzy logic controllers. They do not require a mathematical model of the system, unlike model-based control techniques that require either an analytical or an experimental model. We will also attempt to optimize the Fuzzy logic controller (FLC) with the intelligent optimization method particle swarm optimized (PSO). The optimization algorithm will tune FLC membership function parameters such as triangular membership function base coordinates and output gain values. By simulating these controllers in the MATLAB Simulink environment, we can see their performance and compare the results.

One of the controlling methods of robotic manipulators was proposed in [5], where it is shown that linear feedback of generalized coordinates and their derivatives are effective for dynamic control. With this method, they used a PD controller and reported the system's behavior and response. This breakthrough was the start of the analytical and theoretical control of robotic manipulators. Other conventional feedback controllers have been proposed since then. In [6] researchers showed how an adaptive PD controller can be designed and used to ensure global asymptotic stability of the system even with unknown inertia and gravity parameters.

A robot contact language has been proposed in [7] for manipulation planning and a manual control of a social robotic arm has been discussed in [8]. A decentralized continuous sliding PID controller was designed in [9] for tracking tasks that yield semi-global stability for all closed-loop signals. Implementing the controller on a robotic arm and comparing the results with both PD and PID controllers validates the design. Sequential composition is another control approach which has been implemented in [10], [11] to enable cooperation between robotic manipulators and mobile robots [12].

Fuzzy controllers have two main benefits, the first being because of them being non model-based they can be used in situations where there is uncertainty or transient parameters in the system. Another benefit is that they have the potential to incorporate human expertise and experience in design and decision making [13], [14]. Fuzzy rules can be tuned to learn the input for a sliding mode controller [15]. This circumvents a challenge in designing sliding mode control, namely switching the manipulator during sampling. A numerical example was given to illustrate the effectiveness of the proposed method.

By combining neural networks and fuzzy logic, researchers were able to compensate dynamic and structural uncertainties [16], [17]. The model was capable of reducing the computational complexity and providing a learning ability that conventional fuzzy systems did not have. The system showed to be stable in tracking both set-point and dynamic targets. Another application of Neuro-Fuzzy control is seen in [18]. The researchers controlled robotic exoskeletons with EMG signals from physically weak persons such as the elderly or disabled. Fuzzy logic has also been used for navigating autonomous robots. In [19] the design of a fuzzy logic-based navigation algorithm for autonomous robots is designed that achieves correct environment modeling even with noisy and uncertain

978-1-6654-2094-5/21/$31.00 ©2021 IEEE

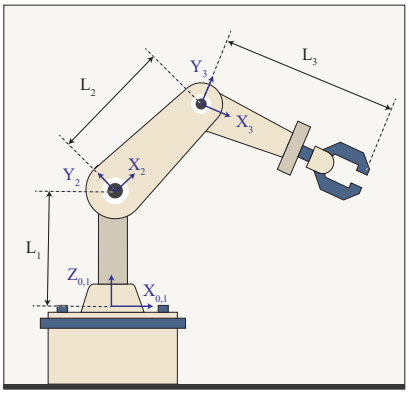


Fig. 1: Schematic of the 3R non-planar robotic manipulator

TABLE I: Robotic manipulator and object physical properties

Link	Length (m)	Weight (kg)
1	1.5	2
2	0.8	1
3	0.7	1
M	-	1

sensory data. Due to fuzzy logic being computationally light, they were able to do the processing on low-cost hardware equipment. They showed that such an algorithm performs well in a wide range of environments.

II. Robot Model with desired Trajectory

A. 3R Robot kinematic equations

The robotic manipulator used to simulate and test each controller is a three revolution (3R) joint non-planar robot as shown in Fig. 1. This is a simple but popular robot in the related literature.

The robot parameters such as link length and weight are stated in Table I. The last row (M) is the mass of the object which the robot is expected to carry. Using these values, we can calculate the inertia of each link. After calculating the inertia, the equation of motion for the 3R non-planer robot can be written as follows [20]

$$\begin{aligned}
\ddot{\theta}_1 = {} & (-670\dot{\theta}_1\dot{\theta}_2\sin(\theta_3) + 290\dot{\theta}_1(\dot{\theta}_2+\dot{\theta}_3)\sin(2\theta_2+2\theta_3) \\
& + 670\dot{\theta}_1\dot{\theta}_2\sin(2\theta_2+\theta_3) + 380\dot{\theta}_1\dot{\theta}_2\sin(2\theta_2) \\
& + 2400\tau_1)/(340\cos(2\theta_2+\theta_3) + 190\cos(2\theta_2) \\
& + 340\cos(\theta_3) + 150\cos(2\theta_2+2\theta_3) + 1500), \\
\ddot{\theta}_2 = {} & -0.38\sin(2\theta_2)\dot{\theta}_1^2 - 18\cos(\theta_2) + 4.7\tau_2 - 4.7\tau_3, \\
\ddot{\theta}_3 = {} & 16\cos(\theta_2-\theta_3) + 85\cos(\theta_2+\theta_3) + 18\cos(\theta_2) \\
& - 0.86\dot{\theta}_1^2\sin(\theta_3) - 1.7\dot{\theta}_2^2\sin(\theta_3) + 0.32\dot{\theta}_1^2\sin(2\theta_2 \\
& - \theta_3) - 0.38\dot{\theta}_1^2\sin(2\theta_2+2\theta_3) - 0.54\dot{\theta}_1^2\sin(2\theta_2 \\
& + \theta_3) + 0.38\dot{\theta}_1^2\sin(2\theta_2) + 11\tau_3 - 4.7\tau_2.
\end{aligned} \tag{1}$$

B. Control Problem and Assumptions

Like in any control application, we need to specify a control task. In our case, we want to have the robot follow a specified route from 3 consecutive points and at the end, return to its initial position (referred to as home). These points and their corresponding times are shown in Table II. Note that at each stage specified by "*", the robot will be balanced at a static reference point.

TABLE II: Reference trajectory for the robot simulation

time (s)	x (m)	y (m)	z (m)
0	1.5	0	1.5
2	0	0	3
2-10*	0	0	3
20	0.7	0.7	1.3
20-22*	0.7	0.7	1.3
22	1.5	0	1.5
22-30*	1.5	0	1.5

*Stationary stage, just balancing

A trajectory planning system will calculate the necessary reference points for the robotic manipulator at each time step. The system takes in the desired cartesian points in the workspace and calculates the joint angle and velocities using a second-order parabolic equation. A schematic of the robot's movement can be seen in Fig. 2.

The maximum allowable torque to the system is limited as

$$|T_{motor}| < 20, \tag{2}$$

where this amount is appropriate for a robot of this size and application. Limiting torque is necessary for comparing such a diverse set of controllers such as PID and Fuzzy logic. We achieved this by adding a saturation block in the Simulink diagram for the PID-controlled system. To achieve the same effect in the FLC controller, we limited the output membership function range.

C. Evaluation Criteria

In order to evaluate a controller, there are two general approaches that one could take. The first is to compare some standard performance measures such as stability margin, overshoot, settling, or rise time [21]. Another method is to calculate the total error with respect to some definition. A few popular methods of calculating the total error are stated by

$$ISE = \int e(t)^2 dt, \; IAE = \int |e(t)| dt. \tag{3}$$

In this work, the ISE is used as the main performance measure when comparing results. The ISE formula amplifies larger errors so the controller is further penalized for large error values. This helps the optimization algorithm to converge faster. We will also calculate the IAE at the end to get a more relatable error value.

III. Controller Design

In this section, the three controllers designed and used will be illustrated. First a simple PID controller, then Fuzzy logic (FLC) and PSO optimized Fuzzy logic controller (PSOFLC) will be discussed.

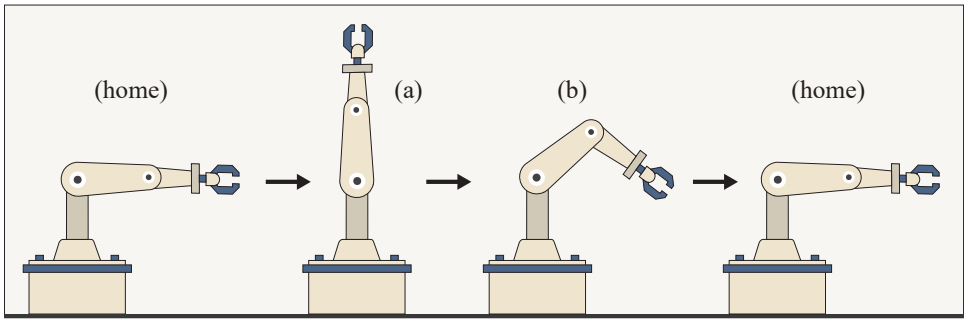


Fig. 2: The 3R robotic manipulator movements at different points of the trajectory for the given task

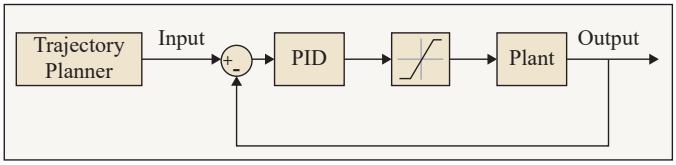


Fig. 3: Diagram of the simulated system in Simulink for the PID controller

A. PID Controller

A Simulink system was set up as shown in Fig. 3 to simulate the system controlled by a PID controller. The error signal is shown in (4). Unlike the fuzzy controller discussed in a later part of this paper, the PID controller does not need an angular speed error. This is because the controller has a derivative part to calculate the error change rate. Since we need three control signals, we must use three PID controllers. The parameters K_P = 50, K_I = 50, K_D = 2 were chosen such that the system response is acceptable considering the torque limitations stated earlier, given as

$$e_i = \theta_{i,desired} - \theta_i \quad (i = 1, 2, 3). \tag{4}$$

B. Fuzzy logic controller

In this part of the research, we attempt to design an FLC for controlling the 3R robotic manipulator. The Simulink system gives us feedback of both angles θ and angular velocities ω. So we can use these for inputs when designing the FLC. The fuzzy inference system has a total of 6 inputs, corresponding to errors in angles (θ_i) and angular velocity (ω_i)

$$e_i = \theta_{i,desired} - \theta_i \quad (i = 1, 2, 3), \tag{5}$$

$$e_i = \omega_{j,desired} - \omega_j \quad (i = 4, 5, 6, j = 1, 2, 3). \tag{6}$$

Using these inputs and by evaluating the fuzzy inference rules given to it, the FLC outputs three torque (τ) values corresponding to the required torque for each link. A Simulink system was designed as shown in Fig. 4 to simulate the system controlled by an FLC. Note that two gains were added before and after the controller since the input-output ranges were set at [-1, 1].

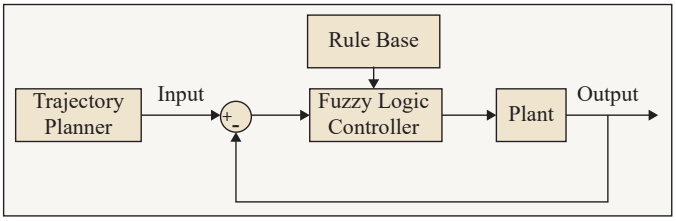


Fig. 4: Diagram of the simulated system in Simulink for the Fuzzy logic controller (FLC)

TABLE III: Input membership function parameters

Linguistic term	Parameters
N	$[-1, -1, -0.5, 0]$
Z	$[-0.5, 0, 0.5]$
P	$[0, 0.5, 1, 1]$

1) FLC Membership Functions and Gains: In the selection of the membership functions, we first chose a simple but popular combination. The inputs each have three membership functions, two trapezoidal and a single triangular, as depicted in Table III. The outputs are a bit more detailed with each having 5 triangular MF's, as shown in Fig. 5. As for the Gain values, K_{Inputs} = 4 and $K_{Outputs}$ = 100 were chosen, as given in Table IV.

2) Fuzzy Rules: Since there are six inputs, each with three membership functions, we can write a total of 3^6 = 729 rules. To address this problem, we can consider that each motor torque only depends on its own angular and angular velocity

TABLE IV: Output membership functions

Linguistic term	Parameters
NB	$[-1, -1, -0.5]$
NS	$[-1, -0.5, 0]$
Z	$[-0.5, 0, 0.5]$
PS	$[0, 0.5, 1]$
PB	$[0.5, 1, 1.5]$

TABLE V: Fuzzy inference rules

	N	**Z**	**P**
N	NB	N	Z
Z	N	Z	P
P	Z	P	PB

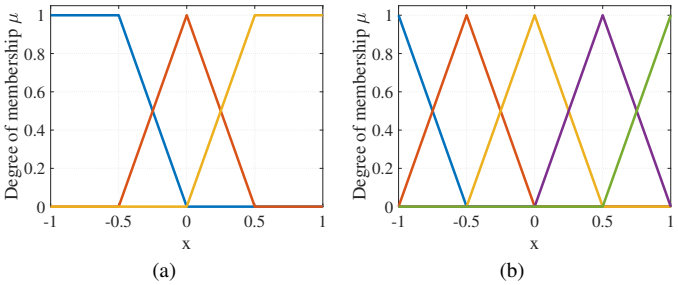


Fig. 5: Input-Output membership functions of the FLC controller. (a) Input membership functions, membership functions in order of left to right: N-Z-P; (b) Output membership functions, membership functions in order of left to right: NB-NS-Z-PS-PB

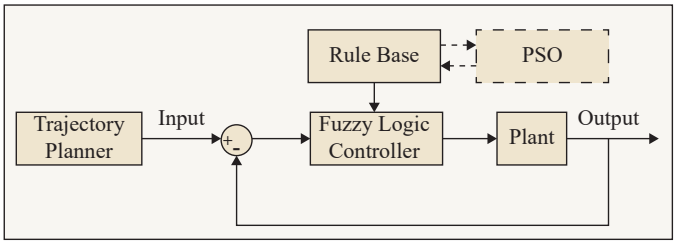


Fig. 6: Diagram of the simulated system in Simulink for the PSO optimized Fuzzy logic controller (PSOFLC)

error. this reduces the required rules to $3^2 \times 3 = 27$. The rules are shown in Table V. The rules are the same for each one of the three joint actuators, so only one set of nine rules have been tabulated.

C. PSO optimized Fuzzy logic controller

To obtain better results from the FLC, we can fine-tune its parameters with optimization algorithms. Popular nature-based optimization algorithms include Genetic algorithms and Particle swarm optimization. In this paper, we will use PSO since it has shown better results for such problems. The schematic of such a system is shown in Fig. 6.

In total, eighteen parameters have been tuned using PSO, six for the input functions (bases of triangular and trapezoidal membership functions), three for the outputs, and nine parameters for the input and output gains. Note that the rules are not needed to be optimized. Hyper-parameters for the algorithm was mostly left as default, except the SwarmSize = 25 and MaxGenerations = 40. The initial population was also set to be the designed FLC controller. Limits on membership function parameters were [0.1, 0.9], input gain limits were [0.1, 10] and output gain limit was [0.1, 100].

IV. Results

A. Optimized Membership Functions and Gains

The optimized membership functions are shown in Fig. 7 and Fig. 8. As it can be seen in the figures, membership functions for angular error have been squashed near the origin.

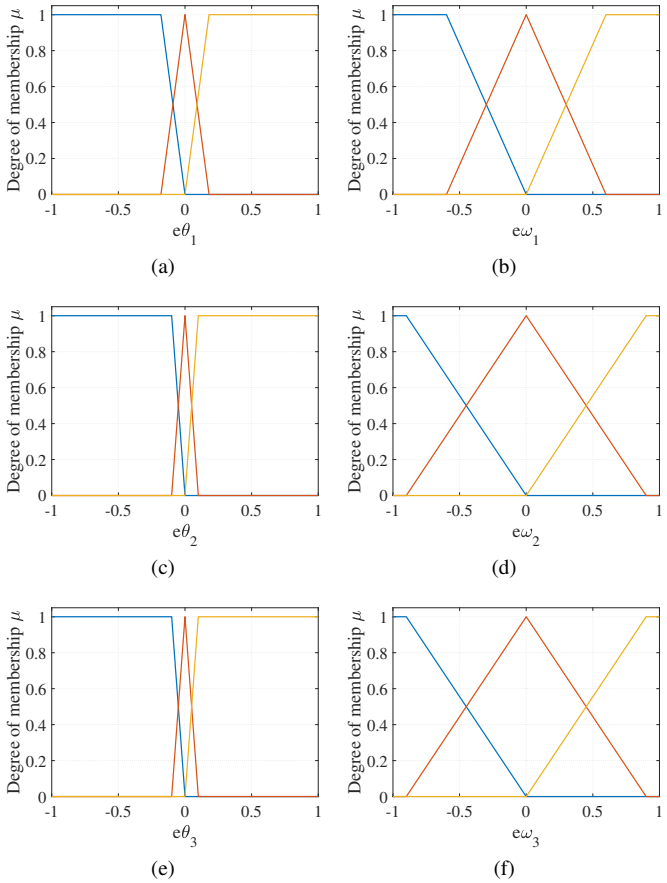


Fig. 7: The PSO optimized membership functions for inputs. Membership functions in order of left to right: N-Z-P

TABLE VI: The optimized Input-Output gains for the PSO Fuzzy logic controller

Input	Signal	Gain
Input1	θ_1	9.98
Input2	ω_1	10
Input3	θ_2	10
Input4	ω_2	0.1
Input5	θ_3	10
Input6	ω_3	0.1
Output1	τ_1	100
Output2	τ_2	100
Output3	τ_3	100

This shows that for even low errors in the θ values, the controller still outputs a fair amount of torque. This helps the system to eliminate any steady-state errors as we will see in the results section. In contrast, in the case of angular velocity membership functions, base points were pushed away from the origin towards the endpoints. This way in case of large errors, the links could accelerate quickly but as the errors diminish, it outputs less torque to prevent overshooting. A similar situation to the angular errors happens to the output membership functions. The optimized values are near the origin, to eliminate the steady-state error and always outputting at least some little amount of torque.

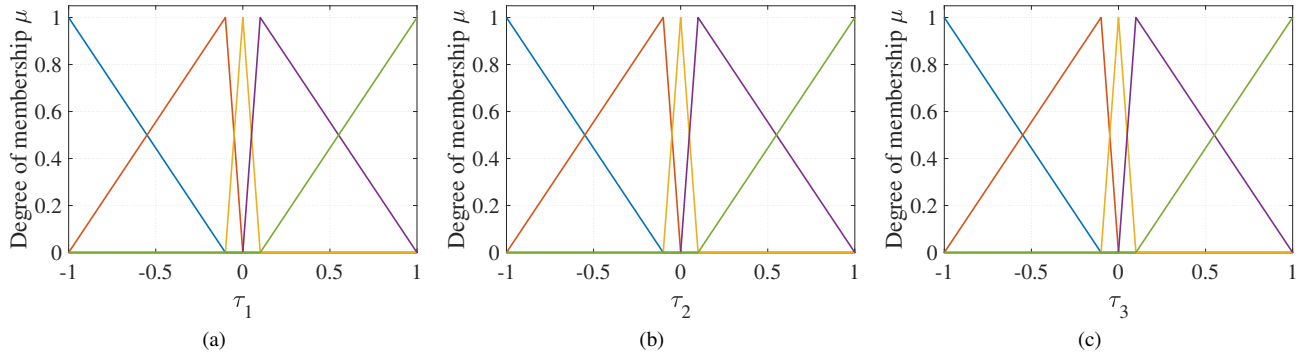


Fig. 8: The PSO optimized membership functions for outputs: NB-NS-Z-PS-PB respectively

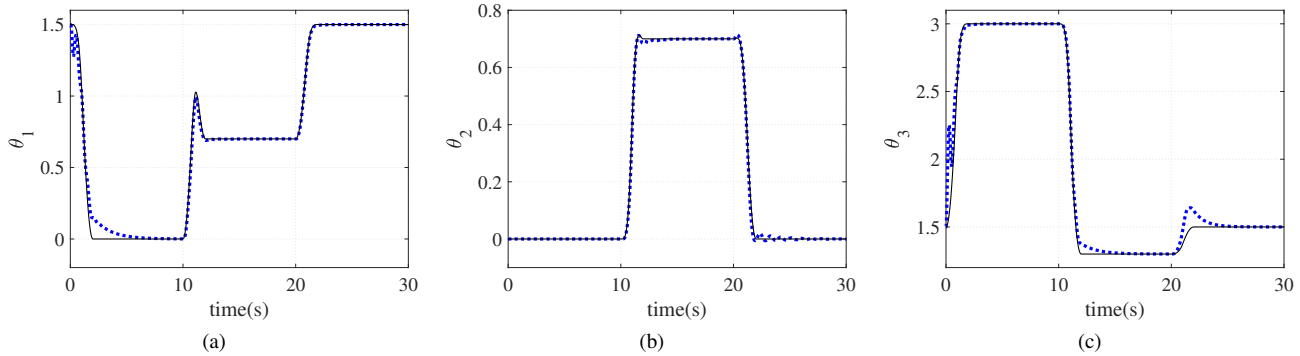


Fig. 9: Manipulator end-effector position with PID controller. (a) End-effector x position, (b) End-effector y position, (c) End-effector z position. Black line: Desired path; Blue dashed line: Actual path

The gains are also reported in Table VI. The gain values are mostly the maximum amount in the limits, except for ω_1 and ω_2. This shows that these two values are of lesser importance and should be considered much less compared to the other inputs in this application.

B. Manipulator positions

By simulating and implementing the controllers designed in the previous sections, we can see each system response and compare the results. Fig. 9, Fig. 10, and Fig. 11 show end manipulator positions following desired trajectories with a PID, FLC and PSOFLC controllers respectively. As it can be seen, the PSOFLC controller had the best performance, following FLC and PID. Some performance measures and controller settings have been summarized in Table VII. The results show that the FLC controller outperforms the PID controller by a magnitude of 4. The PSOFLC controller results show several orders of magnitude improvement compared to the other two methods.

V. Conclusion

In this paper, a particle swarm optimized fuzzy logic controller has been designed and compared with conventional methods such as Fuzzy logic and PID controllers. The simulation results conclude that at the cost of added computation, the proposed PSO Fuzzy logic controller outperforms the conventional methods. For future works, the proposed algorithm can be combined with PID controllers or Neuro-Fuzzy controllers. Moreover, these algorithms can be simulated in other software like ROS/Gazebo and deployed on real industrial robotic manipulators to further study and validation of the controllers.

References

[1] R. Hirschfeld, F. Aghazadeh, and R. Chapleski, "Survey of robot safety in industry," *International Journal of Human Factors in Manufacturing*, vol. 3, no. 4, pp. 369–379, 1993.

[2] M. Mirzae, N. Karimi, and E. Najafi, "ROS-based motion planner for Gazebo-simulated rescue robots in RoboCup," in *Proceedings of the 21st International Conference on Research and Education in Mechatronics*, 2020, pp. 1–5.

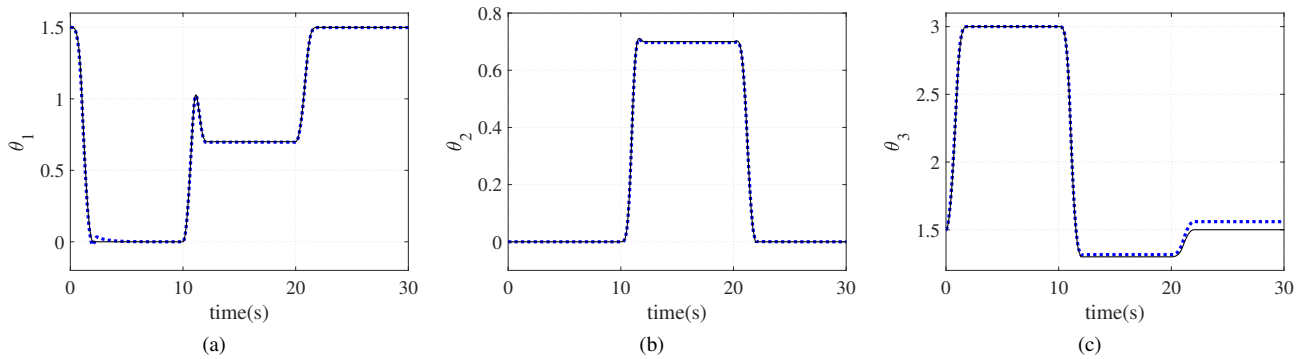
[3] V. Y. Philippart, K. O. Snel, A. M. de Waal, J. S. Jeedella, and E. Najafi, "Model-based design for a self-balancing robot using the Arduino microcontroller board," in *roceedings of the 23rd International Conference on Mechatronics Technology*, 2019, pp. 1–6.

[4] E. Najafi, T. Laugs, P. N. Rubio, and S. Alers, "An implementation of AutomationML for an Industry 4.0 case study: RoboCup@ Work," in *Proceedings of the 20th International Conference on Research and Education in Mechatronics*, 2019, pp. 1–6.

[5] M. Takegaki and S. Arimoto, "A New Feedback Method for Dynamic Control of Manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 103, no. 2, pp. 119–125, 1981.

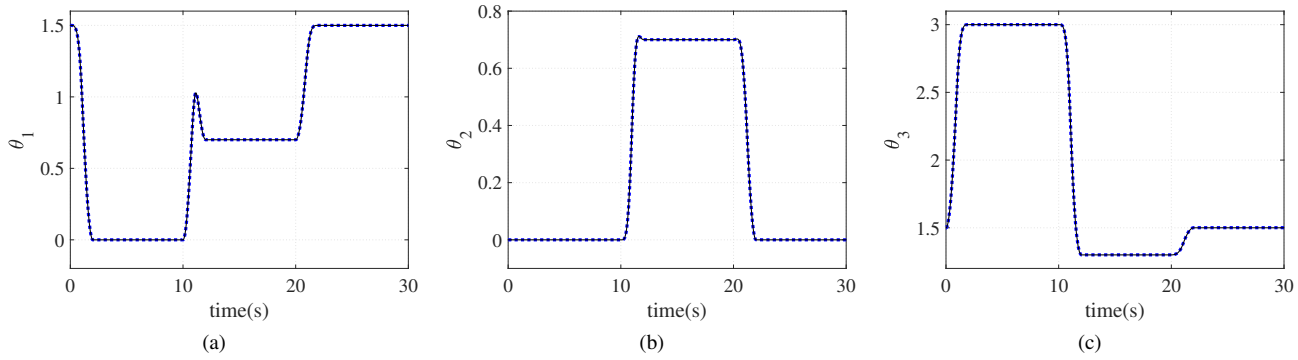
[6] P. Tomei, "Adaptive PD controller for robot manipulators," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 4, pp. 565–570, 1991.

[7] E. Najafi, A. Shah, and G. A. Lopes, "Robot contact language for manipulation planning," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 3, pp. 1171–1181, 2018.



(a) (b) (c)

Fig. 10: Manipulator end-effector position with Fuzzy logic controller. (a) End-effector x position, (b) End-effector y position, (c) End-effector z position. Black line: Desired path; Blue dashed line: Actual path



(a) (b) (c)

Fig. 11: Manipulator end-effector position with PSO Fuzzy logic controller. (a) End-effector x position, (b) End-effector y position, (c) End-effector z position. Black line: Desired path; Blue dashed line: Actual path

TABLE VII: Summary of the results

Method	**ISE** ($\times 10^{-3}$)	**IAE** ($\times 10^{-3}$)	**Hyper parameters**	**Required human experience**	**Computation cost**
PID	204	1380	3	Low	Low
FLC	41	930	~ 30	High	Low
PSOFLC	0.074	12.5	~ 10	Low	Very High

[8] P. Khajehpour and E. Najafi, "Design and manual control of a 3 degrees of freedom social robotic manipulator," in *Proceedings of the 21st International Conference on Research and Education in Mechatronics*, 2020, pp. 1–5.

[9] V. Parra-Vega, S. Arimoto, Y.-H. Liu, G. Hirzinger, and P. Akella, "Dynamic sliding PID control for tracking of robot manipulators: Theory and experiments," *IEEE Transactions on Robotics and Automation*, vol. 19, no. 6, pp. 967–976, 2003.

[10] E. Najafi and G. A. Lopes, "Towards cooperative sequential composition control," in *Proceedings of the 55th IEEE Conference on Decision and Control*, 2016, pp. 4758–4763.

[11] E. Najafi, R. Babuška, and G. A. Lopes, "An application of sequential composition control to cooperative systems," in *Proceedings of the 2015 10th International Workshop on Robot Motion and Control*, 2015, pp. 15–20.

[12] G. A. Lopes, E. Najafi, S. P. Nageshrao, and R. Babuška, "Learning complex behaviors via sequential composition and passivity-based control," in *Handling Uncertainty and Networked Structure in Robot Control*. Springer, 2015, pp. 53–74.

[13] J. F. Baldwin and N. Guild, "Modelling controllers using fuzzy relations," *Kybernetes*, 1980.

[14] C.-C. Lee, "Fuzzy logic in control systems: fuzzy logic controller. I," *IEEE Transactions on systems, man, and cybernetics*, vol. 20, no. 2, pp. 404–418, 1990.

[15] Y. Ohtani and T. Yoshimura, "Fuzzy control of a manipulator using the concept of sliding mode," *International journal of systems science*, vol. 27, no. 2, pp. 179–186, 1996.

[16] L. Peng and P.-Y. Woo, "Neural-fuzzy control system for robotic manipulators," *IEEE Control Systems Magazine*, vol. 22, no. 1, pp. 53–63, 2002.

[17] E. Najafi, R. Babuška, and G. A. Lopes, "Learning sequential composition control," *IEEE transactions on cybernetics*, vol. 46, no. 11, pp. 2559–2569, 2015.

[18] K. Kiguchi, T. Tanaka, and T. Fukuda, "Neuro-fuzzy control of a robotic exoskeleton with emg signals," *IEEE Transactions on fuzzy systems*, vol. 12, no. 4, pp. 481–490, 2004.

[19] F. Cupertino, V. Giordano, D. Naso, and L. Delfine, "Fuzzy control of a mobile robot," *IEEE robotics & automation magazine*, vol. 13, no. 4, pp. 74–81, 2006.

[20] K. M. Lynch and F. C. Park, *Modern robotics*. Cambridge University Press, 2017.

[21] K. Krishnakumar and D. E. Goldberg, "Control system optimization using genetic algorithms," *Journal of Guidance, Control, and Dynamics*, vol. 15, no. 3, pp. 735–740, 1992.