User-Tailored Fuzzy-Based Grasp Strength Regulation in Myocontrolled Robotic Hands

Mohammad Sheikhsamad*, Roberto Meattini†, Davide Chiaravalli†, Raúl Suárez*, Jan Rosell*, Gianluca Palli†
*Institute of Industrial and Control Engineering (IOC), Universitat Politècnica de Catalunya (UPC), 08028 Barcelona, Spain
†Dept. of Electrical, Electronic and Information Engineering (DEI),University of Bologna, Viale del Risorgimento 2, 40136 Bologna, Italy.
Emails: mohammad.sheikhsamad@upc.edu, roberto.meattini@unibo.it, davide.chiaravalli2@unibo.it,
raul.suarez@upc.edu, jan.rosell@upc.edu, gianluca.palli@unibo.it

***Abstract*—Myocontrolled robotic hands require accurate and responsive control to regulate grasp strength effectively. However, many human-in-the-loop (HITL) control systems still lack robust closed-loop solutions for fine grip force regulation, limiting their performance. This paper presents a novel control system for myocontrolled hands that combines contact force sensing and vibrotactile feedback to enable more natural and precise grasp interaction. The system features an advanced force controller based on fuzzy logic, with parameter optimization guided by user preferences collected through a graphical user interface (GUI) using Global Learning of Input–Output Strategies from Pairwise Preferences (GLISp). It is compared against heuristic model and neural network based controllers. The system was validated through real-world experiments using the AR10 robotic hand with OptoForce fingertip sensors, demonstrating improved adaptability and fine force regulation capabilities for the user.**

***Index Terms*—Grasp strength regulation, Myocontrolled, Vibrotactile, Fuzzy logic, Neural network**

I. Introduction

The human hand can perform complex grasps with great precision and flexibility by coordinating numerous joints, muscles, and sensory signals. This capability allows us to handle a wide range of objects with ease. However, despite years of research, achieving similar grasping abilities in robotic hands remains a major challenge.

A Human–Robot Interface (HRI) is essential for controlling robotic hands in tasks that require precise grasp force, such as telemanipulation, assistive robotics, and prosthetics. For example, over 57 million people worldwide live with limb amputations [1]. Surface electromyography (sEMG) is a non-invasive HRI method that uses muscle signals recorded from the skin to control movement or recognize patterns. It also plays a key role in Human-in-the-Loop (HITL) systems, where users actively adjust control during operation.

Robotic grasping requires precise and flexible control strategies, which are typically categorized into three main types: (i) Heuristic model-based: These methods use mathematical representations of the robotic hand's kinematics and dynamics, contact interactions, and environmental conditions to compute control actions [2]; (ii) Machine learning-based: These methods rely on data-driven approaches that enable robotic systems to autonomously learn control policies through experience, interaction with the environment, or demonstrations [3]; (iii) HITL-based: These methods use bio-signals, such as postural synergies [4], to translate human motor intent into control commands.

Effective grasping especially in prosthetic robotic systems also requires reliable sensory feedback. However, most current HRIs rely primarily on visual feedback, where users observe the prosthesis to assess performance. Other forms of feedback are often lacking. Vibrotactile feedback, which conveys grasp force through vibration intensity, has shown promise as a simple, non-invasive method for providing force-related information [5].

This paper aims to improve the precision and stability of strength control in myocontrolled robotic hands during tripod grasps. We propose a non-invasive, HITL control architecture using a fuzzy-based force controller optimized through Global Learning from Imprecise and Surrogate Preferences (GLISp). The approach is compared with heuristic model-based and neural network (NN) force controllers. Building on previous work [5], our main contributions are: (i) a novel fuzzy force controller with a graphical user interface (GUI) to collect user preferences for GLISp optimization, representing the first application of preference-based tuning for fuzzy force control in this field, (ii) a new NN controller trained on data generated by the fuzzy system, enabling the network to learn human-optimized policies without requiring large-scale datasets, which is a novel approach in this field, and (iii) experimental validation on the AR10 robotic hand[1] equipped with OptoForce[2] fingertip sensors.

II. System Set-Up

The proposed control architecture follows the approach described in [5] and includes five main modules, as illustrated in Fig. 1 and described in the following subsections.

A. HRI Components, AR10 Hand, and Contact Force Sensors

The HRI setup includes an sEMG system and a vibrotactile feedback module (see Fig. 2(b, c)). sEMG signals were recorded using the OYMotion armband[3], a wearable device equipped with dry electrodes positioned around the forearm. Data were sampled at 1 kHz and transmitted via Bluetooth to a computer. Moreover, vibrotactile feedback was provided by a small vibration motor embedded in a custom wristband and connected to a microcontroller via a Grove interface[4].

[1] www.active8robots.com
[2] www.onrobot.com
[3] www.oymotion.com
[4] www.seeedstudio.com

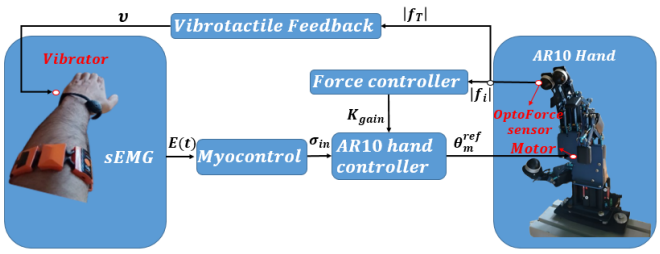

Fig. 1: Control architecture for grasp strength regulation.

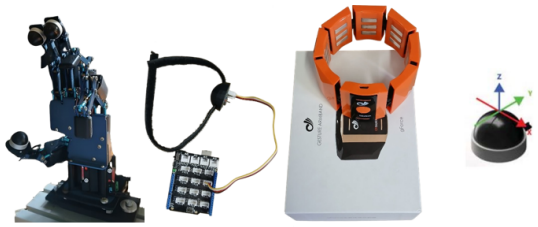

Fig. 2: Illustration of (a) the AR10 robotic hand, (b) the vibrotactile module, (c) the OYMotion sEMG armband, and (d) the OptoForce sensor.

The motor operated at 1 Hz, with vibration intensity controlled through pulse-width modulation (PWM). A normalized input signal, $\nu \in [0, 1]$, adjusted the duty cycle of a 1-second binary waveform and vibration was delivered during the high phase and suppressed during the low phase. The AR10 robotic hand used in this study (Fig. 2(a)), developed by Active8 Robots, is a humanoid, anthropomorphic hand designed for grasping and manipulation. It features 10 joints ($n_J = 10$) and 10 linear actuators ($n_m = 10$), powered at 12 V with a peak current of 5 A and a maximum power of 60 W. The hand has a sturdy aluminum frame combined with lightweight plastic components to ensure both precision and durability. It supports USB and serial communication, provides four analog inputs for external sensors, and is compatible with ROS[5]. To measure contact forces, three OptoForce OMD-20-SE-40N sensors are mounted on the thumb, index, and middle fingertips using 3D-printed supports (Fig. 2(d)). Each sensor captures forces along the x, y, and z axes. The total force at fingertip i is computed as:

$$|f_i| = \sqrt{f_{x,i}^2 + f_{y,i}^2 + f_{z,i}^2}, \quad i \in \{T, I, M\}, \tag{1}$$

where T, I, and M refer to the thumb, index, and middle finger, respectively.

B. Myocontrol

The myocontrol system used in this study (Fig. 3) processes sEMG signals and outputs a control signal, σ_{input}, representing the user's hand closure level. To prepare the sEMG signals, a pre-processing pipeline was used, including a 50 Hz notch filter (to remove power-line noise), a 20 Hz high-pass filter

[5]www.ros.org

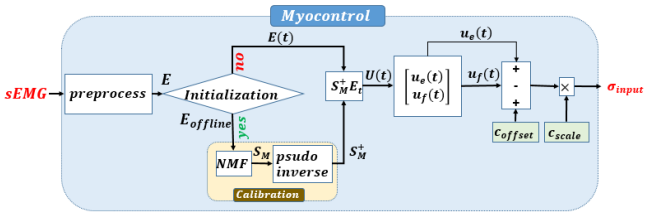

Fig. 3: Myocontrol module for grasp strength regulation.

(to remove low-frequency noise), and a 200 ms sliding window to calculate the Root Mean Square (RMS) value [6]. The processed sEMG signals are stored in an input matrix $E \in \mathbb{R}^{n_E \times n}$, where $n_E = 2$ is the number of sEMG channels, and n is the number of samples. A short calibration phase is first performed, where the user opens and closes their hand twice. The resulting data, E_{offline}, is then factorized using Non-negative Matrix Factorization (NMF) [7]. In this context, NMF helps extract muscle synergies related to the user's hand motions [4] as follows:

$$E_{\text{offline}} \approx S_M U_{\text{offline}} \tag{2}$$

where $S_M \in \mathbb{R}^{n_E \times n_U}$ is the muscular synergy matrix, and $U_{\text{offline}} \in \mathbb{R}^{n_U \times n}$ is the offline neural drive matrix, with $n_U = 2$ representing the number of supraspinal neural drives (i.e., control signals originating from the brain). After calibration, the neural drives during real-time operation are computed as:

$$U(t) = S_M^+ E(t) \tag{3}$$

In this setup, $U(t) = \begin{bmatrix} u_e(t) & u_f(t) \end{bmatrix}^T \in \mathbb{R}^2$ represents the neural drives, and $E(t) = \begin{bmatrix} e_1(t) & e_2(t) \end{bmatrix}^T \in \mathbb{R}^2$ contains the current sEMG signals. The neural drives are calculated using the pseudoinverse of the synergy matrix, S_M^+. Then, the hand closure level is estimated as:

$$\sigma_{\text{input}} = c_{\text{scale}}(u_e(t) - u_f(t) + c_{\text{offset}}) \tag{4}$$

Here, c_{scale} and c_{offset} are constants used to map σ_{input} into the range $[-0.5, 0.5]$. This signal is then provided as an input to the AR10 hand controller as shown in Fig. 1.

C. AR10 Hand Controller

Let q_{ij}^{ref} denotes the reference position of joint j of finger f_i, with $i \in \{T, M, I\}$ and $j \in \{1, 2\}$;

$\mathbf{q}_i^{\text{ref}} = \begin{bmatrix} q_{i1}^{\text{ref}} & \dots & q_{ij}^{\text{ref}} \end{bmatrix}^T \in \mathbb{R}^2$ be the reference finger configuration of finger f_i;

$\mathbf{Q}^{\text{ref}} = \begin{bmatrix} \mathbf{q}_1^{\text{ref}} & \dots & \mathbf{q}_i^{\text{ref}} & \dots & \mathbf{q}_I^{\text{ref}} \end{bmatrix}^T \in \mathbb{R}^{n_J}$ be the reference hand configuration vector.

The velocity control law is based on postural synergies, is illustrated in Fig. 4 and is defined as:

$$\dot{\boldsymbol{Q}}^{\text{ref}} = K_{\text{gain}} S_P \alpha_v \sigma \tag{5}$$

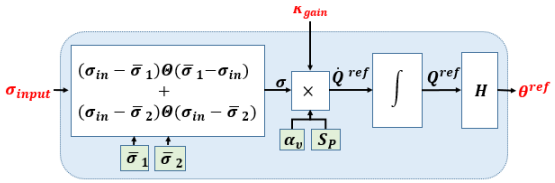

Fig. 4: Velocity-based hand controller, Θ is the Heaviside step function.

where $S_P \in \mathbb{R}^{n_J \times n_S}$ is the postural synergy matrix, and $\dot{\mathbf{Q}}_i^{\text{ref}} \in \mathbb{R}^{n_J}$ is the reference hand velocity, with $n_S = 3$ representing the number of eigen-postures (principal components). $K_{\text{gain}} \in \mathbb{R}^{n_J \times n_J}$ is a diagonal gain matrix. The control input $\sigma \in \mathbb{R}$ is driven by the myocontrol system, and $\alpha_v \in \mathbb{R}^{n_S}$ defines the activation levels of each synergy. For example, a tripod grasp would activate the thumb, index, and middle synergies, while the others remain zero. Both σ and α_v are defined as:

$$\sigma = \begin{cases} \sigma_{\text{input}} - \bar{\sigma}_1, & \text{if } \sigma_{\text{input}} \leq \bar{\sigma}_1 \\ 0, & \text{if } \bar{\sigma}_1 < \sigma_{\text{input}} < \bar{\sigma}_2 \\ \sigma_{\text{input}} - \bar{\sigma}_2, & \text{if } \sigma_{\text{input}} \geq \bar{\sigma}_2 \end{cases} \tag{6}$$

where, $\bar{\sigma}_1, \bar{\sigma}_2 \in \mathbb{R}$ are threshold values that define a dead zone, typically set to -0.1 and 0.1, respectively, and:

$$\alpha_v = \frac{\alpha_{\text{closed}} - \alpha_{\text{open}}}{\|\alpha_{\text{closed}} - \alpha_{\text{open}}\|} \tag{7}$$

where $\alpha_{\text{open}}, \alpha_{\text{closed}} \in \mathbb{R}^{n_S}$ represent the fully open and fully closed hand postures in the synergy space. The reference hand configuration $\boldsymbol{Q}^{\text{ref}} \in \mathbb{R}^{n_J}$ is then updated by integrating the velocity over time:

$$\boldsymbol{Q}^{\text{ref}}(t + \Delta t) = \boldsymbol{Q}^{\text{ref}}(t) + \dot{\boldsymbol{Q}}^{\text{ref}} \cdot \Delta t \tag{8}$$

where Δt denotes the time step of the control loop.

D. Force Controller

Stable and adaptive grasping requires effective force regulation, handled by the force controller module (see Fig. 1). This module takes the contact forces f_i, measured by the OptoForce sensors for each finger $i \in \{T, I, M\}$, and computes the diagonal gain matrix $K_{\text{gain}} \in \mathbb{R}^{n_J \times n_J}$, (used in Eq. 5) as:

$$\mathbf{K}_{\text{gain}} = \text{diag}(K_T, K_T, \ldots, K_I, K_I, \ldots, K_M, K_M) \tag{9}$$

Each gain value $K_i \in \mathbb{R}$ is computed using one of three methods: Heuristic model-based, Fuzzy-based, and Neural network-based. Heuristic model-based force controller using a piecewise control law to calculate the gain values, $K_i \in \mathbb{R}$:

$$K_i = \begin{cases} \bar{K}\left(\dfrac{1-\gamma}{\gamma}\dfrac{|f_i|}{\bar{f}} + 1\right) & \text{if } |f_i| < \bar{f} \\ \bar{K}/\gamma & \text{if } |f_i| \geq \bar{f} \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

where $\bar{f} \in \mathbb{R}^+$ is the minimum force required to detect contact, $\bar{K} \in \mathbb{R}^+$ is the default gain when no contact is present, and $\gamma \in \mathbb{R}^+$ is a scaling factor that reduces the gain when contact is detected for a specific finger. The other controllers are described later in Sections III and IV.

E. Vibrotactile Feedback

To provide users with intuitive feedback on grasp strength, a vibrotactile signal $\nu \in [0, 1]$ was generated based on the error between the measured thumb contact force $|f_T|$ and its target value F_{ref}. In tripod grasps, the thumb force is treated as an estimate of the internal grasp force, as it approximately balances the forces applied by the index and middle fingers. The normalized vibrotactile signal is computed using a piecewise function:

$$\nu = \begin{cases} 0, & \text{if } \big||f_T| - \bar{f}_T\big| \leq \beta \\ \dfrac{\big||f_T| - \bar{f}_T\big|}{c_{\text{norm}}}, & \text{if } \beta < \big||f_T| - \bar{f}_T\big| \leq c_{\text{norm}} \\ 1, & \text{if } \big||f_T| - \bar{f}_T\big| > c_{\text{norm}} \end{cases} \tag{11}$$

where c_{norm} is a scaling constant that defines how the feedback signal is normalized, and β is a tolerance margin. In our experiments, β was set to 0.03 N to reflect $\pm 5\%$ of the highest target force level. The signal ν was then sent to a vibration motor.

III. Fuzzy-Based Force Controller

Compared to heuristic model-based force controllers, the proposed fuzzy-based force controller augmented with a GUI for real-time user feedback and optimized through GLISp offers greater adaptability and user-centered tuning, enabling more intuitive and personalized grasp strength regulation in dynamic tasks.

A. Fuzzy Inference System for Gain Modulation

To compute the gain $K_i \in \mathbb{R}$ for each finger, $i \in \{T, I, M\}$, based on tactile feedback, a Fuzzy Inference System (FIS) was designed with two inputs: (i) the tracking force error, $e_{fi} = F_{ref} - f_i \in \mathbb{R}$, and (ii) the rate of change of contact force $\dot{f}_i = \frac{df_i}{dt}$. These inputs capture both the intensity and the dynamic behavior of the fingertip contact interaction, which are essential to achieve responsive and adaptive force control during grasping. The FIS uses these inputs to compute the gain through the following four layer processing:

(i) Fuzzification layer: Each variable $x \in \{ e_{fi},\ \dot{f}_i,\ K_i \}$ is represented by a fuzzy set $m_x \in \{ m_{e_{fi}},\ m_{\dot{f}_i},\ m_{K_i} \}$, which describes its values using predefined categories or fuzzy labels (i.e. linguistic labels), and is defined as:

$$\begin{aligned} m_{e_{fi}} &= \{\textit{large neg.},\ \textit{negative},\ \textit{zero},\ \textit{positive},\ \textit{large pos.}\} \\ m_{\dot{f}_i} &= \{\textit{falling},\ \textit{steady},\ \textit{rising}\} \\ m_{K_i} &= \{\textit{very low},\ \textit{medium},\ \textit{very high}\} \end{aligned}$$

Fuzzy sets are represented mathematically using membership functions (MFs), $\mu_{m_x}(x)$, which assign a degree of membership to each input. While several types of MFs can be used in fuzzy systems, this work employs triangular membership functions due to their simplicity, computational efficiency, and suitability for real-time applications. Each MF is defined as:

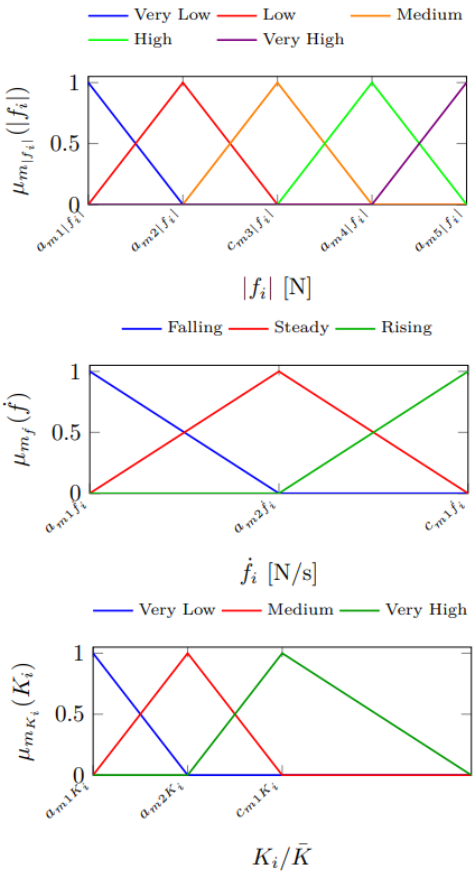

Fig. 5: Fuzzy membership functions used in the force controller: tracking force error e_{fi} (top), force rate $\dot{f}_i$ (middle), and output gain K_i (bottom).

$$\mu_{m_x}(x) = \max\left(\min\left(\frac{x - a_{m_x}}{b_{m_x} - a_{m_x}}, \frac{c_{m_x} - x}{c_{m_x} - b_{m_x}}\right), 0\right) \tag{12}$$

where, a_{m_x}, b_{m_x}, and c_{m_x} define the start, peak and end points of the triangular MF used for both input and output variables.

(ii) Rule Evaluation (Product) Layer: After fuzzifying the inputs, the FIS evaluates a set of fuzzy rules that link input combinations to output actions. These rules focus on the most important and intuitive responses for adaptive force control, as shown in Fig. 5. Each rule's activation (firing strength) is calculated by multiplying the membership degrees of the two inputs:

$$w_{go} = \mu_{m_{ge_{fi}}}(e_{fi}) \cdot \mu_{m_{o\dot{f}_i}}(\dot{f}_i), g = 1, \ldots, 5; o = 1, \ldots, 3 \tag{13}$$

where, $m_{ge_{fi}}$ is the g-th fuzzy lable (e.g, *negative*, *zero*, *positive*) in the fuzzy set $m_{e_{fi}}$.

(iii) Normalization Layer: This layer calculates the relative strength of each rule by dividing its firing strength by the total across all rules:

$$\bar{w}_{go} = \frac{w_{go}}{\sum_{g=1}^{5}\sum_{o=1}^{3} w_{go}}, \quad g = 1, \ldots, 5; \quad o = 1, \ldots, 3 \tag{14}$$

where, $\bar{w}_{go}$ denotes the *normalized firing strength* for the rule using the g-th fuzzy label of $m_{e_{fi}}$.

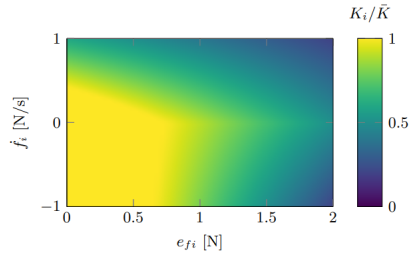

Fig. 6: Heatmap of the fuzzy output gain K_i as a function of the tracking force error e_{fi} and its rate of change $\dot{f}_i$.

(iv) Defuzzification Layer: This layer combines all the normalized firing strengths, $\bar{w}_{go}$, to compute the final output gain as:

$$K_i = \sum_{g=1}^{5}\sum_{o=1}^{3} \bar{w}_{go} \cdot z_{go}, \quad i \in \{T, I, M\} \tag{15}$$

In the zero-order Sugeno-type FIS used in this work [8], each $z_{go} \in \mathbb{R}$ is a fixed output value for its rule. These values don't depend on the inputs and can be set manually or tuned during optimization. A heatmap showing how the output varies with the inputs is presented in Fig. 6.

B. GLISp Optimization via User Preference Feedback

To tune the MF parameters of the FIS, the GLISp algorithm [9] was employed. It optimizes the boundary values of the triangular membership functions for the inputs $e_f \in \mathbb{R}$, $\dot{f} \in \mathbb{R}$, and the output gain $K \in \mathbb{R}$. These are denoted by: $M_{ef} \in \mathbb{R}^5$, $M_{df} \in \mathbb{R}^3$, and $M_K \in \mathbb{R}^3$, forming a parameter vector $\mathbf{M} \in \mathbb{M} \subseteq \mathbb{R}^{11}$ within a bounded search space $\mathbb{M}$ (see Fig. 5):

$$\begin{aligned} M_{ef} &= [a_{m1e_{fi}}, a_{m2e_{fi}}, c_{m3e_{fi}}, a_{m4e_{fi}}, a_{m5e_{fi}}] \\ M_{df} &= [a_{m1\dot{f}_i}, a_{m2\dot{f}_i}, c_{m1\dot{f}_i}] \\ M_K &= [a_{m1K_i}, a_{m2K_i}, c_{m1K_i}] \\ \mathbf{M} &= [M_f, M_{df}, M_K] \end{aligned} \tag{16}$$

These parameters control the shape and placement of the triangular MFs, affecting how the controller responds to changes in contact force. GLISp then optimizes parameters through the following steps:

(i) Task Execution by the User: The user completes two rounds of a grasp force control in any experiment using the GUI and the fuzzy controller, each time with a different parameter set $\mathbf{M} \in \mathbb{R}^{11}$.

(ii) User Preference Feedback Collection: After the second round of the experiment with parameter set $\mathbf{M}_s$, the user compares their performance with the first round with parameter set $\mathbf{M}_{s-1}$ based on subjective factors such as thumb force tracking error e_{f_T}, grasp stability, responsiveness, or overall task efficiency, and provide feedback (*better*, *worse*, or *equal*) via the GUI. Each experiment produces one preference sample $[\mathbf{M}_{s-1}, \mathbf{M}_s, \text{label}]$, considering N as the total number

of experiments, the user is required to complete at least three experiments, i.e., $N > 3$.

(iii) Subjective Performance Assessment: User preferences are interpreted as pairwise comparisons. Formally, for two parameter sets $\mathbf{M}_1$ and $\mathbf{M}_2$, the preference function $\pi : \mathbb{R}^{11\times 11} \to \{-1, 0, 1\}$ is defined as:

$$\pi(\mathbf{M}_1, \mathbf{M}_2) = \begin{cases} -1 & \text{if } \mathbf{M}_2 \text{ is "worse" than } \mathbf{M}_1 \\ 0 & \text{if } \mathbf{M}_1 \text{ and } \mathbf{M}_2 \text{ are "equal"} \\ 1 & \text{if } \mathbf{M}_2 \text{ is "better" than } \mathbf{M}_1 \end{cases} \tag{17}$$

Assume we have $N > 3$ samples $\{\mathbf{M}_1, \ldots, \mathbf{M}_s, \ldots, \mathbf{M}_N\}$, where each $\mathbf{M}_s$ is unique; i.e., for $\mathbf{M}_s, \mathbf{M}_r \in \mathbb{R}^{11}$, $\mathbf{M}_s \neq \mathbf{M}_r$ for all $s \neq r$, with $s, r = 1, \ldots, N$. For each pair of samples, an experiment is performed and the user provides a preference. These preferences are collected in a vector $B = [b_1, \ldots, b_E]^T \in \{-1, 0, 1\}^E$, where:

$$b_h = \pi(\mathbf{M}_{s(h)}, \mathbf{M}_{r(h)}) \tag{18}$$

Each index pair $h \in \{1, \ldots, E\}$, with $s(h), r(h) \in \{1, \ldots, N\}$ and $s(h) \neq r(h)$, refers to two different parameter sets, and E is the total number of preferences given. Note that each element b_h reflects the user's judgment on which of the two parameter sets performed better.

(iv) Surrogate Modeling: User preferences are used to build a surrogate function $J : \mathbb{R}^{11} \to \mathbb{R}$, modeled as a weighted sum of radial basis functions (RBFs):

$$\hat{J}(\mathbf{M}) = \sum_{s=1}^{N} \beta_s \, \phi(\lambda d(\mathbf{M}, \mathbf{M}_s)) \tag{19}$$

where, $d : \mathbb{R}^{11\times 11} \to \mathbb{R}$ is the squared Euclidean distance between parameter sets (i.e., $d(\mathbf{M}, \mathbf{M}_s) = \|\mathbf{M} - \mathbf{M}_s\|_2^2$), $\lambda > 0$ is a scaling factor, and $\beta = [\beta_1, \ldots, \beta_N]^T$ are weights learned from user preferences. The RBF $\phi : \mathbb{R} \to \mathbb{R}$ can be, for example: (i) inverse quadratic, $\phi(\lambda) = \frac{1}{1+(\lambda)^2}$, or (ii) Gaussian, $\phi(\lambda) = e^{-(\lambda)^2}$. According to the user's preference labels (see Eq. 17), the surrogate function $\hat{J} \in \mathbb{R}$ must satisfy the following constraints:

$$\begin{aligned} \hat{J}(\mathbf{M}_{s(h)}) &\leq \hat{J}(\mathbf{M}_{r(h)}) - \Sigma + \varepsilon_h, \text{if } \pi(\mathbf{M}_{s(h)}, \mathbf{M}_{r(h)}) = -1 \\ \hat{J}(\mathbf{M}_{s(h)}) &\geq \hat{J}(\mathbf{M}_{r(h)}) + \Sigma - \varepsilon_h, \text{if } \pi(\mathbf{M}_{s(h)}, \mathbf{M}_{r(h)}) = 1 \\ |\hat{J}(\mathbf{M}_{s(h)}) - \hat{J}(\mathbf{M}_{r(h)})| &\leq \Sigma + \varepsilon_h, \text{if } \pi(\mathbf{M}_{s(h)}, \mathbf{M}_{r(h)}) = 0 \end{aligned} \tag{20}$$

where $\Sigma > 0$ is a tolerance value, and ε_h is positive slack variables that allows flexibility in satisfying the preference constraints. Using these constraints, the weight vector β is computed by solving the following Quadratic Programming problem:

$$\begin{aligned} \min_{\beta,\varepsilon} \quad & \sum_{h=1}^{E} \varepsilon_h + \frac{\lambda}{2} \sum_{s=1}^{N} \beta_s^2 \\ \text{s.t.} \quad & \sum_{s=1}^{N} \left(\phi(\lambda d(\mathbf{M}_{s(h)}, \mathbf{M}_s)) - \phi(\lambda d(\mathbf{M}_{r(h)}, \mathbf{M}_s))\right) \beta_s \leq -\Sigma + \varepsilon_h \\ & \forall h : b_h = -1, \\ & \sum_{s=1}^{N} \left(\phi(\lambda d(\mathbf{M}_{s(h)}, \mathbf{M}_s)) - \phi(\lambda d(\mathbf{M}_{r(h)}, \mathbf{M}_s))\right) \beta_s \geq \Sigma - \varepsilon_h \\ & \forall h : b_h = 1, \\ & \left| \sum_{s=1}^{N} \left(\phi(\lambda d(\mathbf{M}_{s(h)}, \mathbf{M}_s)) - \phi(\lambda d(\mathbf{M}_{r(h)}, \mathbf{M}_s))\right) \beta_s \right| \leq \Sigma + \varepsilon_h \\ & \forall h : b_h = 0, \end{aligned} \tag{21}$$

(v) Acquisition Function Optimization: Once a surrogate function $\hat{J}$ is learned, it can be minimized to find the best parameter vector $\mathbf{M} \in \mathbb{R}^{11}$ for the fuzzy-based force controller by following these steps:

(1) Minimize the surrogate function $\hat{J}$:

$$\mathbf{M}_{N+1} = \arg\min \hat{J}(\mathbf{M}) \quad \text{s.t.} \quad \mathbf{M} \in \mathbb{M}; \tag{22}$$

(2) Ask the user to compare $\mathbf{M}_{N+1}$ with the best parameter found so far, $\mathbf{M}_N^*$, and provide a preference.

(3) Update the surrogate $\hat{J}$ using the new preference data via Eq. 21.

(4) Iterate over N.

To choose the next sample $\mathbf{M}_{N+1}$, GLISp uses an acquisition function that balances exploration and exploitation. This function includes an exploration term based on inverse distance weighting (IDW), defined by the function $z : \mathbb{R}^{11} \to \mathbb{R}$ as:

$$z(\mathbf{M}) = \begin{cases} 0 & \text{if } \mathbf{M} = \mathbf{M}_s \text{ for some } s, \\ \arctan\left(\frac{1}{\sum_{s=1}^{N} w_s(\mathbf{M})}\right) & \text{otherwise} \end{cases} \tag{23}$$

Here, $w_s(\mathbf{M}) = d(\mathbf{M}, \mathbf{M}_s)^{-1}$, and the arc tangent function prevents $z(\mathbf{M})$ from becoming excessively large when $\mathbf{M}$ is far from all sampled points. Then, using an exploration parameter $\delta \geq 0$, the acquisition function $a : \mathbb{R}^{11} \to \mathbb{R}$ is defined as:

$$a(\mathbf{M}) = \frac{\hat{J}(\mathbf{M})}{\Delta\hat{J}} - \delta z(\mathbf{M}), \tag{24}$$

where

$$\Delta\hat{J} = \max_s\{\hat{J}(\mathbf{M}_s)\} - \min_s\{\hat{J}(\mathbf{M}_s)\}. \tag{25}$$

The range of the surrogate function over the current samples $\{\mathbf{M}_1, \ldots, \mathbf{M}_N\}$ is used to normalize the acquisition function and simplify the selection of the exploration parameter δ. Given this sample set and the preference vector B (see Eq. 18), the next parameter $\mathbf{M}_{N+1}$ of the force controller is computed as the solution of the (non-convex) optimization problem:

$$\mathbf{M}_{N+1} = \arg\min_{\mathbf{M}\in\mathbb{M}} a(\mathbf{M}). \tag{26}$$

(vi) Iterative Refinement: The process is repeated for a number of iterations (e.g., 20), allowing the algorithm to gradually find parameter sets that best match the user's preferences.

(vii) Parameter Evolution: This stage highlights how key membership function parameters evolve over iterations. By continuously incorporating user preferences, the optimization refines the fuzzy controller to improve performance.

IV. Neural Network-Based Force Controller

The neural network-based force controller learns directly from data, enabling it to generalize across diverse tasks and adapt to unmodeled nonlinearities and uncertainties in real-world interactions. To generalize gain estimation and reduce manual tuning, a Deep Neural Network (DNN) is implemented in the force controller to learn the mapping from sensory inputs to gain values for the thumb, index, and middle fingers. The training data is generated by running grasp regulation simulations using the fuzzy-based controller with GLISp optimization, as described earlier. The dataset includes n_s samples, represented as $\mathbf{T}_T = D[\mathbf{X}_T, \mathbf{Y}_T] \in \mathbb{R}^{n_s\times 7}$, where:

- $\mathbf{X}_T = D[F_{\text{ref}}, f_T, f_I, f_M] \in \mathbb{R}^{n_s\times 4}$ is the input matrix, containing the force reference signal $F_{\text{ref}} \in \mathbb{R}$, which the user is expected to follow during the experiment, along with the measured contact forces from the thumb (f_T), index (f_I), and middle (f_M) fingers.
- $\mathbf{Y}_T = D[K_T, K_I, K_M] \in \mathbb{R}^{n_s\times 3}$ is the output matrix, containing the fuzzy-based gain values assigned to each finger (see Eq. 15).

The dataset is split into 80% for training and 20% for testing using a fixed random seed for reproducibility. All input and output features are standardized to zero mean and unit variance to accelerate training convergence. During each training iteration τ, the DNN processes a mini-batch of B samples $\mathbf{T}_B \subset \mathbf{T}_T$, where $b = 1, \ldots, B$. The DNN parameters are updated after processing each mini-batch until the entire dataset $\mathbf{T}_T$ is covered, constituting one epoch. This process is repeated over 200 epochs. The DNN architecture (see Fig. 7) consists of a four-layer fully connected network: an input layer, two hidden layers, and an output layer. Each hidden layer $l \in \{1, 2\}$ contains 64 neurons and is fully connected to the preceding layer, forming a dense structure. The training process involves four key steps:

(i) Forward Pass: The network generates predictions based on the input features using pre-activation vectors ($\mathbf{z}_l$) and activation functions ($\mathbf{a}_l$). For the first hidden layer ($l = 1$):

$$\mathbf{z}_1 = \mathbf{W}_1\mathbf{X}_B + \mathbf{b}_1 \tag{27}$$

where $\mathbf{W}_1 \in \mathbb{R}^{64\times 4}$ and $\mathbf{b}_1 \in \mathbb{R}^{64}$ are the weights and biases. For the second hidden layer ($l = 2$):

$$\mathbf{z}_2 = \mathbf{W}_2\mathbf{a}_1 + \mathbf{b}_2 \tag{28}$$

TABLE I: DNN specification for the NN-based force controller.

Parameter	Value	Parameter	Value
Input Features	4	Output Features	3
Hidden Layers	2	Neurons/Layer	64
Connectivity	Dense	Activation	ReLU
Regularization (λ)	None	ϵ	10^{-8}
Loss Function	MSE	Optimizer	Adam
Learning Rate	0.001	Batch Size	32
Epochs	200	Output Layer	Dense

where, $\mathbf{W}_2 \in \mathbb{R}^{64\times 64}$, $\mathbf{b}_2 \in \mathbb{R}^{64}$ and $\mathbf{a}_l = \text{ReLU}(\mathbf{z}_l)$. The output layer computes a 3-dimensional vector representing the predicted gains ($\hat{\mathbf{Y}}_B$) for the mini-batch:

$$\hat{\mathbf{Y}}_B = \mathbf{W}_{\text{out}}\mathbf{a}_2 + \mathbf{b}_{\text{out}} \tag{29}$$

where $\mathbf{W}_{\text{out}} \in \mathbb{R}^{3\times 64}$ and $\mathbf{b}_{\text{out}} \in \mathbb{R}^3$ are the weights and biases of the output layer.

(ii) Error Calculation: After the forward pass, the difference between the predicted gains $\hat{\mathbf{Y}}_B$ and the true gains $\mathbf{Y}_B$ in the mini-batch $\mathbf{T}_B$ is computed using the Mean Squared Error (MSE) loss function:

$$L = \frac{1}{B}\sum_{b=1}^{B} \|\mathbf{Y}_{B_b} - \hat{\mathbf{Y}}_{B_b}\|^2 \tag{30}$$

where B is the number of samples in the mini-batch.

(iii) Backpropagation: Backpropagation calculates how the loss L changes with respect to the network's weights and biases ($\mathbf{W}_l, \mathbf{b}_l$) using the chain rule. It starts from the output layer and moves backward through the hidden layers. In this work, the gradients are computed automatically using the PyTorch framework.

(iv) Parameter Update: The gradients are used to update the network parameters using the Adam optimizer, which adapts the learning rate based on estimates of the first and second moments of the gradients. The update rules for the weights $\mathbf{W}$ (and similarly for biases $\mathbf{b}$) are:

$$\begin{aligned}
\mathbf{m}_\tau &= \beta_1\mathbf{m}_{\tau-1} + (1-\beta_1)\nabla_{\mathbf{W}}L \\
\mathbf{v}_\tau &= \beta_2\mathbf{v}_{\tau-1} + (1-\beta_2)(\nabla_{\mathbf{W}}L)^2 \\
\hat{\mathbf{m}}_\tau &= \frac{\mathbf{m}_\tau}{1-\beta_1^\tau} \\
\hat{\mathbf{v}}_\tau &= \frac{\mathbf{v}_\tau}{1-\beta_2^\tau} \\
\mathbf{W}_{\tau+1} &= \mathbf{W}_\tau - \frac{\eta}{\sqrt{\hat{\mathbf{v}}_\tau}+\epsilon}\hat{\mathbf{m}}_\tau
\end{aligned} \tag{31}$$

where $\mathbf{m}_\tau$ and $\mathbf{v}_\tau$ are the first and second moment estimates, $\beta_1 = 0.9$ and $\beta_2 = 0.999$ are the exponential decay rates, $\eta = 10^{-3}$ is the learning rate, and $\epsilon = 10^{-8}$ is a small constant to prevent division by zero. After training, the model is evaluated on the test set using the MSE loss to assess generalization performance. Table I summarizes the specifications of the DNN architecture, and the trained model's weights and scalers are saved for future use.

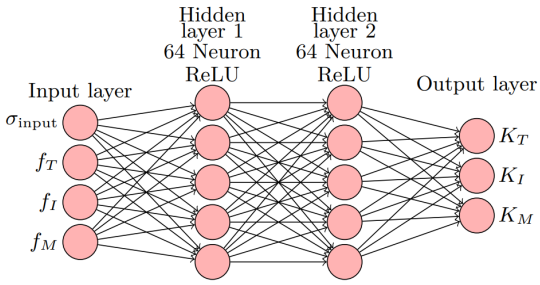

Fig. 7: Deep neural network structure embedded in the NN-based force controller.

V. Experimental Validation

A. Implementation and Description of the Experiment

The video of the experimental trials is available as supplementary material at: https://www.youtube.com/watch?v=20Gyn_JhGXY (accessed: June 4, 2025). The system was developed in Python 3.13.3 and ran on a PC with Ubuntu 20.04, an Intel Core i5 processor, 16 GB RAM, and an NVIDIA RTX 3060 GPU. ROS Noetic was used for communication and control. The AR10 hand was connected via serial communication, and the control loop ran at 50 Hz, enabling real-time myocontrol and responsive force regulation. It was tested in a force regulation task using tripod grasps. Two right-handed participants (ages 25 and 35), with no prior experience in myoelectric control, took part in the study. Each participant sat at a table and wore the sEMG armband and vibrotactile device on their dominant hand. The robotic hand was placed in front of them, and the test object was randomly selected and grasped as shown in Fig. 8. Three different objects were used to introduce variation in shape and contact conditions: a cube (5 cm edge), a cylinder (5 cm diameter, 6 cm height), and a sphere (6 cm diameter), all made of hard plastic (Fig. 9). The experiment consisted of three phases:

Phase 1: sEMG Control Calibration. As described in Section II-B, the user performed two hand openings and closings to calibrate the NMF-based model and set the scaling parameters (c_{scale}, c_{offset}).

Phase 2: User Preferences Collection. As shown in the GUI in Fig. 10 (left), the process begins by the user clicking on *Start Grasp*, allowing time to become familiar with the system and complete a stable grasp. Next, the user presses *First Round* and follows a reference force for 30 s using the current fuzzy parameters, then proceeds with *Second Round*, performing the same task but with a different set of fuzzy parameters. The user then compares both rounds' tracking errors and overall operation, and provides preference feedback by selecting whether the second round was *Better*, *Equal*, or *Worse*. Each experiment consists of two rounds, and users are expected to complete at least three full experiments. All collected data are saved and used for optimizing the fuzzy controller through the GLISp framework.

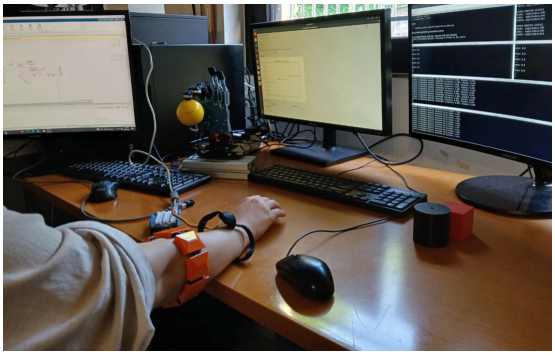

Fig. 8: Experimental setup for grasp strength regulation.

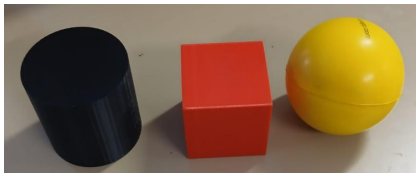

Fig. 9: Objects used in the Experiment

Phase 3: Experiment Execution. As shown in the GUI in Fig. 10 (right), after selecting one of the available controllers rule-based, fuzzy-based, or neural network-based, the user clicks *Start Grasp* to initiate and stabilize the object grasp. Next, they press *Start Experiment* to begin the test, during which the user attempts to modulate the grasp force to match predefined target levels $F_{ref} = \{0.8, 1, 1.2\}$ N. The reference and actual force signals are displayed in real time, allowing the user to visually monitor performance. These target forces were chosen to simulate light, medium, and firm grasp intensities within the comfortable control range of the robotic hand. During the experiment, if the tracking error exceeds ± 0.03 N, the vibrotactile motor activates to alert the user. At the same time, on the GUI, the *"Close a bit"* or *"Open a bit"* buttons start blinking when the applied force is below or above the reference value, respectively, helping the user adjust their muscle activation accordingly. Each subject completed all combinations of object and force controller type.

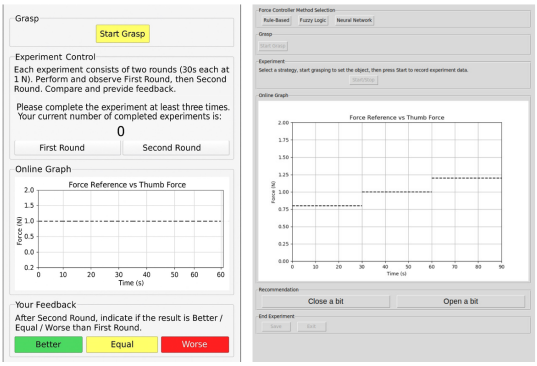

Fig. 10: GUI for collecting user preferences applied in fuzzy force controller (left) and experiment execution (right).

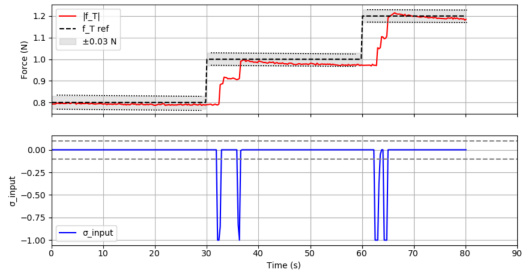

Fig. 11: Force tracking and user input using the heuristic model-based controller.

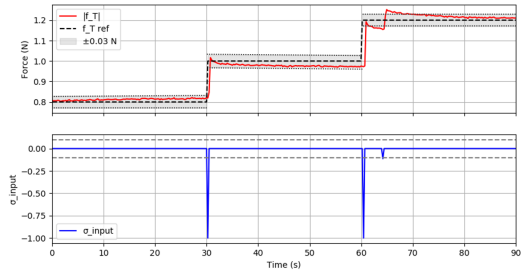

Fig. 12: Force tracking and user input using the fuzzy-based controller.

B. Experimental Results

The performance of the proposed control architecture was evaluated across 10 trials. The analysis focused on the system's ability to accurately modulate contact forces, respond to user intent, and dynamically adapt control gains through different controllers: heuristic model-based, fuzzy-based, and neural network-based. As a sample of the system's performance, Figures 11, 12, and 13 illustrate representative trials for each controller, performed by one participant using the sphere object, showing the measured thumb contact force $|f_T|$, the target force F_{ref}, and the user input signal σ_{input}. The shaded region around the reference force indicates the $\pm 0.03\,\text{N}$ error tolerance band used to trigger vibrotactile feedback. The heuristic model-based controller exhibited higher overshoot during transitions to higher force levels. The fuzzy-based controller reduced overshoot, but still showed moderate fluctuations around the target force and occasional high-frequency corrections. In contrast, the neural network-based controller achieved the smoothest and most stable response across all force levels. The peaks are caused by a combination of user input variability and the force controller's ability to adjust the gain effectively to prevent overshoot and minimize tracking error. The vibrotactile feedback alerts the user when abnormal force deviations occur and prompts them to take appropriate corrective actions, such as slightly opening or closing the hand. The Root Mean Square Error (RMSE) between the measured thumb force $f_T(t)$ and the reference force $F_{\text{ref}}(t)$ over a time window of n samples is presented in Table II.

VI. Conclusions and Future Works

We presented a fuzzy and neural network-based force control system for myocontrolled hands using vibrotactile feedback. The fuzzy controller was refined via user preferences using GLISp, while the NN model learned optimal gains from data. Experiments showed the NN approach yielded the lowest tracking error. Although this study was conducted under controlled conditions, the adaptive fuzzy controller and the DNN's generalization help ensure robustness to moderate noise and input variability. Future work includes extending to multi-finger tasks and broader evaluation across users and grasp types.

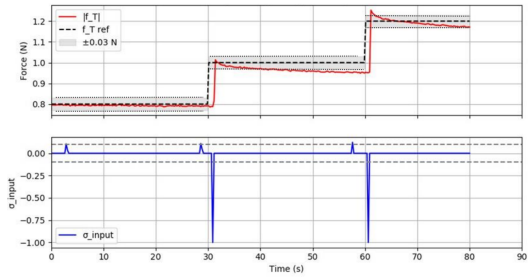

Fig. 13: Force tracking and user input using the NN-based controller.

TABLE II: RMSE of force tracking error for each controller.

Controller	Model-Based	Fuzzy-based	Neural Network-Based
RMSE (N)	0.0525	0.040	0.025

Acknowledgment

This work was partially supported by:

- Project PID2020-114819GB-I00, funded by MICIU/AEI/10.13039/501100011033.
- European Commission's Horizon Europe Programme: IntelliMan (Grant 101070136); MUR project "Sustainable Mobility Center" (Grant CN00000023-CUP J33C22001120001); and the MICS Extended Partnership funded by the EU Next-GenerationEU (PNRR – Mission 4, Component 2, Investment 1.3 – D.D. 1551.11-10-2022, PE00000004).

References

[1] C. L. McDonald, S. Westcott-McCoy, M. R. Weaver, J. Haagsma, and D. Kartin, "Global prevalence of traumatic non-fatal limb amputation," *Prosthetics and orthotics international*, vol. 45, no. 2, pp. 105–114, 2021.

[2] A. Montaño and R. Suárez, "Dexterous manipulation of unknown objects using virtual contact points," *Robotics*, vol. 8, no. 4, p. 86, 2019.

[3] M. Sheikhsamad, R. Suárez, and J. Rosell, "Learning-based planner for unknown object dexterous manipulation using anfis," *Machines*, vol. 12, no. 6, p. 364, 2024.

[4] R. Meattini, S. Benatti, U. Scarcia, D. De Gregorio, L. Benini, and C. Melchiorri, "An semg-based human–robot interface for robotic hands using machine learning and synergies," *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 8, no. 7, pp. 1149–1158, 2018.

[5] R. Meattini, L. Biagiotti, G. Palli, D. De Gregorio, and C. Melchiorri, "A control architecture for grasp strength regulation in myocontrolled robotic hands using vibrotactile feedback: Preliminary results," *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR)*, pp. 1272–1277, 2019.

[6] R. Meattini, S. Benatti, U. Scarcia, L. Benini, and C. Melchiorri, "Experimental evaluation of a semg-based human-robot interface for human-like grasping tasks," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2015, pp. 1030–1035.

[7] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[8] M. Sugeno, *Fuzzy modeling and control: selected works of M. Sugeno*. cRc Press, 1999.

[9] L. Roveda, B. Maggioni, E. Marescotti, A. A. Shahid, A. M. Zanchettin, A. Bemporad, and D. Piga, "Pairwise preferences-based optimization of a path-based velocity planner in robotic sealing tasks," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 6632–6639, 2021.