A comparison between a two-feedback control loop and a reinforcement learning algorithm for compliant low-cost series elastic actuators

Filippo Sanfilippo
Dept. of Engineering Sciences
University of Agder (UiA)
filippo.sanfilippo@uia.no

Tuan Minh Hua
Dept. of Science and Industry Systems
University of South-Eastern Norway (USN)
huaminhtuan94@gmail.com

Steven Bos
Dept. of Science and Industry Systems
University of South-Eastern Norway (USN)
steven.bos@usn.no

Abstract

Highly-compliant elastic actuators have become progressively prominent over the last years for a variety of robotic applications. With remarkable shock tolerance, elastic actuators are appropriate for robots operating in unstructured environments. In accordance with this trend, a novel elastic actuator was recently designed by our research group for Serpens, a low-cost, open-source and highly-compliant multi-purpose modular snake robot. To control the newly designed elastic actuators of Serpens, a two-feedback loops position control algorithm was proposed. The inner controller loop is implemented as a model reference adaptive controller (MRAC), while the outer control loop adopts a fuzzy proportional-integral controller (FPIC). The performance of the presented control scheme was demonstrated through simulations. However, the efficiency of the proposed controller is dependent on the initial values of the parameters of the MRAC controller as well as on the effort required for a human to manually construct fuzzy rules. An alternative solution to the problem might consist of using methods that do not assume a priori knowledge: a solution that derives its properties from a machine learning procedure. In this way, the controller would be able to automatically learn the properties of the elastic actuator to be controlled. In this work, a novel controller for the proposed elastic actuator is presented based on the use of an artificial neural network (ANN) that is trained with reinforcement learning. The newly designed control algorithm is extensively compared with the former approach. Simulation results are presented for both methods. The authors seek to achieve a fair, non-biased, risk-aware and trustworthy comparison.

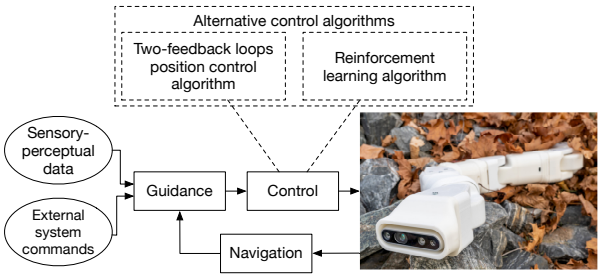

Figure 1. The underlying idea of developing a novel low-level controller based on the use of an artificial neural network (ANN) that is trained with reinforcement learning (RL). The newly proposed controller is extensively compared with a two-feedback loops position control algorithm that was previously developed by our research group.

1. Introduction

Biological limbless organisms like snakes are capable of achieving locomotion by exploiting rocks, stones, branches, obstacles, or other irregularities in the terrain as a means of propulsion [1]. This unique capability enables biological snakes to be extremely adaptable to various types of environments. Snake robots that can mimic this range of behaviour could enable a variety of possible applications for use in challenging real-life operations and hazardous or confined areas that conventional robots (i.e. wheeled, tracked and legged) and humans are unable to access, such as explorations of earthquake-hit areas, pipe inspections for the oil and gas industry, fire-fighting operations, and search-and-rescue activities (SAR) [2]. Snake robot locomotion in a cluttered environment where the snake robot utilises a sensory-perceptual system to exploit the surrounding operational space and identifies walls, obstacles, or other external objects for means of propulsion can be defined as *perception-driven obstacle-aided locomotion* (POAL) [3, 4]. The development of POAL is known to be challenging because of the complex interaction between the snake robot and the adjacent cluttered environment. From a control point of view, achieving POAL requires to precisely identify potential push-points and to accurately determine achievable contact reaction forces.

Accomplishing this with traditional rigidly-actuated robots is extremely demanding because of the absence of compliance. To ease the control complexity, compliant motion and fine torque control is desirable. To achieve this, *Serpens*, a newly-designed low-cost, open-source and highly-compliant multi-purpose modular snake robot with series elastic actuator (SEA), was introduced by our research group [5, 6]. To control the newly designed elastic actuators of *Serpens*, a two-feedback loops position control algorithm was proposed [7]. The proposed controller has two loops: the inner loop is implemented as a model reference adaptive controller (MRAC), while the outer loop adopts a fuzzy proportional-integral controller (FPIC). The advantage of combining the FPIC and the MRAC controllers is the possibility of achieving independence with respect to imprecise system parameters. Experimental simulation highlighted the effectiveness of the proposed algorithm with respect to the influence of external torque on the considered elastic actuator. Nonetheless, the performance of the proposed controller is dependent on the initial values of the parameters of the MRAC controller as well as on the effort required for a human expert to manually construct fuzzy rules. Specifically, the challenge is that to synthesise an efficient control law for the system, fuzzy rules must be provided by an expert, which state the action(s) to do in typical situations. The key to achieve an efficient controller requires to master the semantics of the fuzzy rules. If this is lacking, a learning methodology may be more appropriate [8]. Particularly, an alternative solution to the problem might consist of using methods that do not assume a priori knowledge: a solution that derives its properties from a machine learning procedure. Accordingly, the controller would be able to automatically learn the properties of the elastic actuator to be controlled. The main contribution of this work is a novel controller for the proposed elastic actuator of *Serpens*. The novel controller is based on the use of an artificial neural network (ANN) that is trained with reinforcement learning (RL). The newly designed control algorithm is extensively compared with the former approach based on the two-feedback loops position control. The authors seek to achieve a fair, non-biased, risk-aware and trustworthy comparison. This will make it possible to explore a new way of looking at the considered control problem. The underlying idea is shown in Fig. 1. The comparison is performed by using the same model for the elastic actuator. Only the controller is switched between the two methods to be compared. This approach guarantees identical conditions for the comparison. The contributions of this article are the newly designed control algorithm and the comparison with the former method.

The paper is organised as follows. A review of the related research work is described in Section 2. The proposed mathematical model of the elastic actuator is summarised in Section 3. The two considered control methods are presented in Section 4. In Section 5, related simulation results are outlined. Finally, conclusions and future work are discussed in Section 6.

2. Related research works

Literature shows some examples of creating controllers for elastic actuators by employing fuzzy rules. For instance, a fuzzy position/force control was implemented for a robot leg with a flexible gear system [9, 10]. The fuzzy position/force control approach and an intelligent walking strategy are implemented into the robot leg to realise walking on unknown and uneven terrain. By following a similar approach, a two-feedback loops position control algorithm was proposed by our research group [7]. The inner controller loop is implemented as an MRAC to cope with uncertainties in the system parameters, while the outer control loop adopts an FPIC to reduce the effect of external disturbances on the load. The advantage of combining the FPIC and the MRAC controllers is the possibility of achieving independence with respect to imprecise system parameters. Simulation results were used to prove the efficiency of the presented control method. However, one of the main limitations of fuzzy based controllers is that rules and criterion must be understandable by humans. Mostly these rules must be defined by a domain expert. Essentially the fuzzy logic requires a lot of human intervention [11]. An alternative approach to this challenge might be based on the use of methods that do not assume a priori knowledge. For instance, a viable approach might consist in searching for a solution that derives its properties from a machine learning procedure. In this way, the controller would be able to automatically learn the properties of the elastic actuator to be controlled. A few examples can be found in the literature. For instance, the application of RL to improve the performance of highly dynamic single legged locomotion with compliant series elastic actuators was presented in [12]. Another example is reported in [13], where a deep-learning approach is adopted to learn the elasticity of a series elastic actuator for accurate torque control. Regarding other elastic actuators, an adaptive control and an adaptive ANN control for tendon-driven robotic mechanisms with elastic tendons was presented in [14]. One of the main

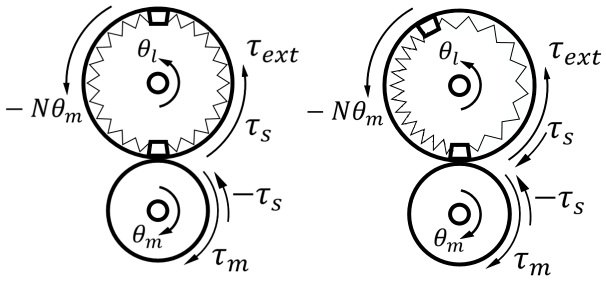

Figure 2. Elastic actuator model: (a) without external force/ torque; (b) Compressed/tensed by external action. The gear ratio is $N = N_l/N_m$.

advantages of this approach is that it can be applied to serial or parallel tendon-driven manipulators having linear or non-linear elastic tendons. However, research about controlling elastic actuators with machine learning procedures is still relatively limited, especially when considering snake robots. The main contribution of this paper is to propose such a method and compare its performance with the former controller implemented by our research group [7].

3. Mathematical model

This mathematical model of the proposed elastic actuator is briefly summarised in this section. For further details, the reader is referred to [7]. The schematics of *Serpens*' elastic actuator [5, 6] are illustrated in Fig. 2. There are two gears, the load and the motor gears, whose numbers of teeth are N_l and N_m, respectively. The gear ratio is $N = N_l/N_m$. The torques depicted in the diagrams are: motor torque (τ_m), spring reaction torque (τ_s) and external torque (τ_{ext}). The system with no external force/torque is illustrated in Fig. 2-a, while the system influences by an external action is illustrated in Fig. 2-b.

The mathematical model of the elastic actuator system is:

$$d_m + \tau_m - N^{-1}\tau_s = J_m\ddot{\theta}_m + D_m\dot{\theta}_m, \quad (1)$$

$$\tau_s = K_s(-N^{-1}\theta_m - \theta_l) + D_s(-N^{-1}\dot{\theta}_m - \dot{\theta}_l), \quad (2)$$

$$d_l + \tau_s + \tau_{ext} = J_l\ddot{\theta}_l + D_l\dot{\theta}_l, \quad (3)$$

in which external disturbances affecting the motor and the load behaviour are d_m and d_l, respectively, θ_m is the motor angular position, θ_l is the load angular position, J_m is the rotor inertia, D_m is the motor damping coefficient, K_s is the stiffness coefficient of the spring, D_s is the spring damping coefficient, J_l is the load inertia, and D_l is the load damping coefficient.

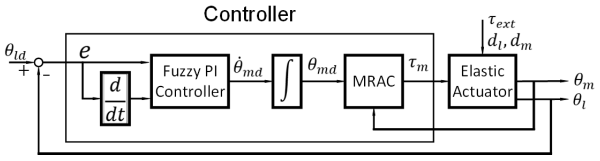

Figure 3. The proposed nested system controller.

Equation 1 shows the relationship on the motor-side between the motor torque, the spring torque, and the motor angular position. The spring torque τ_s is obtained by equation 2. The interaction between the spring torque, the external torque, and the load angular position is illustrated by equation 3.

4. Control methods

In this section, the two control methods to be compared are described. The first method is the previously presented two-feedback loops position control algorithm. For further details, the reader is referred to [7]. The second method is a novel controller based on the use of RL.

4.1. Two-feedback loops position control algorithm

For controlling the load position when considering external disturbances on the load and uncertainties in system parameters, a two-loop controller is proposed. An FPIC is applied for the load-side to reduce the effect of external disturbances on the load. The output of the FPIC is used as the desired angular position of the motor. An MRAC is used for the motor-side to deal with uncertainties in system parameters. The proposed control algorithm diagram is presented in Fig. 3.

4.1.1. Fuzzy PI Controller The effect of the external disturbances on the elastic actuator system is reduced by adopting the FPIC in the load-side. Inputs for the FPIC are the feedback error (e) and the change of this error ($\dot{e}$), while the output of the FPIC is the change of control signal ($\dot{u}$). The membership functions for the input and the output are shown in Fig. 4 and Fig. 5, in which $c1 - c5$ are parameters to be adjusted. The membership functions for the error and the change of error are similar, including the rules base of the fuzzy model: negative big (NB), negative small (NS), zero (ZE), positive small (PS), and positive big (PB). The membership functions for the change of the control signal are in singleton over the output, given NB, NS, ZE, PS, and PB. On the basis of the input and output membership functions, 25 fuzzy inference rules are

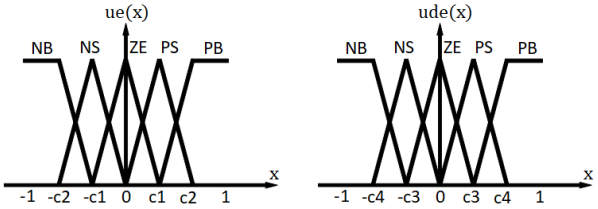

Figure 4. Input membership functions [10].

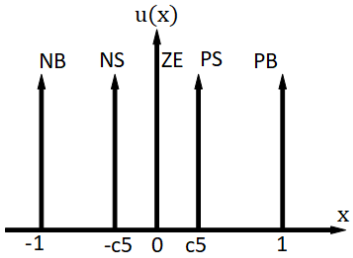

Figure 5. Output membership functions.

established as shown in Table 1.

4.1.2. Model Reference Adaptive Controller Firstly, the control law is derived. From the system equations 1 and 2, the motor-side system equations can be rewritten (ignoring the motor-side external disturbance d_m) as:

$$\dot{\theta} = \begin{bmatrix} \dot{\theta}_m \\ \ddot{\theta}_m \end{bmatrix} = A\theta + B(\tau_m + K_s N^{-1}\theta_l + D_s N^{-1}\dot{\theta}_l), \quad (4)$$

in which the state matrix A, input matrix B and state vector θ can be obtained as:

$$A = \begin{bmatrix} 0 & 1 \\ \frac{K_s N^{-2}}{J_m} & \frac{D_s N^{-2} - D_m}{J_m} \end{bmatrix}, \; B = \begin{bmatrix} 0 \\ \frac{1}{J_m} \end{bmatrix}, \; \theta = \begin{bmatrix} \theta_m \\ \dot{\theta}_m \end{bmatrix}. \quad (5)$$

The motor-side system equations have the form of a second-order system, so the reference model is a second-order system model with the desired signal θ_{md},

Table 1. Fuzzy inference rules.

$\dot{e}$ \ e	NB	NS	ZE	PS	PB
NB	PB	PB	PB	PS	ZE
NS	PB	PB	PS	ZE	NS
ZE	PB	PS	ZE	NS	NB
PS	PS	ZE	NS	NB	NB
PB	ZE	NS	NB	NB	NB

natural frequency ω_n, and damping coefficient ξ:

$$\dot{\theta}_{ref} = \begin{bmatrix} \dot{\theta}_r \\ \ddot{\theta}_r \end{bmatrix} = A_r\theta_{ref} + B_r\theta_{md}, \quad (6)$$

in which the state matrix A_r, input matrix B_r and state vector θ_{ref} of the reference model can be obtained as:

$$A_r = \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\xi\omega_n \end{bmatrix}, \; B_r = \begin{bmatrix} 0 \\ \omega_n^2 \end{bmatrix}, \; \theta_{ref} = \begin{bmatrix} \theta_r \\ \dot{\theta}_r \end{bmatrix}. \quad (7)$$

A general control law for the system with state equations 4 is:

$$\tau_m = M\theta_{md} - L\theta - \hat{K}_s N^{-1}\theta_l - \hat{D}_s N^{-1}\dot{\theta}_l. \quad (8)$$

$\hat{D}_m$, $\hat{J}_m$, $\hat{K}_s$, $\hat{D}_s$, $\hat{D}_l$, $\hat{J}_l$ are estimated parameters, while matrices M and L need to be determined.

Secondly, the error equation is determined. The feedback error, which is the difference between the output of the real system and the output of the reference model, is determined as:

$$e = \theta - \theta_{ref}. \quad (9)$$

The derivative of error is determined as:

$$\dot{e} = \dot{\theta} - \dot{\theta}_{ref} = A_r e + \Psi(\Phi - \Phi^*), \quad (10)$$

where it should be noted that:

$$\Psi = B\left[-\theta^T \;\; \theta_{md} \;\; (-N^{-1}\theta_l) \;\; (-N^{-1}\dot{\theta}_l)\right],$$
$$\Phi = \begin{bmatrix} L^T \\ M \\ \hat{K}_s \\ \hat{D}_s \end{bmatrix}, \; \Phi^* = \begin{bmatrix} L^{*T} \\ M^* \\ K_s \\ D_s \end{bmatrix}. \quad (11)$$

Thirdly, the adaptation law is going to be obtained. To achieve robustness, Lyapunov's stability theory is applied in this case. The Lyapunov function is introduced as:

$$V = \frac{1}{2}\gamma e^T P e + \frac{1}{2}(\Phi - \Phi^*)^T(\Phi - \Phi^*), \quad (12)$$

where P is a 2-by-2 symmetric positive definite matrix and γ is a learning rate. The function V is positive definite. The derivative of V is obtained as:

$$\begin{aligned} \dot{V} &= \frac{1}{2}\gamma\dot{e}^T P e + \frac{1}{2}\gamma e^T P\dot{e} + (\Phi - \Phi^*)^T\dot{\Phi} \\ &= -\frac{1}{2}\gamma e^T Q e + (\Phi - \Phi^*)^T(\dot{\Phi} + \gamma\Psi^T P e) \end{aligned}, \quad (13)$$

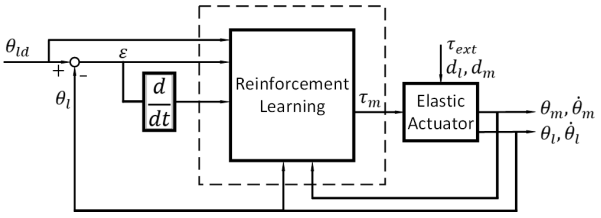

Figure 6. The proposed reinforcement learning system controller.

where $A_r^T P + P A_r = -Q$ and Q is a symmetric positive definite matrix. The existence of matrix Q is demonstrated by using the Kalman-Yakubovich lemma [15]. If the adaptation law is chosen to be:

$$\dot{\Phi} = -\gamma \Psi^T P e, \tag{14}$$

then the derivative of Lyapunov function $\dot{V}$ is negative definite with all $e \neq 0$, which means that the feedback error between the output of the real system and the reference model will converge to zero when time goes to infinity.

In this article, matrices P and Q are chosen as:

$$P = \begin{bmatrix} p_1 & p_2 \\ p_2 & p_3 \end{bmatrix}, \; Q = \begin{bmatrix} q_1 & 0 \\ 0 & q_2 \end{bmatrix}, \tag{15}$$

where q_1 and q_2 will be tuned appropriately. From equation $A_r^T P + P A_r = -Q$, p_1, p_2 and p_3 can be obtained by using the following formulas:

$$p_2 = \frac{q_1}{2\omega_n^2}, p_3 = \frac{2p_2 + q_2}{4\xi\omega_n}, p_1 = 2\xi\omega_n p_2 + \omega_n^2 p_3. \tag{16}$$

4.2. Controller based on the use of an ANN trained with reinforcement learning (RL)

The proposed control scheme is shown in Fig. 6. The input signals are the motor angular position θ_m, the motor angular velocity, $\dot{\theta}_m$, the load angular position, θ_l, the load angular velocity, $\dot{\theta}_l$, the load desired angular position, θ_{ld}, the error between the load desired position and the load actual position, $\varepsilon = \theta_{ld} - \theta_l$, and the error derivative $\dot{\varepsilon}$. The output signal is the motor torque τ_m.

The goal of RL is to design a policy π that maps states to probabilities of selecting each possible action of the controller, and improving it so that π is optimal (i.e. maximises the expected reward) [16]. Policies can be either stochastic $\pi(a|s)$, which given a state s, each action $a \in A(s)$ has an associated probability distribution, or deterministic $\pi(s)$, which directly maps a state s to a determined action a. The steps of a RL algorithm can be summarised as shown in Fig. 7.

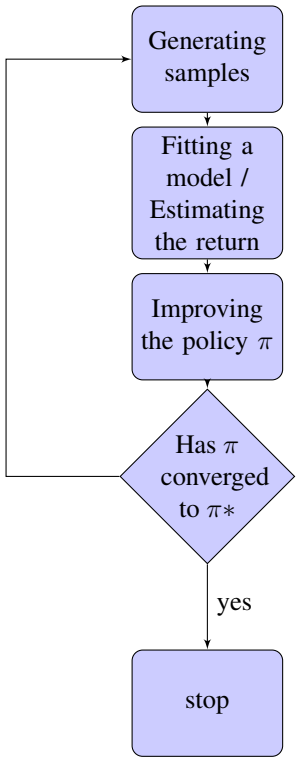

Figure 7. The steps of a RL algorithm [16].

The different RL algorithms available in the literature follow these steps. Among all available classes of algorithms, the class of policy gradient algorithms is the one used in this work. In particular, an alternative controller is developed based on the RL technique called Proximal Policy Optimisation (PPO). For further details, the reader is referred to [17, 16]. In particular, PPO adopts an ANN to approximate the ideal function that maps an agent's observations to the best action an agent can take in a given state.

The adopted ANN topology is shown in Fig. 8. It is a four-layer ANN which consists of one input, one output and two hidden layers. Each neuron in the network is fully connected with every other neuron of the next layer. Adaptive weights are associated to the neuron's interconnections. A Swish self-gated activation function is used [18].

The remaining configuration parameters of the PPO algorithm are detailed in Table 2.

The following rules are considered for the reinforcement learning controller:

- the generated torque τ_m must be within τ_{mmin} and τ_{mmax} to prevent damage;
- the resulting θ_m must be within θ_{mmin} deg and θ_{mmax} to prevent damage;

Table 2. Parameters used in the PPO algortithm.

Parameter	Value
Episode length	50000 ts
Batch size	64 eps
Optimisation steps	1.0e5
Discount (γ)	0.995
Learning rate	3.0e-4

- the motor angular angular velocity, $\dot{\theta}_m$, must reach stable state (reward exponentially);
- the feedback error, ϵ, must be minimised over time;
- the time to reach a stable state must be as fast as possible, giving exponential penalty for time.

Based on these rules, the proposed reward function is given by:

$$r_t = e^{-\alpha|\epsilon|} + e^{-\alpha|\dot{\epsilon}|} + e^{-\alpha|\dot{\theta}_m|}, \tag{17}$$

where α is a scaling reward factor. In this work, this scaling factor is empirically set to 0.001.

5. Simulation results

In this section, the two presented control methods are considered for an extensive comparison. This comparison is performed in simulation. The *Unity* [19] real-time development platform is chosen to implement the model and the controllers to be compared. To implement the controller based on the neural network trained with reinforcement learning, the *Unity ML-Agents Toolkit* [20] is adopted. This toolkit is an open-source *Unity* plugin that enables simulations to run as environments for training intelligent agents. In this case study, agents are trained using reinforcement learning through a simple-to-use *Python* Application Programming Interface (API). During training, a block

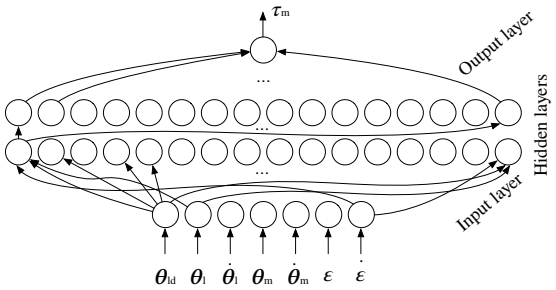

Figure 8. The adopted neural network topology.

Table 3. System parameters.

Parameter	Value	Parameter	Value
Gear ratio(N)	1	Spring stiffness (K_s)	100
Load damping coefficient(D_l)	0.006	Motor damping coefficient(D_m)	0.6
Load inertia (J_l)	0.065	Motor inertia (J_m)	0.1
Spring damping coefficient(D_s)	5	Sampling rate (T)	0.0001

of agent experiences is generated. These experiences become the training set for a neural network used to optimise the agent's policy. To apply reinforcement learning, the neural network optimises the policy by maximising the expected rewards. The output of the training process is a model file containing the optimised policy. This model file is a *TensorFlow* [21] data graph containing the mathematical operations and the optimised weights selected during the training process. The considered elastic actuator is simulated in the *Unity* environment, as shown in Fig. 9. The simulation environment also shows the plots related to the load (position and velocity), the motor (position and velocity), the torque and the error.

The responses of the considered elastic actuator with a step input are presented for both the control methods to be transparently compared. In this article, the system parameters are shown in Table 3.

5.1. Response of the load-side system with step signal without external torque

This section demonstrates the response of the load-side system without external torque for both control methods to be compared. Regarding the two-feedback loops position control algorithm, the step response is shown in Fig. 10, in which the red dashed line is the desired input, the blue solid line in the upper diagram is the load angular position and the blue solid line in the lower diagram is the load angular velocity. There is an unstable stage at the beginning of the simulation, in which the MRAC is in a learning phase. As shown in Fig. 10, the response on the load-side of the elastic actuator depends on the motor-side system, which has a second order system form.

Likewise, for the method based on reinforcement learning, the step response is shown in Fig. 11.

The two considered control methods show comparable performances. However, the method based on reinforcement learning show slightly worse results in terms of position error. Moreover, the method

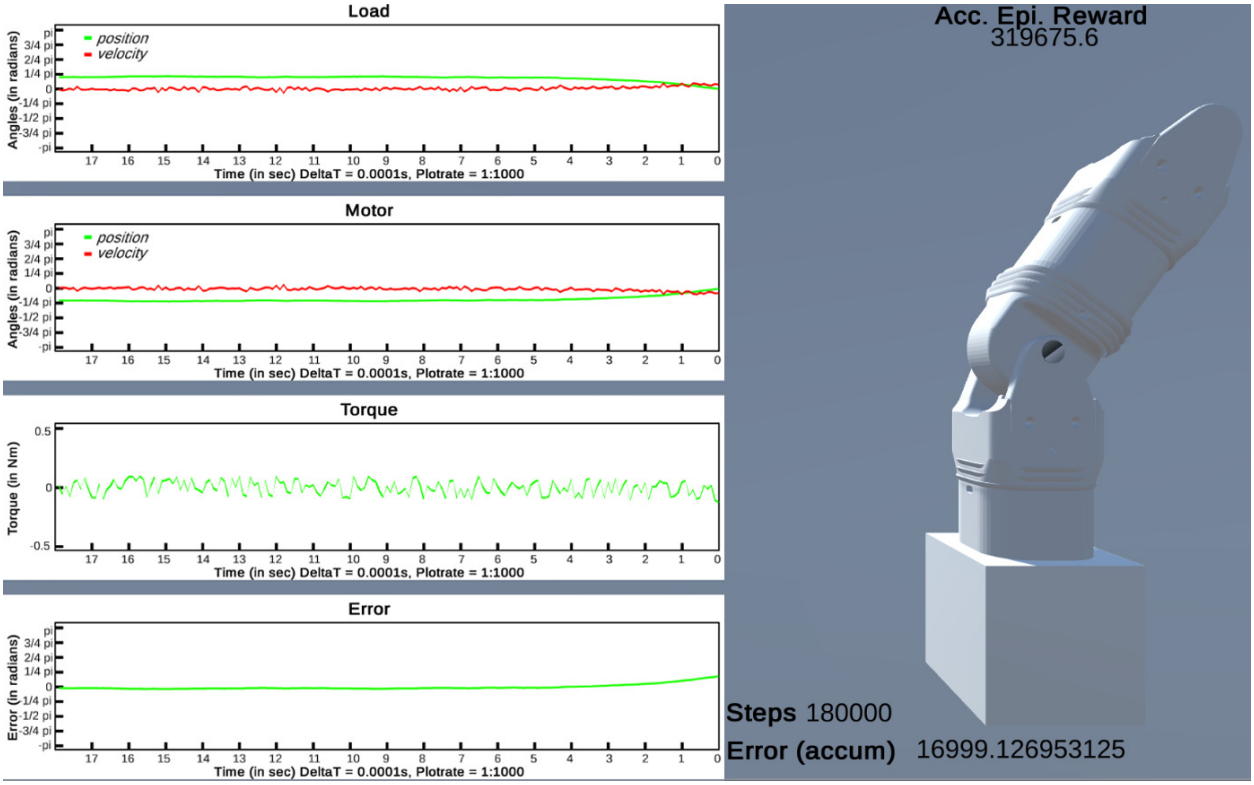

Figure 9. The elastic actuator is simulated in the Unity environment.

based on reinforcement learning is also able to perform a quicker response (about 5s quicker). Furthermore, from an energy perspective, the method based on reinforcement learning generally required slightly more torque (more fluctuations) for the joints over time.

6. Conclusion and future work

A novel elastic actuator was recently designed by our research group for *Serpens*, a low-cost, open-source and highly-compliant multi-purpose modular snake robot [5, 6]. To control the newly developed elastic actuators of *Serpens*, a robust and reliable control algorithm is required. In this perspective, our research group previously proposed a two-feedback loops position control algorithm. The performance of the presented control scheme was demonstrated through simulations [7]. However, the efficiency of the proposed controller is dependent on the effort required for a human to manually construct fuzzy rules and to tune the control parameters. In this paper, the authors explore an alternative approach to the problem based on the use of methods that do not assume a priori knowledge: a solution that derives its properties from a machine learning procedure. This makes it possible to automatically learn the properties of the elastic actuator to be controlled. In particular, a novel controller is presented based on the use of a neural network that is trained with reinforcement learning by applying Proximal Policy Optimisation (PPO). The former two-feedback loops position control algorithm and the newly proposed neural network approach were considered for an extensive comparison in a simulated environment. According to the obtained results, the two considered control methods show similar performances. However, the method based on reinforcement learning showed slightly better response (about 5s quicker) but worse results in terms of position error. Moreover, from an energy point of view, the method based on reinforcement learning also required slightly more torque for the joints over time. It is quite logical to suppose that bigger differences may be identified when controlling redundant manipulators with several actuators. The lesson learned from this study is the significance of applying a machine learning procedure to reduce the need of manually defining fuzzy rules for controlling elastic actuators. However, the implementation of the method based on machine learning requires also requires human intervention in terms of deciding the topology the neural network as well as the parameters of the PPO algorithm. Even though our results show that the two-loop controller

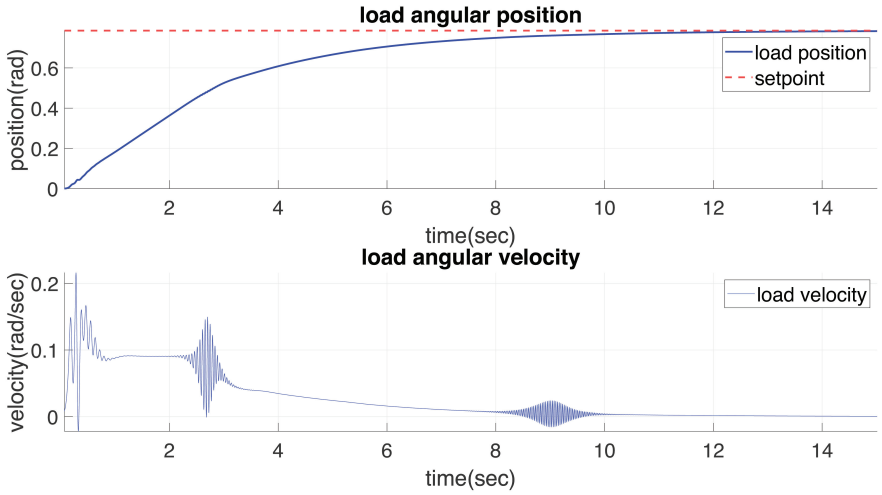

Figure 10. Load-side step response with the two-loop controller.

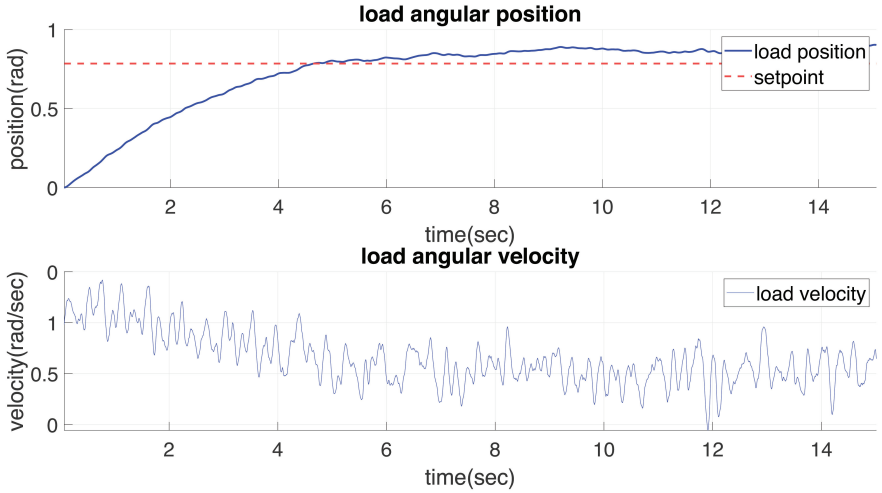

Figure 11. Load-side step response with the neural-network controller.

slightly out-perform the method based on machine learning, it might be useful to adopt the former when the definition of fuzzy rules is more complex. In the authors' opinion, the neural network with reinforcement learning may excel when the model of the system is difficult or impossible to obtain [22].

As future work, iterative machine learning (IML) techniques may be tested to improve the overall performance of the control method based on machine learning. This is in line with the findings presented in [23]. When considering the use of elastic actuators for safety-critical applications, there is a significant need for designing robust machine learning systems that can generate reliable and trustworthy results in the presence of hardware-level faults while also preserving security and privacy [24]. This aspect must be investigated in the future. Furthermore, the control of multiple degrees of freedom may be taken into consideration. Moreover, the validation of the two compared controllers with the physical robot will be implemented. This will also make it possible to realise a general virtual and physical rapid-prototyping framework that allows for the design, simulation and control of series elastic actuators.

References

[1] H. Marvi, C. Gong, N. Gravish, H. Astley, M. Travers, R. L. Hatton, J. R. Mendelson, H. Choset, D. L. Hu, and D. I. Goldman, "Sidewinding with minimal slip: Snake and robot ascent of sandy slopes," *Science*, vol. 346,

no. 6206, pp. 224–229, 2014.

[2] F. Sanfilippo, Ø. Stavdahl, and P. Liljebäck, "Snakesim: a ros-based control and simulation framework for perception-driven obstacle-aided locomotion of snake robots," *Artificial Life and Robotics*, pp. 1–10, 2018.

[3] F. Sanfilippo, J. Azpiazu, G. Marafioti, A. A. Transeth, Ø. Stavdahl, and P. Liljebäck, "A review on perception-driven obstacle-aided locomotion for snake robots," in *Proc. of the 14th International Conference on Control, Automation, Robotics and Vision (ICARCV), Phuket, Thailand*, pp. 1–7, 2016.

[4] F. Sanfilippo, J. Azpiazu, G. Marafioti, A. A. Transeth, Ø. Stavdahl, and P. Liljebäck, "Perception-driven obstacle-aided locomotion for snake robots: the state of the art, challenges and possibilities," *Applied Sciences*, vol. 7, no. 4, p. 336, 2017.

[5] F. Sanfilippo, E. Helgerud, P. A. Stadheim, and S. L. Aronsen, "Serpens: A highly compliant low-cost ros-based snake robot with series elastic actuators, stereoscopic vision and a screw-less assembly mechanism," *Applied Sciences*, vol. 9, no. 3, p. 396, 2019.

[6] F. Sanfilippo, E. Helgerud, P. A. Stadheim, and S. L. Aronsen, "Serpens, a low-cost snake robot with series elastic torque-controlled actuators and a screw-less assembly mechanism," in *Proc. of the IEEE 5th International Conference on Control, Automation and Robotics (ICCAR), Beijing, China*, pp. 133–139, 2019.

[7] T. M. Hua, F. Sanfilippo, and E. Helgerud, "Serpens, a low-cost snake robot with series elastic torque-controlled actuators and a screw-less assembly mechanism," in *Proc. of the IEEE International Conference on Systems, Man, and Cybernetics (IEEE SMC 2019), Bari, Italy*, 2019.

[8] D. Dubois and H. Prade, "Basic issues on fuzzy rules and their application to fuzzy control," in *International Joint Conference on Artificial Intelligence*, pp. 1–14, Springer, 1991.

[9] A. Yang and K. H. Low, "Fuzzy position/force control of a robot leg with a flexible gear system," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2, pp. 2159–2164, 2002.

[10] C. W. De Silva, *Intelligent control: fuzzy logic applications*. CRC press, 2018.

[11] H. Ying, "General analytical structure of typical fuzzy controllers and their limiting structure theorems," *Automatica*, vol. 29, no. 4, pp. 1139–1143, 1993.

[12] P. Fankhauser, M. Hutter, C. Gehring, M. Bloesch, M. A. Hoepflinger, and R. Siegwart, "Reinforcement learning of single legged locomotion," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 188–193, 2013.

[13] B. Yu, J. de Gea Fernández, Y. Kassahun, and V. Bargsten, "Learning the elasticity of a series-elastic actuator for accurate torque control," in *Proc. of the International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pp. 543–552, Springer, 2017.

[14] H. Kobayashi and R. Ozawa, "Adaptive neural network control of tendon-driven mechanisms with elastic tendons," *Automatica*, vol. 39, no. 9, pp. 1509–1519, 2003.

[15] K. J. Åström and B. Wittenmark, *Adaptive control*. Courier Corporation, 2013.

[16] G. C. Lopes, M. Ferreira, A. da Silva Simões, and E. L. Colombini, "Intelligent control of a quadrotor with proximal policy optimization reinforcement learning," in *Proc. of the IEEE Latin American Robotic Symposium, Brazilian Symposium on Robotics (SBR) and Workshop on Robotics in Education (WRE)*, pp. 503–508, 2018.

[17] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[18] P. Ramachandran, B. Zoph, and Q. V. Le, "Swish: a self-gated activation function," *arXiv preprint arXiv:1710.05941*, vol. 7, 2017.

[19] Unity Technologies, "Unity Real-Time Development Platform." `https://unity.com/`, 2019. [Online; accessed 14-June-2019].

[20] A. Juliani, V.-P. Berges, E. Vckay, Y. Gao, H. Henry, M. Mattar, and D. Lange, "Unity: A general platform for intelligent agents," *arXiv preprint arXiv:1809.02627*, 2018.

[21] TensorFlow, "An end-to-end open source machine learning platform." `https://www.tensorflow.org/`, 2019. [Online; accessed 14-June-2019].

[22] N. M. Yazdani and A. Y. Seqerloo, "Performance comparison between classic and intelligent

methods for position control of dc motor.," *International Journal of Electrical & Computer Engineering (2088-8708)*, vol. 4, no. 3, 2014.

[23] N. Banka, W. T. Piaskowy, J. Garbini, and S. Devasia, "Iterative machine learning for precision trajectory tracking with series elastic actuators," in *Proc. of the IEEE 15th International Workshop on Advanced Motion Control (AMC)*, pp. 234–239, 2018.

[24] M. A. Hanif, F. Khalid, R. V. W. Putra, S. Rehman, and M. Shafique, "Robust machine learning systems: Reliability and security for deep neural networks," in *Proc. of the IEEE 24th International Symposium on On-Line Testing And Robust System Design (IOLTS)*, pp. 257–260, 2018.