Artificial neural network based compliant control for robot arms

Vince Jankovics[1], Stefan Mátéfi-Tempfli[1] and Poramate Manoonpong[2]

[1] The Mads Clausen Institute
University of Southern Denmark
Sonderborg, Denmark
Email: vincejankovics@gmail.com
[2] Embodied AI and Neurorobotics Lab
Center for BioRobotics
The Maersk Mc-Kinney Moeller Institute
University of Southern Denmark
Odense M, Denmark
Email: poma@mmmi.sdu.dk

Abstract. The aim of this paper is to present an artificial neural network (ANN) based adaptive nonlinear control approach of a robot arm, with highlight on its capability as a compliant control scheme. The approach is based on a computed torque law and consists of two main components: a feedforward controller (approximated by the ANN) and a proportional-derivative (PD) feedback loop. Here, the feedforward controller is used to approximate the nonlinear system dynamics and can also adapt to the long-term dynamics of the arm while the PD feedback loop can be tuned to obtain proper compliant behaviour to deal with instantaneous disturbances (e.g., collisions). The employed controller structure makes it possible to decouple these two components for individual parameter adjustments. The performance of the control approach is evaluated and demonstrated in physical simulation which shows promising results.

Keywords: nonlinear control, artificial neural network, compliance, robot arm

1 Introduction

Although robot arms have been developed in the past decades, there are still several concerns about their development. In many cases linear control strategies are sufficient by suppressing the nonlinear characteristics of the robot arm system, or using gain scheduled techniques [21]. However, in some cases nonlinear behaviour due to dry friction and backlash can be observed. Therefore, further investigation has to be made in order to achieve a high performance control system that can compensate the nonlinearity [4, 6]. Control techniques to deal with nonlinear systems usually require precise knowledge of the system (e.g. dynamic inversion). Thus, they are difficult to implement in many cases.

From this point of view, model-free control techniques can be used to provide a solution for the system where the equations governing the system are unknown [17]. Artificial neural networks (ANNs) are universal approximators [2], i.e., they are able to approximate any unknown function after a sufficient learning phase. This makes them a good solution for model-free control of nonlinear systems [3, 9, 12, 13, 17].

In addition to the nonlinear control aspect, safety is a key concern for robot arms, since they might have to interact with humans. In order to avoid injuries and achieve safe human-robot interaction different approaches have been developed, such as variable stiffness actuators [18, 14], which requires special hardware development, or safe planning [16], which requires complete perception of the environment. Other approaches include collision detection and reaction [10, 1] that can deal with more dynamic collisions but can complicate the control system design process.

Compared to all these approaches, we present here an alternative control technique that combines the adaptive nature of ANN based control and the virtual compliance control provided by an additional proportional-derivative (PD) control law. This control technique, inspired by [8, 15], can decouple between short-term (collisions) and long-term (changes in the environment and the system) disturbance compensation.

This provides a simple and intuitive controller design without any hardware modification for a robot arm. Additionally, in this study, we aim to also investigate whether the tracking error of the arm can be kept low via the ANN based control and proper compliant tuning such that the arm can react to collisions with flexibility.

The article is organized as follows. First we describe the robot arm model used in this study. Second, we present the artificial neural network based compliant control approach together with its subcomponents for generating movement and compliant behaviour of the arm. Third, we illustrate the performance of the controller as an adaptive compliant control solution, followed by conclusion.

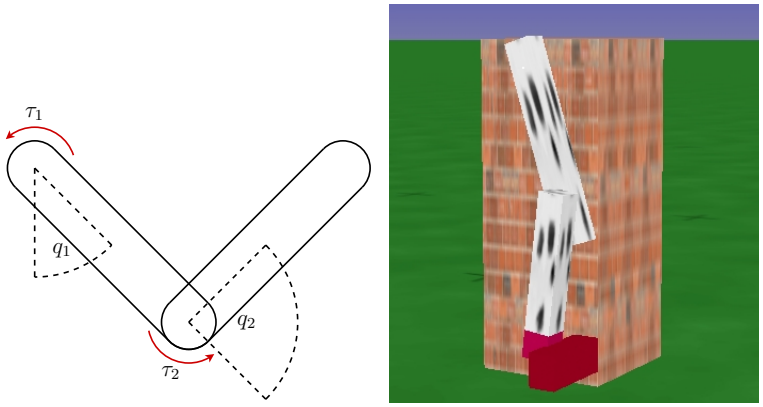
2 Robot arm model

The dynamics of an n-link rigid robotic manipulator can be expressed in the Lagrange form as:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) + \mathbf{F}(\dot{\mathbf{q}}) + \boldsymbol{\tau}_d = \boldsymbol{\tau}(t), \tag{1}$$

where $\mathbf{q}(t) \in \mathbb{R}^n$ the joint variable vector, $\mathbf{M}(\mathbf{q})$ the inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ the Coriolis and centripetal matrix, $\mathbf{G}(\mathbf{q})$ the gravity vector, and $\mathbf{F}(\dot{\mathbf{q}})$ the friction. Bounded unknown disturbances (including modeling errors) are denoted by $\boldsymbol{\tau}_d$, and the applied torque is $\boldsymbol{\tau}(t)$ [7]. The structure of the 2DOF robot arm used in this study is shown in Fig. 1a and its physically 3D model simulated in the realistic robot simulator LpzRobots [11] is shown in Fig. 1b.

The computed torque control law can cancel all nonlinearity of the system by adding a linear error correction term to the feedforward part, which can be



(a) (b)

Fig. 1: (a) 2-link planar robot arm. (b) The simulated robot arm using the LPZRobots simulation environment [11].

described as:

$$\boldsymbol{\tau}(t) = \underbrace{\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) + \mathbf{F}(\dot{\mathbf{q}})}_{\text{feedforward}} - \underbrace{\mathbf{M}(\mathbf{q})(\mathbf{K}_v\dot{\mathbf{e}} + \mathbf{K}_p\mathbf{e})}_{\text{feedback}}, \tag{2}$$

where $\mathbf{e} = \mathbf{q}_d - \mathbf{q}$ (subtracting desired joint position vector from the actual one), and $\mathbf{K}_v$, $\mathbf{K}_p$ are constant gain matrices.

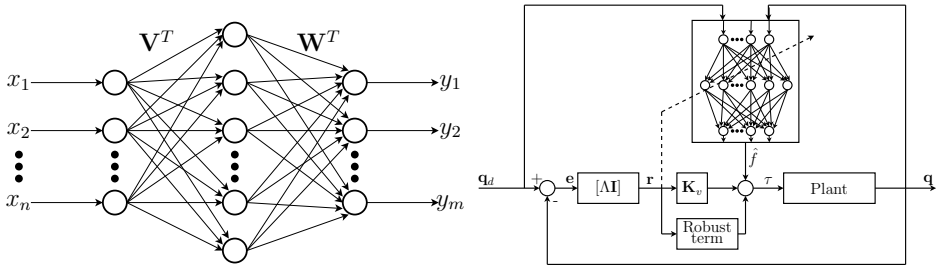
With this strategy, the computed torque consists of two components, which are feedforward and feedback. This separability makes it possible to generate compliant behaviour, and still keep the tracking error low. The feedforward part, acting as trajectory control, takes care of the desired movement generation of the system (i.e., carrying an object along a path) while the feedback part with its gains ($\mathbf{K}_v$, $\mathbf{K}_p$) allows for compliance and flexibility of the robot joints when instantaneous disturbances (such as collisions) occur.

3 Artificial neural network based compliant control

Based on (2), the control of the 2-link robot arm (Fig. 1) consists of two parts, where an ANN can approximate the feedforward component (Fig. 2).

Function representation: A multilayer ANN is proven to be able to approximate any function with finitely many discontinuities to arbitrary precision [5]. This makes it an excellent tool for nonlinear control systems [8], i.e. a properly trained network can represent any function, so:

$$\mathbf{f}(\mathbf{x}) = \mathbf{W}^T\mathcal{F}(\mathbf{V}^T\mathbf{x}) + \varepsilon(\mathbf{x}), \tag{3}$$



(a) (b)

Fig. 2: The ANN's and the control system's structure. (a) A 3 layer ANN with $\mathbf{x}$ as input (e.g. $[\mathbf{q}_d,\ \mathbf{q}]$) and $\mathbf{y}$ as output ($\mathbf{f}$, the torques required), which is described by (3), (b) The whole system with the control and tuning signals.

where $\mathbf{f}(x)$ is an unknown function, and $\varepsilon(\mathbf{x})$ is the reconstruction error. It is proven in [2] that there exists ideal weights $\mathbf{W}$ and $\mathbf{V}$ with finite number of neurons such that the reconstruction error is 0 ($\varepsilon = 0$). In practice it is sufficient if $\|\varepsilon\| < \varepsilon_n$, i.e., the function approximation is good enough for the application.

The choice of activation function can change the approximation capabilities of the network significantly [5]. Even though it is possible in theory to reconstruct an arbitrary function, a prior knowledge of the system can reduce the adaptation time and increase the overall performance.

Since the system in question is mechanical, it can be mostly approximated with smooth functions but there are also nonlinear friction effects (static friction) which can be approximated by piecewise continuous functions. Taking this into account an augmented network structure was considered as it is suggested in [15], which consist of a hidden layer with 2 different neurons, with $\mathcal{F}_1$ and $\mathcal{F}_2$ activation functions (smooth and nonsmooth, respectively).

The network's output is then:

$$\mathbf{f}(\mathbf{x}) = \hat{\mathbf{W}}_1^T \mathcal{F}_1(\hat{\mathbf{V}}_1^T \mathbf{x}) + \hat{\mathbf{W}}_2^T \mathcal{F}_2(\hat{\mathbf{V}}_2^T \mathbf{x}) + \varepsilon(\mathbf{x}), \tag{4}$$

where the weights $\mathbf{W}_1$, $\mathbf{V}_1$ connect the neurons with smooth, and the weights $\mathbf{W}_2$, $\mathbf{V}_2$ connect the neurons with the nonsmooth activation function to the input and output layers.

Here, a sigmoid transfer function is used as our smooth activation function while a nonsmooth activation function for friction compensation can be modelled as [15]:

$$\mathcal{F}_2(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ 1 - \exp(-x) & \text{if } x > 0 \end{cases} \tag{5}$$

The computed torque law as shown in (2) combined with (4) becomes:

$$\boldsymbol{\tau}(t) = \underbrace{\hat{\mathbf{W}}_1^T \mathcal{F}_1(\hat{\mathbf{V}}_1^T \mathbf{x}) + \hat{\mathbf{W}}_2^T \mathcal{F}_2(\hat{\mathbf{V}}_2^T \mathbf{x})}_{\text{feedforward}} + \underbrace{\mathbf{K}_v \mathbf{r}}_{\text{feedback}} + \underbrace{\mathbf{K}_z(\|\hat{\mathbf{Z}}\| + Z_M)\mathbf{r}}_{\text{robustifying term}}, \tag{6}$$

where the signals:

$$\begin{aligned}
&\mathbf{e} = \mathbf{q}_d - \mathbf{q} && \text{tracking error,}\\
&\mathbf{r} = \dot{\mathbf{e}} + \boldsymbol{\Lambda}\mathbf{e} && \text{filtered tracking error,}\\
&\mathbf{x} = \left[\mathbf{e}^T \dot{\mathbf{e}}^T \mathbf{q}_d^T \dot{\mathbf{q}}_d^T \ddot{\mathbf{q}}_d^T\right]^T && \text{ANN input vector,}
\end{aligned}$$

and $\hat{\mathbf{Z}}$ is a block diagonal matrix containing $\hat{\mathbf{V}}$ and $\hat{\mathbf{W}}$.

The *robustifying term* is added to guarantee stability with higher weights, proposed by [8], and the design parameters $\boldsymbol{\Lambda}$, $\mathbf{K}_v$, $\mathbf{K}_z$ are symmetric, positive definite gain matrices, and Z_M is a bound on the unknown target weight norms.

Weight tuning: We use a standard error backpropagation learning algorithm with an additional so-called forgetting term to train the network. The forgetting term is used to introduce saturation of the weights, which guarantees bounded weights, i.e. stable behaviour [9]. The weights are adjusted during the learning process, which is described by the differential equations:

$$\dot{\hat{\mathbf{W}}}_1 = \mathbf{F}\left[\left(\hat{\mathcal{F}}_1 - \hat{\mathcal{F}}_1'\hat{\mathbf{V}}_1\mathbf{x}\right)\mathbf{r}^T - \kappa\|\mathbf{r}\|\hat{\mathbf{W}}_1\right], \tag{7a}$$

$$\dot{\hat{\mathbf{W}}}_2 = \mathbf{G}\left[\hat{\mathcal{F}}_2\mathbf{r}^T - \kappa\|\mathbf{r}\|\hat{\mathbf{W}}_2\right], \tag{7b}$$

$$\dot{\hat{\mathbf{V}}}_1 = \mathbf{H}\left[\mathbf{x}\left(\hat{\mathcal{F}}_1'^T\hat{\mathbf{W}}_1\mathbf{r}\right)^T - \kappa\|\mathbf{r}\|\hat{\mathbf{V}}_1\right], \tag{7c}$$

$$\dot{\hat{\mathbf{V}}}_2 = 0. \tag{7d}$$

$\mathbf{F}$, $\mathbf{G}$, and $\mathbf{H}$ are the learning rates, and $\hat{\mathcal{F}} \equiv \mathcal{F}(\hat{\mathbf{V}}^T x)$ and in case of sigmoid activation function, the derivative is $\hat{\mathcal{F}}' \equiv \hat{\mathcal{F}}(1 - \hat{\mathcal{F}})$. Choosing higher rates makes the leaning process faster, but increasing them too much can produce oscillatory and unstable behaviour. $\hat{\mathbf{V}}_2$ is here kept constant.

Note that this learning rule is described in continuous time, but in a discrete time controller the forward Euler method is sufficient with small enough $\mathrm{d}t$, so the weights are updated at each time step as:

$$\hat{\mathbf{W}}_{1,k+1} = \hat{\mathbf{W}}_{1,k} + \mathrm{d}t\dot{\hat{\mathbf{W}}}_{1,k}\ , \tag{8a}$$

$$\hat{\mathbf{W}}_{2,k+1} = \hat{\mathbf{W}}_{2,k} + \mathrm{d}t\dot{\hat{\mathbf{W}}}_{2,k}\ , \tag{8b}$$

$$\hat{\mathbf{V}}_{1,k+1} = \hat{\mathbf{V}}_{1,k} + \mathrm{d}t\dot{\hat{\mathbf{V}}}_{1,k}\ . \tag{8c}$$

Compliant control: As mentioned above, the feedforward part is responsible for providing the necessary input to the system based on the modelled dynamics, and on top of that the feedback part (PD loop) eliminates the unmodelled or instantaneous disturbances, which are not present in the system long enough for the adaptation, so the behaviour can be described by:

$$\mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\tau}_d = \hat{\mathbf{f}} + \boldsymbol{\tau}_{PD}, \tag{9}$$

where $\mathbf{f}$ describes the dynamics of the system, as it is described by (1), $\hat{\mathbf{f}}$ is the approximation of this by the ANN, and $\boldsymbol{\tau}_{PD}$ is the PD compensation. If it is assumed that $\mathbf{f} \approx \hat{\mathbf{f}}$, then the disturbance is eliminated by the feedback control. The PD law is described as:

$$\boldsymbol{\tau}_{PD} = \mathbf{K}_p\mathbf{e} + \mathbf{K}_d\dot{\mathbf{e}}. \tag{10}$$

This law results in a virtual spring-damper system, with $\mathbf{e} = 0$ equilibrium point. Choosing higher or lower $\mathbf{K}_p$ and $\mathbf{K}_d$ can make the system's response stiffer or softer.

This can be analogously applied with the control law described by (6) for robot arm control. It is important to note that the robustifying term, which has the same effect as the PD part, needs to be taken into account when choosing $\mathbf{K}_v$ and $\boldsymbol{\Lambda}$. It will make the arm stiffer with the increasing weight norms, so the compliant performance requirements can be violated in the initial learning phase and in extreme approximation errors (i.e., when the network is not trained properly).

4 Simulation results

The control algorithm was tested in simulations. The performance was compared to a conventional PID controller.

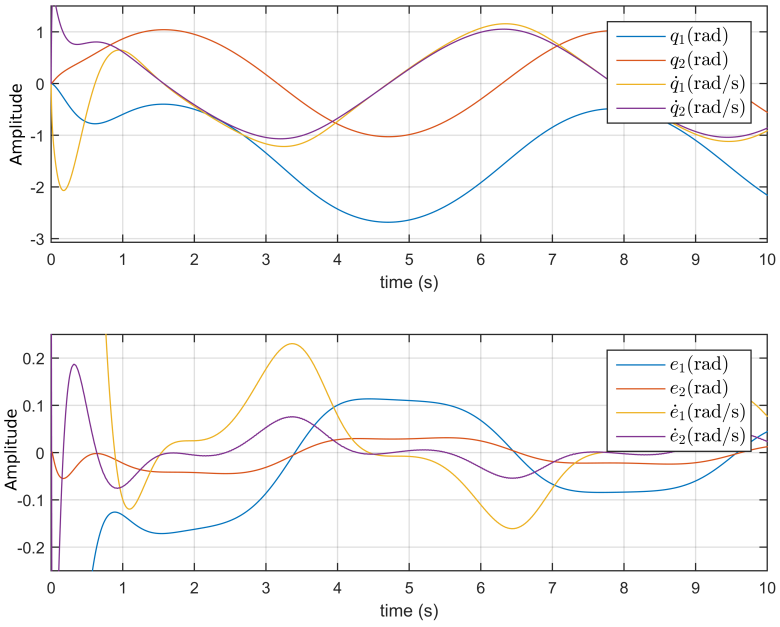
Here, the parameters of our developed controller, described by (6) and (7) were set to: $K_v = 1,\ \Lambda = 5,\ K_z = 20,\ Z_m = 1,\ \kappa = 0.0001,\ F = 100,\ G = 100,\ H = 100$. The network's input was the vector $\mathbf{x} = \left[\mathbf{e}^T\dot{\mathbf{e}}^T\mathbf{q}_d^T\dot{\mathbf{q}}_d^T\ddot{\mathbf{q}}_d^T\right]^T$ and the output was the approximated torques τ required to follow the desired trajectories. Therefore, the number of input, output, and hidden neurons was set to 10, 2, and 16, respectively.

Figure 3 shows the performance of the PID and the ANN controller for a sinusoidal reference signal. The PD gains for the developed algorithm is kept low so the response mainly represents the ANN's performance. The learning curve is shown in Fig. 4. It can be seen that the network successfully learned after t=0.5s where the weights converge.

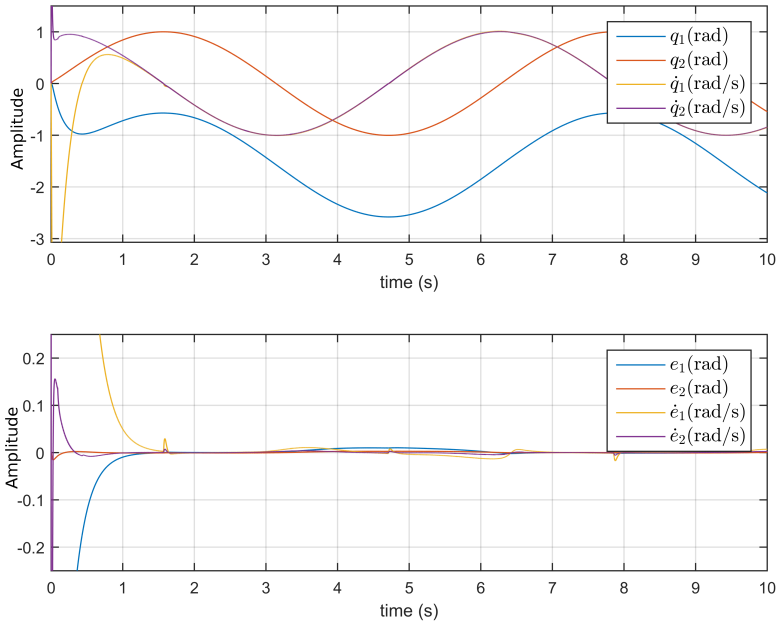
Figure 5 shows the response to disturbances, an instantaneous push force at t=2.5s, and a change of the weight of the carried mass at t=5.0s. The learning curve shown in Fig. 6, which shows that the weights are not changed significantly at t=2.5s, so the disturbance is compensated by the PD loop, however at t=5.0s the ANN learns the new dynamics of the system eliminating the tracking error, that the PID controller cannot do with low feedback gains.

Changing the learning rates ($F,\ G,\ H$) can increase the speed of learning, but it also means that the network reacts to collisions which is undesirable (and can also lead to oscillatory response), so the rates were empirically set to the values described above to achieve the desired performance.

The arm in motion can be seen at youtu.be/ZHHx3eUzBc4.



(a)



(b)

Fig. 3: The system's performance for sinusoidal reference signal, showing the joint positions and velocities, and the tracking errors. (a) PID with $K_p = 100$, $K_d = 30$, $K_i = 20$. (b) ANN with K_v=1, L=5, K_z=20, Z_m=1, κ=0.0001, F=100, G=100, H=100.

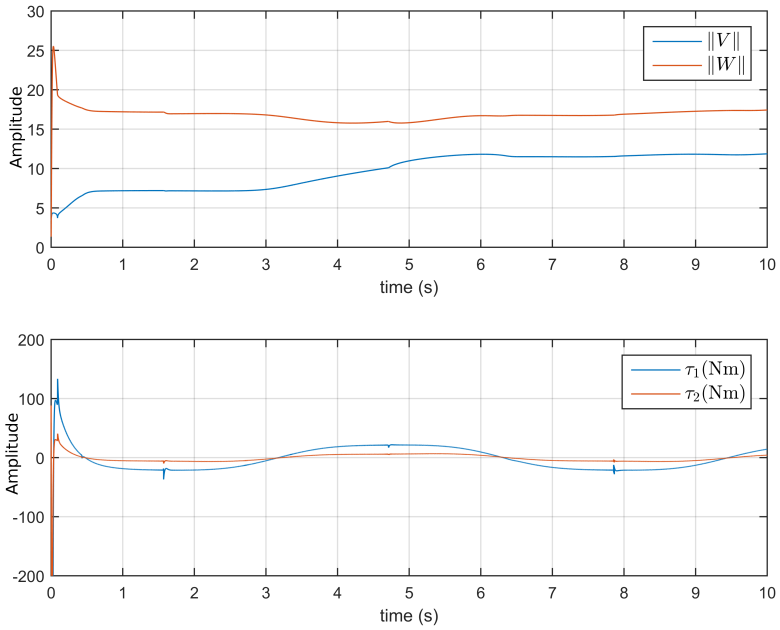
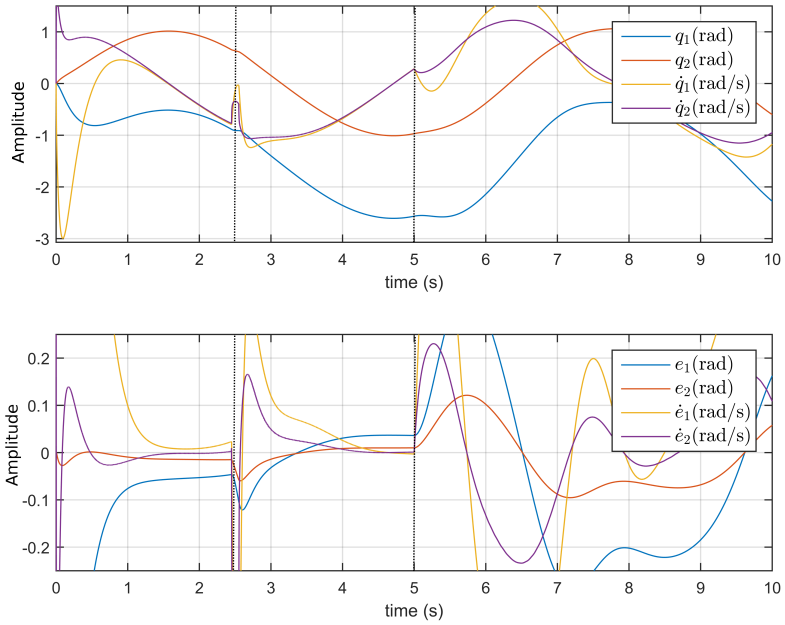
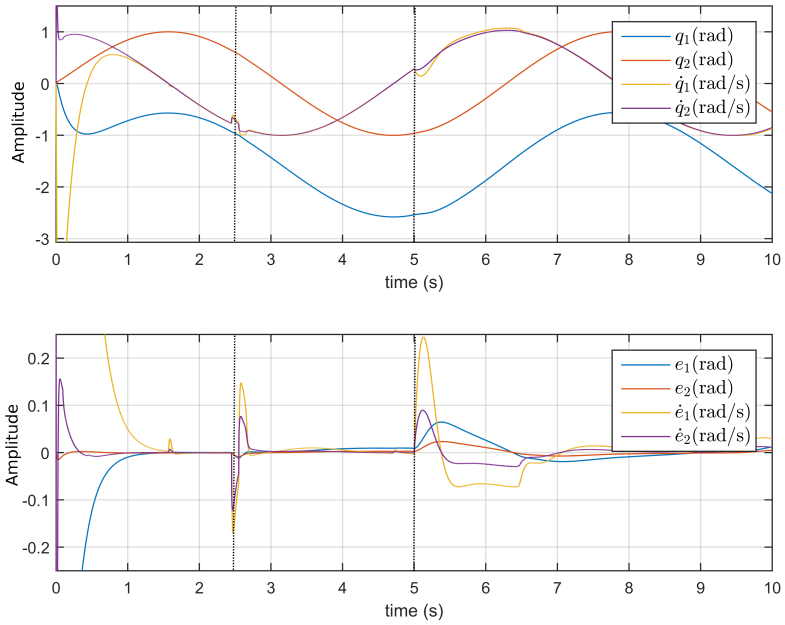


Fig. 4: ANN learning curve and actuator torques with K_v=1, L=5, K_z=20, Z_m=1, κ=0.0001, F=100, G=100, H=100, without disturbances.



(a)



(b)

Fig. 5: The system's performance for sinusoidal reference signal with disturbances at $t = 2.5$s and $t = 5$s (dotted lines), showing the joint positions and velocities, and the tracking errors. (a) PID with $K_p = 300$, $K_d = 90$, $K_i = 60$. (b) ANN with K_v=1, L=5, K_z=20, Z_m=1, κ=0.0001, F=100, G=100, H=100.

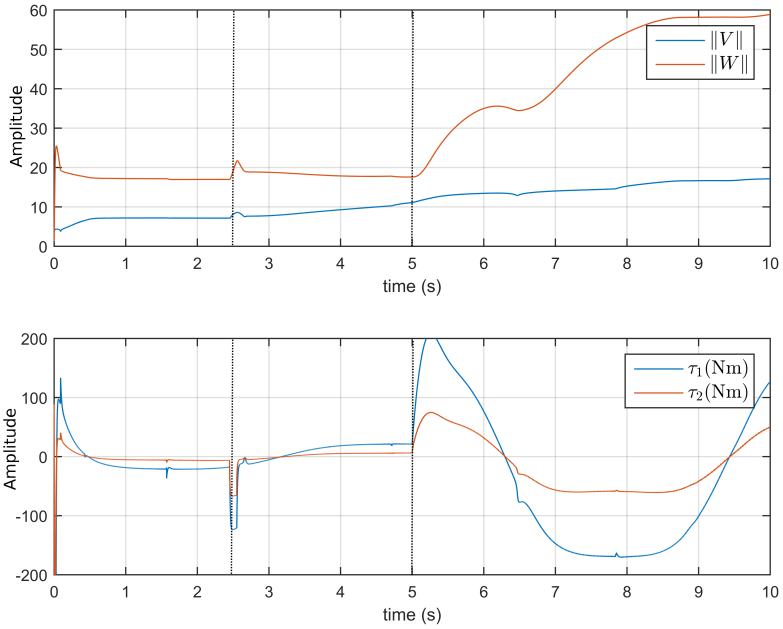


Fig. 6: ANN learning curve and actuator torques with K_v=1, L=5, K_z=20, Z_m=1, κ=0.0001, F=100, G=100, H=100, with disturbances at $t = 2.5$s and $t = 5$s (dotted lines).

5 Conclusions

In this work we developed an ANN based adaptive compliant control algorithm and investigated its performance. The main benefit of using this algorithm is the simple and intuitive way of decoupling the adaptive ANN based control and compliant behaviour making the controller design more intuitive. The algorithm was successfully applied to a simulated 2DOF robot arm for robust, adaptive, compliant behaviour generation of the arm.

A similar approach is described by [20, 19], using a modular neural network together with a virtual agonist-antagonist muscle mechanism to generate energy-efficient walking pattern on different surfaces. That solution requires force sensors at the end-effectors (i.e., the tip of the legs) to achieve the compliant behaviour; while, in our solution and implementation, extra force sensors are not required. We only use joint position feedback embedded in the motors. Furthermore, our control approach is torque control which is different from the position control approach of [20, 19]. In principle, torque control allows for better compliant adaptation and regulation (e.g., when the robot is in contact with objects or receives disturbances) compared to position control; thereby, our developed approach here is suitable for robot arm control and object manipulation.

In the future work, we will implement the controller on a real robot arm and test its performance to evaluate in pick and place tasks involving disturbances.

References

1. Haddadin, S., Albu-Schäffer, A., Luca, A.D., Hirzinger, G.: Collision detection and reaction: A contribution to safe physical human-robot interaction. In: Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on. pp. 3356–3363. IEEE (2008)
2. Hornik, K., Stinchcombe, M., White, H.: Multilayer Feedforward Networks are Universal Approximators. Neural Networks 2, 359–366 (1989)
3. Hunt, K.J., Sbarbaro, D., Zbikowski, R., Gawthrop, P.J.: Neural networks for control systems - A survey. Automatica 28(6), 1083–1112 (1992)
4. Isidori, A.: Nonlinear control systems. Springer Science & Business Media (2013)
5. Kröse, B., van der Smagt, P.: An Introduction to Neural Networks. The University of Amsterdam, 8th edn. (Nov 1996)
6. Krstic, M., Kokotovic, P.V., Kanellakopoulos, I.: Nonlinear and adaptive control design. John Wiley & Sons, Inc. (1995)
7. Lewis, F.L., Dawson, D.M., Abdallah, C.T.: Robot Manipulator Control. Marcel Dekker, Inc. (2004)
8. Lewis, F.L., Yegildirek, A., Liu, K., Member, S.: Multilayer Neural-Net Robot Controller woth Guaranteed Tracking Performance. IEEE transactions on neural networks 7(2) (1996)
9. Lewis, F., Jagannathan, S., Yesildirak, A.: Neural network control of robot manipulators and non-linear systems. CRC Press (1998)
10. Luca, A., Albu-Schaffer, A., Haddadin, S., Hirzinger, G.: Collision Detection and Safe Reaction with the DLR-III Lightweight Manipulator Arm. pp. 1623–1630. IEEE (Oct 2006)

11. Martius, G., Hesse, F., Güttler, F., Der, R.: LPZROBOTS: a free and powerful robot simulator. Available online at: http://robot.informatik.uni-leipzig.de/software (2010)
12. Miller, W.T., Werbos, P.J., Sutton, R.S.: Neural networks for control. MIT press (1995)
13. Omidvar, O., Elliott, D.L.: Neural systems for control. Elsevier (1997)
14. Schiavi, R., Grioli, G., Sen, S., Bicchi, A.: VSA-II: a novel prototype of variable stiffness actuator for safe and performing robots interacting with humans. In: Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on. pp. 2171–2176. IEEE (2008)
15. Selmic, R.R., Lewis, F.L.: Neural-network approximation of piecewise continuous functions: Application to friction compensation. IEEE Transactions on Neural Networks 13(3), 745–751 (2002)
16. Sisbot, E.A., Marin-Urias, L.F., Alami, R., Simeon, T.: A human aware mobile robot motion planner. Robotics, IEEE Transactions on 23(5), 874–883 (2007)
17. Spall, J.C., Cristion, J.A.: Model-free control of nonlinear stochastic systems with discrete-time measurements. Automatic Control, IEEE Transactions on 43(9), 1198–1210 (1998)
18. Tonietti, G., Schiavi, R., Bicchi, A.: Design and control of a variable stiffness actuator for safe and fast physical human/robot interaction. In: Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on. pp. 526–531. IEEE (2005)
19. Xiong, X., Wörgötter, F., Manoonpong, P.: Neuromechanical control for hexapedal robot walking on challenging surfaces and surface classification. Robotics and Autonomous Systems 62(12), 1777–1789 (2014)
20. Xiong, X., Wörgötter, F., Manoonpong, P.: Virtual agonist-antagonist mechanisms produce biological muscle-like functions: An application for robot joint control. Industrial Robot: An International Journal 41(4), 340–346 (2014)
21. Yang, W., Hammoudi, N., Herrmann, G., Lowenberg, M., Chen, X.: Dynamic gain-scheduled control and extended linearisation: extensions, explicit formulae and stability. International Journal of Control 88(1), 163–179 (2015)