Artificial Intelligence 312 (2022) 103771

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Q-Learning-based model predictive variable impedance control for physical human-robot collaboration ☆

Loris Roveda [a,*], Andrea Testa [b], Asad Ali Shahid [a], Francesco Braghin [b], Dario Piga [a]

[a] *Istituto Dalle Molle di studi sull'Intelligenza Artificiale (IDSIA), Scuola Universitaria Professionale della Svizzera Italiana (SUPSI), Università della Svizzera italiana (USI), via la Santa 1, 6962, Lugano, Switzerland*
[b] *Politecnico di Milano, Department of Mechanical Engineering, via La Masa 1, 20156, Milano, Italy*

A R T I C L E   I N F O

Article history:
Received 21 September 2021
Received in revised form 18 July 2022
Accepted 6 August 2022
Available online 11 August 2022

Keywords:
Physical human-robot collaboration
Industry 4.0
Machine learning
Model-based reinforcement learning control
Neural networks
Q-Learning
Stability
Variable impedance control

A B S T R A C T

Physical human-robot collaboration is increasingly required in many contexts (such as industrial and rehabilitation applications). The robot needs to interact with the human to perform the target task while relieving the user from the workload. To do that, the robot should be able to recognize the human's intentions and guarantee safe and adaptive behavior along the intended motion directions. The robot-control strategies with such attributes are particularly demanded in the industrial field, where the operator guides the robot manually to manipulate heavy parts (*e.g.*, while teaching a specific task). With this aim, this work proposes a Q-Learning-based Model Predictive Variable Impedance Control (Q-LMPVIC) to assist the operators in a physical human-robot collaboration (pHRC) tasks. A Cartesian impedance control loop is designed to implement a decoupled compliant robot dynamics. The impedance control parameters (*i.e.*, setpoint and damping parameters) are then optimized online in order to maximize the performance of the pHRC. For this purpose, an ensemble of neural networks is designed to learn the modeling of the human-robot interaction dynamics while capturing the associated uncertainties. The derived modeling is then exploited by the model predictive controller (MPC), enhanced with the stability guarantees by means of Lyapunov constraints. The MPC is solved by making use of a Q-Learning method that, in its online implementation, uses an actor-critic algorithm to approximate the exact solution. Indeed, the Q-learning method provides an accurate and highly efficient solution (in terms of computational time and resources). The proposed approach has been validated through experimental tests, in which a Franka EMIKA panda robot has been used as a test platform. Each user was asked to interact with the robot along the controlled vertical z Cartesian direction. The proposed controller has been compared with a model-based reinforcement learning variable impedance controller (MBRLC) previously developed by some of the authors in order to evaluate the performance. As highlighted in the achieved results, the proposed controller is able to improve the pHRC performance. Additionally, two industrial tasks (a collaborative assembly and a collaborative deposition task) have been demonstrated to prove the applicability of the proposed solution in real industrial scenarios.

☆ This paper is part of the Special Issue: "Risk-aware Autonomous Systems: Theory and Practice".
* Corresponding author.
E-mail addresses: loris.roveda@idsia.ch (L. Roveda), andrea8.testa@mail.polimi.it (A. Testa), asadali.shahid@idsia.ch (A.A. Shahid), francesco.braghin@polimi.it (F. Braghin), dario.piga@supsi.ch (D. Piga).

https://doi.org/10.1016/j.artint.2022.103771
0004-3702/© 2022 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

1. Introduction

1.1. Context

To meet customers needs, which are becoming more and more oriented on tailor-made products, companies are updating their production processes by means of new flexible and agile tools [1]. In this context, collaborative robotics plays a key role [2], providing powerful solutions to assist the operators in the execution of different activities, such as co-manipulation [3,4], task's knowledge transfer to the robotic system [5,6], easy programmable and deployable applications [7], etc. Physical human-robot collaboration (pHRC) is currently one of the most investigated topics [8]. In fact, pHRC is nowadays demanded in many fields of applications, both for collaborative robots [9] and exoskeletons [10]. However, many open issues in the state of the art are still to be overcome, in particular considering safety/stability guarantees in the human-robot interaction, human-robot dynamics modeling, human's intention recognition (for active assistance/empowering purposes), and computation efficiency (for real-time control adaptation and optimization).

To tackle the above mentioned issues within the pHRC scenario, this paper proposes a Q-Learning-based Model Predictive Variable Impedance Control (Q-LMPVIC) to assist the operator while physically interacting with a collaborative robot. Based on Cartesian impedance control (providing to the controlled manipulator a compliant and decoupled behavior in the Cartesian space), an MPC is designed in order to online optimize its parameters (*i.e.*, setpoint and damping parameters) to assist the user along the detected intended motion direction(s), maximizing the collaboration performance. The MPC exploits a learned human-robot interaction dynamics model, obtained by means of an ensemble of neural networks. Therefore, the lack of sophisticated analytical models for the human-robot interaction dynamics is overcome, employing a method that is capable to capture the complexity and uncertainties of such a dynamics. An MPC objective function is designed in order to minimize the user's effort during the collaboration with the robot. Indeed, the user's intention of motion can be detected, making it possible to assist him/her along the intended direction(s) of motion. The designed MPC is also enhanced with stability guarantees by means of Lyapunov constraints. In such a way, safety/stability issues are tackled by the proposed methodology. The MPC is then (online) solved making use of a Q-Learning method, exploiting an actor-critic algorithm to approximate its exact solution. The obtained solution is accurate and highly efficient, being able to tackle the issue related to computation efficiency that might compromise the implementation of the controller for real applications.

In the following Section, the state of the art related to the pHRC control is addressed, to highlight the open issues in the field and the solutions provided by the proposed approach.

1.2. Related work

Among other strategies [11,12], physical human-robot collaboration (pHRC) is commonly enabled by implementing a low-level impedance controller [13], that provides the robot with a safe and compliant behavior, suitable for interacting with the surrounding environment (including human subjects [14]). The impedance control parameters (*i.e.*, mass/inertia, stiffness, damping, and setpoint) are then tuned/adapted by means of high-level control strategies during the execution of a task [15], *e.g.*, to achieve human-like adaptability skills [16,17], to maximize the human-robot collaboration performance [18], etc. Such high-level control strategies can be designed using the analytical models of the human-environment interaction [19]. However, the solutions realized through these methods are limited by the specific modeling adapted and the impossibility of the models to capture the complex interaction dynamics. Therefore, machine learning (ML)-based approaches have been investigated to implement flexible controllers. Two types of ML-based solutions are available in the state of the art: model-based ML approaches [20], and model-free ML approaches [21]. Model-based ML approaches provide powerful algorithms for control tuning purposes that are capable of capturing the complex and uncertain interaction dynamics. The main drawback of such strategies consists in the limited variation of task conditions that can be faced by the proposed controllers. In order to be effective, the adopted models should accurately represent the target scenario, losing generalizability [22]. Model-free approaches, on the other hand, allow to achieve acceptable results in a wide set of scenarios by exploiting an autonomous tuning through trial-and-error. However, the tuning procedures are costly (both in terms of the computational resources and the time), requiring a vast amount of trails to achieve the target performance [4].

Many efforts are, therefore, put into the development of combined solutions exploiting the advantages of both model-based and model-free ML solutions. In [23] is implemented a systematic approach to optimize the gains of an admittance controller online, without any prior knowledge of the target position or other task characteristics. A fuzzy Q-Learning algorithm is used to regulate damping so that the robot trajectory approaches the minimum jerk and the cooperation becomes more effective. The partitioning of the robot state with fuzzy sets is an efficient method to deal with the curse of dimensionality of continuous space. However, it requires a number of parameters to be manually tuned and it works selecting a deterministic action from a small set using fuzzy Q values to obtain a quicker convergence. According to [24], restricting the search for optimal action to the agent's action set or a restricted set of Q values, it is a myopic idea. Herein, their proposal is to employ genetic algorithm as stochastic optimizer for action selection at each stage of Fuzzy Q-Learning-based controller. The associated increase in computational burden is mitigated by the improvement in performance. In [25], a Gaussian Process (GP) model is learned as an approximation of the dynamical system, which is then used to predict the long-term state evolution for internal simulation. The impedance controller parameters are then optimized exploiting the learned modeling. However, this computation can take minutes, which is not suitable for a real-time implementation of the controller in industrial applications. Thus, while the suggested approach is able to provide an accurate enough model of the interaction dynamics, it becomes slow for a large amount of data and it can only be used with few loss functions. In addition, model-based approaches generally perform better under an assumption of smooth and continuous dynamics, which is far from what it is observed in human-robot collaboration tasks [20]. Due to the above described reasons, other authors have focused on approximating the human-robot interaction with simplified models defined a priori. [26], based on the research in [27], shows how the humans' central nervous system follows a latent desired trajectory regulated by an impedance control scheme. In the paper, the authors model the dynamics of a human arm using this approach, estimating the impedance parameters and the desired trajectory with GPs. Instead, [28] exploits a theoretical study on the arm movements of two operators involved in a dyadic manipulation task. The resultant interaction model is able to estimate the interaction force during the reaching movement as a function of the measured force applied by the human. With these strategies, it is possible to design variable impedance controllers that can be used in real-time. However, these methods are not able to capture the real non-linear dynamics of the human-robot interaction. In [29], it is proposed a novel framework that exploits electromyography (EMG) and dynamic manipulability measurements of the human arm to provide the robot with the information about the human motor behavior and task requirements. Through this information, the robot can adapt its behavior by a phase-dependent regulation of trajectories, force and impedance (*i.e.*, self-complying/stiffening), to provide a human-like complementary assistive response. This method proves to be intuitive and effective, but it requires a special measurement system to be implemented, and it adapts only the impedance control stiffness in a predefined range. In addition to this, some parameters have to be specified offline to define the task and the expected behavior from the robot. In [30], the motion intention of the human is defined as the desired trajectory in the employed human limb model, which is estimated by radial basis function neural networks (RBFNN). The estimated motion intention is integrated into impedance control to make the robot "actively" following its human partner. However, for small data-sets, one RBFNN could suffer from over-fitting, generating errors in the estimation of the setpoint. Moreover, no other impedance parameters are adapted to obtain a compliant behavior. [20] focused on estimating the non-linear dynamics of the human-robot interaction. An ensemble of Artificial Neural Networks (ANNs) is used to learn the interaction dynamics model to design a Model-Based Reinforcement Learning (MBRL) variable impedance controller for pHRC. ANNs are trained offline to embed the HRC dynamics. As before, with less data, the problem of over-fitting of one ANN may recur; the proposed approach uses an ensemble of five ANNs to overcome this issue while allowing to capture the uncertainties of the real HRC dynamics, thus improving the modeling performance for prediction purposes. The trained model is then kept updated during the execution of a collaborative task. This model is used by a Model Predictive Controller (MPC) with Cross Entropy Method (CEM) for the real-time optimization (*i.e.*, the computation is performed at each control step) of the impedance control parameters (*i.e.*, damping and stiffness parameters), considering as an objective the minimization of the human effort during the HRC task execution (*i.e.*, minimizing the force applied by the human to the robot). However, the proposed approach has some drawbacks. The impedance control setpoint update law is predefined, and its optimization is not considered. Furthermore, the number of samples and iterations in the CEM is limited by the requirement to conclude the computation within a specified time (*i.e.*, enforced by the control frequency). Moreover, no constrains are defined within the MPC to enforce stability. The performance of the controller is strictly dependent on the weights used in the cost function and on the setpoint update function, without any theoretical guarantee of stability.

As highlighted in the above discussion, one of the major open issues in the design of pHRC controllers is related to the stability guarantees. Stability can be imposed by including an appropriately designed terminal cost and/or terminal constraints in the control problem definition [31], but it can also be inherited by the Control Lyapunov Functions (CLFs). CLFs are commonly used to synthesize the stabilizing controllers [32]. However, despite their optimization-based formulation, they often fail to achieve the long-term optimal behavior. This deficiency arises due to the fact that the cost of these optimization problems fails to incorporate the future behavior of the system, being instead point-wise optimal [33]. In contrast, Nonlinear Model Predictive Control (NMPC) emphasizes the performance by solving an optimal control problem online, despite additional assumptions must be met to certify the closed loop stability. Thus, the integration of both these control methodologies has been extensively discussed in the literature to try to combine the benefits and to compensate for the drawbacks. Lyapunov methods have been used to construct stabilizing terminal conditions for NMPCs [34], or to analyze the stability in the absence thereof [35]. Another approach incorporates the stability condition required by a CLF along the prediction horizon found with a NMPC [36]. As noted in [36], this approach has several desirable properties, such as the absence of a terminal cost, and stability for any horizon length. In [37], the focus is on the practical and computational aspects of the unification of these two methodologies to control a robotic platform. The performed tests show that this combination leads to improved performance over CLF methods, significantly reducing the tuning of the prediction horizon length and the terminal conditions for NMPC methods. However, the integration reported so far only considers known simple dynamics, proposing algorithms that are not suitable for the human-robot interaction problem. In fact, many conventional MPC algorithms encounter problems related to the large computational burden caused by the high non-linearity of the considered system dynamics, requiring accurate models. An interesting implementation of an industrial controller based on MPC stabilized with Lyapunov constrains is described in [38]. The controller is applied on a continuous nonlinear system with complex dynamics. The accurate mathematical model of such system is hard to be obtained. To solve this issue, the modeling is realized by exploiting a Reinforcement Learning (RL) technique to approximate the solution of the Lyapunov MPC (LMPC). Specifically, among all the RL algorithms, Q-Learning is used, in which ANNs are exploited to approximate the agent and the Q-function to deal with a continuous state space. Thus, the computational efficiency of the control law is guaranteed regardless of the complexity of system dynamics.

The reviewed updating strategies for the online tuning of the impedance control parameters in the pHRC context are indeed characterized by some difficulties in the simultaneous optimization of the impedance parameters. The online optimization of the setpoint and the other impedance parameters is, in fact, important to obtain an active target-oriented and compliant behavior of the manipulator during pHRC task. In addition, it is commonly difficult to ensure stability in such optimization problem. Moreover, a reliable model of the target dynamics is not always available, making it difficult to optimize the collaboration. Besides, the most of the reviewed strategies are based on the internal simulation of the state evolution, exploiting the prediction capabilities of dynamic models or Q-functions. However, the former are often not enough accurate or efficient to be implemented for real applications execution, and the latter are usually approximated with a fuzzy logic, requiring a precise setting of the fuzzy rules and membership functions to solve “curse of dimensionality” problems.

1.3. Paper contribution

The aim of this paper is to design a variable impedance controller with guaranteed stability for pHRC applications. This controller should be capable of modeling the complex human-robot interaction dynamics, exploiting it for the online optimization of the low-level impedance control parameters. The employed optimization methodology, similarly to the strategy described in [20], is based on the resolution of an MPC. The MPC objective function is designed to minimize the user’s effort during the interaction with the manipulator, simulating the evolution of the system through an ensemble of artificial neural networks (ANNs). In fact, along with GP models (which, however, lose efficiency in high dimensional spaces [25]), an ensemble of ANNs represents an interesting choice to accurately model the unknown dynamics of the system, thus making it possible to capture the uncertainties of the prediction. The MPC is enhanced with stability guarantees by means of Lyapunov constraints. The MPC is then solved online by means of a Q-Learning method that uses an actor-critic algorithm to approximate the exact solution.

The proposed controller has been designed to fix the following three main open issues that are still present in the reviewed state of the art approaches (especially taking into consideration the controller developed in [20] by some of the authors):

(i) computational burden;
(ii) lack of demonstrated stability of the closed loop solution;
(iii) performance-dependence on the weights used for the cost function and for the setpoint update strategy.

(i) is tackled by avoiding to solve the MPC problem with a CEM algorithm at each control step. A Q-learning algorithm, as described in [38], is instead used. Based on the measured state of the manipulator, the Q-learning algorithm (which embeds two neural networks) returns an estimation of the optimal impedance parameters that would have been computed by the MPC, with much lower computational requirements. After an exploration phase, the Q-function learns the utility of all the state-action tuples, allowing the actor to choose the most desirable control policy according to the specified targets. During the training phase of an actor, a lighter CEM algorithm is still used to solve a receding horizon control problem and to update the actor network. In this way, the prediction capability of the ANN is exploited to estimate the state evolution, and to obtain the final result that is more accurate than the one obtained by just selecting the action associated with the highest Q-value. In fact, selecting the action with the highest Q-value, it will not take into account the model uncertainty (as it is done in the proposed implementation).

(ii) is solved by combining the MPC with a Lyapunov-based controller obtained with the point-wise mini-norm method in [33], which introduces the specified constraints at every control step. These two approaches are sub-optimal strategies to solve an optimization problem, and, under proper technical conditions, are equivalent, respectively, to the Euler-Lagrange optimization and to the solution of the Hamilton-Jacobi-Bellman equation [36]. Thus, the proposed approach, from a practical viewpoint, provides the guaranteed stability properties of the CLFs methods, together with the on-line optimization capabilities and performance of receding horizon controllers.

Since the Lyapunov-based point-wise mini-norm controller solves an inverse optimal control problem, it tunes the choice of the best control action not on the weights of the performance index, but on the Lyapunov function(s). In combination with the MPC, it leads to reduce the influence of the cost weights on the final control action [37], thus tackling (iii). This third issue is also addressed by considering the setpoint of the Cartesian impedance control and damping parameters as optimization variables (avoiding the need to manually tune the setpoint control law related gains, as shown in [20]).

Remark 1. The authors developed the proposed Q-LMPVIC working on a control-affine model of the robot system, assuming that the human’s interaction does not affect this property. The authors are confident that the operator does not modify the actuation law since the controller is designed to reduce his/her effort, and he/she is interacting with the robot at its end-effector only. This modeling choice is common in literature, whereas non-affine systems are considered just when the control is clearly non-linear. As a matter of example, this is considered for aircraft dynamics, underwater vehicles, active magnetic bearings, electromagnetic levitation, etc. [39].

The proposed controller has been validated through experimental tests, considering a pHRC application, in which a Franka EMIKA panda robot has been used as a test platform. Two scenarios have been considered: i) interaction between the human and the robot end-effector (*i.e.*, no tool at the robot end-effector); ii) interaction between the human and a sealing gun installed at the robot's end-effector. A total of 20 subjects (*i.e.*, 10 subjects for each scenario) have been involved in the experimental campaign. In both the scenarios, each user was asked to interact with the robot along the controlled z vertical direction. To assess its performance, the proposed controller has been compared with the model-based reinforcement learning variable impedance controller previously developed by some of the authors in [20]. The pHRC performance has been validated on the basis of the questionnaire described in [4,20]. Achieved results highlight the improved performance of proposed controller w.r.t. to the one in [20]. Furthermore, the training process of the proposed controller has been validated in order to verify its applicability in a real industrial context for a specific operator (*i.e.*, each user trains a separate controller). 5 additional users have been asked to perform their own training of the controller, to show the ease of the procedure for customizing the controller for each specific user. Achieved results show the effectiveness of the training process. Finally, the proposed approach has been applied to two complex use-cases (a collaborative assembly task and a collaborative deposition task) in order to show its applicability to real industrial tasks.

1.4. Paper layout

The paper is structured as follows. In Section 2, a detailed description of the control methodology is proposed. The low-level impedance controller is introduced in Section 2.1. Then, the pHRC dynamics is detailed in Section 2.2. The description of the adopted methodology for the online modeling and estimation of the pHRC dynamics is provided in Section 2.3. The Lyapunov-based MPC is detailed in Section 2.4. The Q-learning methodology is stated in Section 2.5, along with its actor-critic implementation in Section 2.6. The CEM algorithm is described in Section 2.7. Finally, the complete Q-LMPC control framework is shown in Section 2.8. Section 3 provides the achieved experimental results, demonstrating the improved pHRC performance w.r.t. the controller proposed in [20]. Finally, Section 4 contains the conclusions and the future work.

2. Methodology

The proposed Q-Learning-based Model Predictive Variable Impedance Control (Q-LMPVIC) for pHRC tasks is made up of two main levels. In the first one (*i.e.*, the low-level control loop), a variable Cartesian impedance controller is realized, such that the outer high-level controller could work considering the manipulator as a decoupled mass-spring-damper system in the Cartesian space. The outer high-level controller is then used to update the setpoint and the damping parameters of the inner controller in order to optimize the pHRC performance (*i.e.*, minimize the interaction force between the human and the robot, and, therefore, the operator's effort). The outer high-level controller is composed of an actor and a critic ANN, which implements a Q-Learning algorithm for the resolution of a nonlinear optimal control problem. An ensemble of ANNs is exploited to estimate the model of the system. An MPC enhanced with stability guarantees (by means of Lyapunov constraints) is implemented to online compute the low-level impedance control parameters, maximizing the pHRC performance.

Fig. 1 shows the proposed Q-LMPVIC schema, highlighting each element composing the proposed methodology. In the following, all the elements composing the Q-LMPVIC (*i.e.*, the low-level Cartesian impedance control, the human-robot interaction dynamics, the modeling estimation methodology based on the ensemble of ANNs, the L-MPC, the Q-learning methodology, its actor-critic ANNs, and the CEM algorithm) are described.

2.1. Cartesian impedance control

As described in [40], a Cartesian impedance controller can be designed to perform a compliant task, providing the reference to the inner position controller. On the basis of the interaction force acting on the manipulator, impedance control allows to calculate the robot accelerations $\ddot{\mathbf{x}}_{imp} = [\ddot{\mathbf{p}}; \ddot{\boldsymbol{\varphi}}_{cd}]$ (where $\ddot{\mathbf{p}}$ are related to the translational degrees of freedom - DoFs -, and $\ddot{\boldsymbol{\varphi}}_{cd}$ are related to the rotational DoFs described by the intrinsic Euler angles representation):

$$\begin{aligned} \ddot{\mathbf{p}} &= \mathbf{M}_t^{-1}\left(-\mathbf{D}_t \Delta\dot{\mathbf{p}} - \mathbf{K}_t\,\Delta\mathbf{p} - \mathbf{f}_t\right), \\ \ddot{\boldsymbol{\varphi}}_{cd} &= \mathbf{M}_{\varphi}^{-1}\left(-\mathbf{D}_{\varphi}\,\dot{\boldsymbol{\varphi}}_{cd} - \mathbf{K}_{\varphi}\,\boldsymbol{\varphi}_{cd} + \mathbf{S}_{\omega}^T(\boldsymbol{\varphi}_{cd})\,\boldsymbol{\tau}_{\varphi}\right). \end{aligned} \tag{1}$$

Considering the translational part of the impedance control, $\mathbf{M}_t$ is the target mass matrix, $\mathbf{D}_t$ is the target damping matrix, $\mathbf{K}_t$ is the target stiffness matrix, $\mathbf{f}_t$ is the external forces vector. $\mathbf{p}$ is the actual Cartesian positions vector, while $\Delta\mathbf{p} = \mathbf{p} - \mathbf{p}^d$ and $\Delta\dot{\mathbf{p}} = \dot{\mathbf{p}} - \dot{\mathbf{p}}^d$, where $\mathbf{p}^d$ is the target positions vector and $\dot{\mathbf{p}}^d$ is the target velocity vector. Considering the rotational part of the impedance control, $\mathbf{M}_{\varphi}$ is the target inertia matrix, $\mathbf{D}_{\varphi}$ is the target damping matrix, $\mathbf{K}_{\varphi}$ is the target stiffness matrix. $\boldsymbol{\varphi}_{cd}$ is the set of Euler angles extracted from $\mathbf{R}_c^d = \mathbf{R}_d^T\mathbf{R}_c$, describing the mutual orientation between the compliant frame $\mathbf{R}_c$ (coincident with the robot end-effector reference frame) and the target frame $\mathbf{R}_d$. $\boldsymbol{\tau}_{\varphi}$ is the external torques vector referred to the target frame. Matrix $\mathbf{S}_{\omega}(\boldsymbol{\varphi}_{cd})$ defines the transformation from the derivatives of Euler angles to angular velocities $\boldsymbol{\omega} = \mathbf{S}_{\omega}(\boldsymbol{\varphi}_{cd})\dot{\boldsymbol{\varphi}}_{cd}$ [41]. The six DoFs impedance control, therefore, results in:

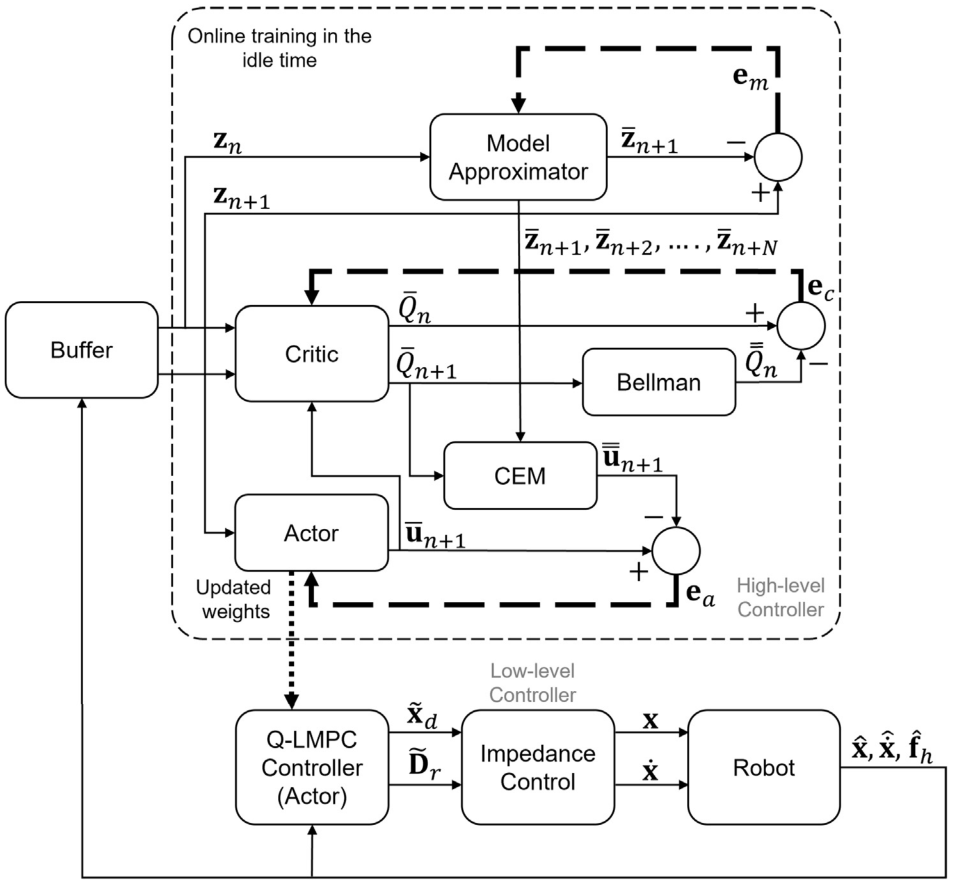


Fig. 1. Overall Q-LMPVIC scheme. The training operations inside the dashed square are performed once the buffer is full (as described in section 2.6), leading to the update of the Q-LMPC controller, that regularly sends the optimal setpoint and damping parameters to the impedance controller. The optimization variables (*i.e.*, the Cartesian impedance control setpoint $\tilde{\mathbf{x}}_d$ and damping $\tilde{\mathbf{D}}_r$) are used as the control input $\bar{\mathbf{u}}$, to be computed on the basis of the robot state $\mathbf{z}$.

$$\mathbf{M}_r\,\ddot{\mathbf{x}}_{Ci} + \mathbf{D}_r\,\Delta\dot{\mathbf{x}}_{Ci} + \mathbf{K}_r\,\Delta\mathbf{x}_{Ci} = \mathbf{f}_{ext}, \tag{2}$$

where $\mathbf{M}_r$, $\mathbf{D}_r$, $\mathbf{K}_r$ are the impedance diagonal matrices composed of both the translational and rotational parts, $\Delta\mathbf{x}_{Ci} = \mathbf{x}_{Ci} - \mathbf{x}^d = [\Delta\mathbf{p}; \boldsymbol{\varphi}_{cd}] = [\Delta\mathbf{x}_t; \Delta\mathbf{x}_r]$ (where $\mathbf{x}^d_{Ci}$ is the six DoFs position reference for the impedance controller), $\Delta\dot{\mathbf{x}}_{Ci} = \dot{\mathbf{x}}_{Ci} - \dot{\mathbf{x}}^d$, and $\mathbf{f}_{ext} = [\mathbf{f}_t; \mathbf{S}^T_\omega(\boldsymbol{\varphi}_{cd})\,\boldsymbol{\tau}_\varphi]$. It should be underlined that the damping matrix can be computed as follows: $\mathbf{D}_r = 2\boldsymbol{\zeta}_r\sqrt{\mathbf{K}_r\mathbf{M}_r}$, where $\boldsymbol{\zeta}_r$ is the damping ratio diagonal matrix.

2.2. pHRC model description

The proposed Q-LMPVIC is designed for pHRC tasks, in which the operator is always considered in contact with the manipulator whenever the motion is required. Thus, pHRC dynamics can be considered as a result of an interaction between two agents, the robot and the human, that can be described by the following model:

$$\mathbf{M}_{tot}\ddot{\mathbf{x}} + \mathbf{D}_{tot}\dot{\mathbf{x}} + \mathbf{K}_{tot}(\mathbf{x} - \mathbf{x}_d) = \mathbf{K}_h(\check{\mathbf{x}}_d - \mathbf{x}), \tag{3}$$

where (2) is modified introducing the operator's dynamics through additional unknown contributions $\mathbf{M}_h$ to the total mass $\mathbf{M}_r + \mathbf{M}_h = \mathbf{M}_{tot}$, additional unknown contribution $\mathbf{D}_h$ to the total damping $\mathbf{D}_r + \mathbf{D}_h = \mathbf{D}_{tot}$, and additional unknown contribution $\mathbf{K}_h$ to the total stiffness $\mathbf{K}_r + \mathbf{K}_h = \mathbf{K}_{tot}$. Considering that the operator wants to reach a certain target position $\check{\mathbf{x}}_d$, the external wrench $\mathbf{f}_{ext}$ (*i.e.*, the human applied wrench $\mathbf{f}_h$) is applied in proportion to the actual error $\check{\mathbf{x}}_d - \mathbf{x}$ and to the human stiffness $\mathbf{K}_h$.

As proposed in this paper, the high-level controller should optimize the impedance control setpoint $\tilde{\mathbf{x}}_d$ ($\tilde{\cdot}$ refers to variables object of optimization), such that it follows the operator's target position $\check{\mathbf{x}}_d$, and the impedance control damping $\tilde{\mathbf{D}}_r$, in order to achieve a smooth motion during the task execution. The stiffness parameter of the impedance control is not considered for optimization purposes. In fact, its optimization does not affect the achieved collaboration significantly (*i.e.*, the simultaneous optimization of the impedance control setpoint and stiffness is not necessary since they are multiplied in the robot dynamics (2)). Therefore, the final controlled system, ideally, should behave as follows:

$$\mathbf{M}_{tot}\ddot{\mathbf{x}} + (\tilde{\mathbf{D}}_r + \mathbf{D}_h)\dot{\mathbf{x}} + \mathbf{K}_{tot}(\mathbf{x} - \tilde{\mathbf{x}}_d) - \mathbf{f}_h = 0, \tag{4}$$

where $\mathbf{f}_h = \mathbf{K}_h\left(\check{\mathbf{x}}_d - \mathbf{x}\right)$.

The external force $\mathbf{f}_h$ applied by the human can be included in the system state. The state space representation can, indeed, be written as follows:

$$\begin{cases} \frac{\mathrm{d}\dot{\mathbf{x}}}{\mathrm{d}t} = \mathbf{M}_{tot}^{-1}\left[-(\tilde{\mathbf{D}}_r + \mathbf{D}_h)\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} - \mathbf{K}_{tot}(\mathbf{x} - \tilde{\mathbf{x}}_d) + \mathbf{f}_h\right], \\ \dot{\mathbf{x}} = \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}, \\ \dot{\mathbf{f}}_h = \mathbf{K}_h\left(\frac{\mathrm{d}\check{\mathbf{x}}_d}{\mathrm{d}t} - \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}\right). \end{cases} \tag{5}$$

For the sake of simplicity, since the Cartesian impedance control decouples the Cartesian degrees of freedom (DoFs), it will be considered a one DoF system in the following analysis. In addition, translational DoFs will be considered in the paper, due to the common management of such DoFs in pHRC w.r.t. rotational DoFs [9]. The one dimensional system, along a translational DoF, can be written as it follows:

$$\begin{cases} \frac{\mathrm{d}\dot{x}}{\mathrm{d}t} = -\frac{D_h}{M_{tot}}\frac{\mathrm{d}x}{\mathrm{d}t} - \frac{K_{tot}}{M_{tot}}x + \frac{f_h}{M_{tot}} + \frac{K_{tot}}{M_{tot}}\tilde{x}_d - \frac{\tilde{D}_r}{M_{tot}}\frac{\mathrm{d}x}{\mathrm{d}t}, \\ \dot{x} = \frac{\mathrm{d}x}{\mathrm{d}t}, \\ \dot{f}_h = K_h\left(\frac{\mathrm{d}\check{x}_d}{\mathrm{d}t} - \frac{\mathrm{d}x}{\mathrm{d}t}\right). \end{cases} \tag{6}$$

The above formulation can be re-written as:

$$\dot{\mathbf{z}}(t) = \mathbf{f}(\mathbf{z}(t)) + \mathbf{g}_1(\mathbf{z}(t))u_1(t) + \mathbf{g}_2(\mathbf{z}(t))u_2(t), \tag{7}$$

where:

$$\mathbf{z}(t) = [\dot{x}, x, f_h]^T \in Z \subset \mathbb{R}^3, \quad u_1(t) \in U_1, \quad u_2(t) \in U_2,$$
$$U_i := \{u_i \in \mathbb{R} \| u_i^{min} \le u_i \le u_i^{max}\},$$

in which the two optimization variables can be seen as the input $\mathbf{u}$ of the system, and the state $\mathbf{z}$ is represented by the robot position x, the robot velocity $\dot{x}$, and the external force f_h. Since the parameters describing the human dynamics are dependent on the characteristics of the interaction, varying during the task execution, the outer controller does not have access to a known dynamics modeling. Therefore, the functions $\mathbf{f}$, $\mathbf{g}_1$ and $\mathbf{g}_2$ are nonlinear and need to be estimated and updated periodically. In order to simplify the considered problem without losing any generality, it is possible to consider the target position $\check{x}_d$ as a piece-wise constant function. In fact, due to the reduced bandwidth of the human motion w.r.t. the one of the low-level robot controller (*i.e.*, few Hz against kHz, [20]), it is reasonable to consider $\frac{\mathrm{d}\check{x}_d}{\mathrm{d}t} = 0$. In addition, it is also important to underline that the considered system dynamics is control-affine. Besides, it can be assumed that $\mathbf{f}(\mathbf{z}(t))$ and $\mathbf{g}_i(\mathbf{z}(t))$ are smooth vector fields, and that the origin is an equilibrium point of the unforced nominal system, which implies that $\mathbf{f}(0) = 0$. Moreover, it is also assumed that the state $\mathbf{z}$ of the considered system is sampled synchronously w.r.t. the inputs (*i.e.*, the control variables), and the time instants at which the state measurement samplings are taken are indicated by the time sequence $t_k > 0$, with $t_k = t_0 + k\Delta, \quad k = 0, 1, 2, ...,$ where t_0 is the initial time and Δ is the sampling time.

2.3. Model estimation

To implement the proposed model-based reinforcement learning approach, the unknown dynamics of pHRC detailed in the previous Section needs to be modeled. To avoid introducing the relevant errors in this delicate modeling phase, it is better to avert to the use of simplified models [26,28]. A possible solution to this issue is, therefore, the adoption of Artificial Neural Networks (ANNs) for the modeling of the pHRC dynamics in an accurate and efficient way. However, ANNs tend to overfit in the regions of state space with less training data. To solve this issue, it is possible to use an ensemble of ANNs in order to capture the model uncertainties where less data is available [42]. In fact, ANNs agree with each other in the regions where more data is available, whereas they disagree with each other in the regions with less data. For this reason, the dynamics uncertainties are captured considering the whole ensemble for the prediction, and used in the CEM algorithm for online resolution of the MPC and the Lyapunov-based controller. Additionally, ANNs can easily scale with the data, thanks to the parallelization with modern GPUs. This functionality is not exploited in the present work, since one of the objectives of this paper is to reduce the computational burden in order to make the algorithm feasible for implementation in pHRC tasks also with a moderate computational power.

The following procedure has been implemented in order to train and update the ensemble of ANNs:

Table 1
Parameters used for the CEM and for the ANNs.

Parameter	Value
Hardware configuration	1 NVIDIA GPU + 12 CPU cores
Training buffer size	5
Model Approximators	
Number of hidden layers	5
Number of hidden units	512
Size of ensemble	2
Activation function	ReLU
Learning rate	1e-3
Actor	
Optimizer	Adam
Number of hidden layers	3
Number of hidden units	64
Activation function	ReLU
Learning rate	1e-4 - 5e-5
Critic	
Optimizer	SGD
Number of hidden layers	3
Number of hidden units	128
Activation function	ReLU
Learning rate	1e-3
CEM	
Discount factor γ	0.9
No. of samples	16
No. of iterations	3
No. of elites	4
Prediction horizon	7

1. The parameters for the network architecture are taken from [42] whereas the ensemble size is found by testing multiple values from 2 to 5 in simulation and experimental tests [20]. The parameters used for the networks are listed in Table 1.
2. The networks learn the target dynamics $\breve{\mathbf{f}}(\mathbf{z}_t, \mathbf{u}_t)$ by predicting the change in the state over the time step duration $\triangle$:
$$\mathbf{z}_{t+t\triangle} = \mathbf{z}_t + \breve{\mathbf{f}}(\mathbf{z}_t, \mathbf{u}_t). \tag{8}$$
This choice is made due to the fact that control input has a little effect on output when two successive states $\mathbf{z}_t$ and $\mathbf{z}_{t+t\triangle}$ are very similar [43].
3. The inputs and outputs are normalized using mean and standard deviation. The initial values of mean and standard deviation for each variable are roughly estimated from the knowledge of the target task. During the first run on the actual robot, these variables are measured to update the previous estimation. Data normalization highly impacts the way in which the neural network interprets the different states of the manipulator, and it is crucial to obtain a good performance, reducing also the distribution shift between the training and the interaction samples.
4. The ensemble of ANNs is updated periodically to adapt to the new human physiological parameters, using measured data during the task execution. In such a way, the adaptability of the approach is improved when the operator changes his/her collaboration modalities with the robot (*i.e.*, when he/she changes the target task, or gets tired). In particular, a period can be defined during which the actual state $\hat{\mathbf{z}} = [\dot{x}, x, f_h]$ is measured and stored in a buffer. At the end of this period, the values are compared with the output of the neural network $\bar{\mathbf{z}}$:
$$\mathbf{e}_m = \hat{\mathbf{z}}(t_k) - \bar{\mathbf{z}}(t_k), \quad E_m = \frac{1}{2}\left\|\mathbf{e}_m\right\|^2. \tag{9}$$
A stochastic gradient descend method is then used to update the layer weights.

Thus, the proposed ANNs are able to provide a model function $\breve{\mathbf{f}}(\mathbf{z}_t, \mathbf{u}_t)$ that represents a system that is control-affine:
$$\breve{\mathbf{f}}(\mathbf{z}_t, \mathbf{u}_t) = \mathbf{f}(\mathbf{z}_t) + \mathbf{g}_1(\mathbf{z}_t)u_{1_t} + \mathbf{g}_2(\mathbf{z}_t)u_{2_t}. \tag{10}$$
However, the general model function is not enough for the definition of the Lyapunov-based controller (adopted to stabilize the MPC). Indeed, it is important to obtain the specific functions $\mathbf{f}$, $\mathbf{g}_1$, and $\mathbf{g}_2$ at every instant. $\mathbf{f}$ can be found evaluating $\breve{\mathbf{f}}(\mathbf{z}_t, \mathbf{u}_t)$ when both the inputs are null:
$$\breve{\mathbf{f}}(\mathbf{z}_t, 0, 0) = \mathbf{f}(\mathbf{z}_t) + \mathbf{g}_1(\mathbf{z}_t)0 + \mathbf{g}_2(\mathbf{z}_t)0 = \mathbf{f}(\mathbf{z}_t). \tag{11}$$

Function $\mathbf{g}_1$ can be obtained by subtracting $\mathbf{f}$ from $\breve{\mathbf{f}}(\mathbf{z}_t, \mathbf{u}_t)$, evaluated when $u_1 = 1$ $u_2 = 0$:

$$\breve{\mathbf{f}}(\mathbf{z}_t, 1, 0) - \mathbf{f}(\mathbf{z}_t) = \mathbf{f}(\mathbf{z}_t) + \mathbf{g}_1(\mathbf{z}_t)1 + \mathbf{g}_2(\mathbf{z}_t)0 - \mathbf{f}(\mathbf{z}_t) = \mathbf{g}_1(\mathbf{z}_t). \tag{12}$$

Function $\mathbf{g}_2$ can be obtained by subtracting $\mathbf{f}$ from $\breve{\mathbf{f}}(\mathbf{z}_t, \mathbf{u}_t)$, evaluated when $u_1 = 0$ $u_2 = 1$:

$$\breve{\mathbf{f}}(\mathbf{z}_t, 0, 1) - \mathbf{f}(\mathbf{z}_t) = \mathbf{f}(\mathbf{z}_t) + \mathbf{g}_1(\mathbf{z}_t)0 + \mathbf{g}_2(\mathbf{z}_t)1 - \mathbf{f}(\mathbf{z}_t) = \mathbf{g}_2(\mathbf{z}_t). \tag{13}$$

These functions $\mathbf{f}$, $\mathbf{g}_1$, and $\mathbf{g}_2$ will be used in the next Sections to guarantee the stability of the controlled system.

2.4. Lyapunov-based model predictive control

The MPC is an online control methodology in which, at each time step t_k, the actual state $\mathbf{z}(t_k)$ of the system to be controlled is sampled and used as the initial state of an optimal control problem. The latter is solved in open loop for a specified time horizon $T = [t_k,\ t_{k+1},\ t_{k+2},\ \dots,\ t_{k+N}]$, and a series of optimal control actions $[\mathbf{u}_k,\ \mathbf{u}_{k+1},\ \mathbf{u}_{k+2},\ \dots,\ \mathbf{u}_{k+N}]$ is returned as an output. Only the first term of the series is the input (*i.e.*, control action) to be applied in the next time step to the controlled system, after which the system evolves and the new measured state is the starting point of the next optimization problem. While the controller must work for an indefinite time, the defined time horizon is limited to T. Moreover, the controller is subjected to constraints on the input, so, the open loop behavior evaluated at each step could differ from the closed loop one (even in the case the system model is perfect). In addition, instability might occur. The stability and the feasibility of the solution of the optimal control problem can be enforced introducing a proper terminal cost and a terminal constraint. Other approaches can also be found in the literature, in particular considering Lyapunov-based Model Predictive Controllers, where stability and robustness properties are inherited by the Lyapunov-based controller, combining it with the online optimization and performance properties of the receding horizon method [44,45,37,36,38]. Thus, a sub-optimal strategy providing excellent results is obtained from the resolution of a reduced complexity optimization problem.

Considering the previously defined nonlinear continuous-time system in (7), the optimal control problem can be solved in an open-loop/Euler-Lagrange fashion, adopting a receding horizon control technique to produce a state feedback control law. However, to approach the optimal control problem in this framework, the original objective function must be adapted, with a consequent modification of the problem as it follows:

$$\int_0^\infty r(\cdot)d\tau \neq \int_{t_k}^{t_{k+N}} r(\cdot)d\tau + \varphi(\mathbf{z}(t_{k+N})) \quad if\ \varphi(\cdot) \neq V^*(\cdot), \tag{14}$$

where $V^*(\cdot)$ represents the value function obtained from the optimization of the original cost functional, $r(\cdot)$ is the instantaneous cost function, and $\varphi(\cdot)$ is the terminal weight. Moreover, the solution here obtained is only locally optimal. To solve these issues, without complex computations, by referring to [36], it is possible to generate an inverse optimal control law $\mathbf{h}$, given a properly chosen CLF V_L, through the point-wise minimization of the control effort:

$$\min_{\mathbf{h}} \mathbf{h}^T\mathbf{h} + q(\mathbf{z}), \tag{15}$$

$$\text{s.t.} \quad \frac{\partial V_L(\mathbf{z})}{\partial \mathbf{z}}(\ \mathbf{f}(\mathbf{z}) + \mathbf{g}_1(\mathbf{z})h_1 + \mathbf{g}_2(\mathbf{z})h_2\ ) \leq -\sigma(\mathbf{z}), \tag{16}$$

where $\sigma(\mathbf{z})$ is a continuous positive definite function, and $q : \mathbb{R}^3 \to \mathbb{R}$ is a continuously differentiable function such that $q(\mathbf{0}) = 0$ and $q(\mathbf{z}) > 0$ when $\mathbf{z} \neq 0$. Since the optimization is limited to the current time step, state $\mathbf{z}$ is already known and only the control action $\mathbf{h}$ has to be imposed. Thus, the presence of the term $q(\mathbf{z})$ is pointless for the optimization purpose. The resulting Lyapunov-based controller, under proper technical conditions [36], is equivalent to the globally optimal solution of the Hamilton-Jacobi-Bellman equation. Extending this point-wise controller to a receding horizon methodology, the objective function should also account for the state $\mathbf{z}$, while the constraint (16) should be extended at all the time steps inside the horizon. Thus, the optimization problem becomes:

$$\min_{\mathbf{u}\in S(\Delta)} \int_{t_k}^{t_{k+N}} r(\bar{\mathbf{z}}(\tau), \mathbf{u}(\tau))d\tau \quad \mathbf{u} \in U, \tag{17}$$

$$\text{s.t.} \quad \dot{\bar{\mathbf{z}}}(t) = \mathbf{f}(\bar{\mathbf{z}}(t)) + \mathbf{g}_1(\bar{\mathbf{z}}(t))u_1(t) + \mathbf{g}_2(\bar{\mathbf{z}}(t))u_2(t), \tag{18}$$

$$\bar{\mathbf{z}}(t_k) = \hat{\mathbf{z}}(t_k), \tag{19}$$

$$\frac{\partial V_L(\bar{\mathbf{z}})}{\partial \bar{\mathbf{z}}}(\ \mathbf{f}(\bar{\mathbf{z}}) + \mathbf{g}_1(\bar{\mathbf{z}})u_1 + \mathbf{g}_2(\bar{\mathbf{z}})u_2\ ) \leq -\sigma(\bar{\mathbf{z}}), \tag{20}$$

where $S(\Delta)$ is the family of piece-wise constant functions with sampling period Δ, N is the prediction horizon, $r(\cdot)$ is the cost function, that for this problem can be set as $f_h {f_h}^T$, $\bar{z}(t)$ is the predicted trajectory in open loop, and $\hat{z}(t)$ is the state measured at time t_k. The proposed methodology can be interpreted as a conceptual blend of HJB and Euler–Lagrange philosophies, that for $N \to 0$ becomes equivalent to the point-wise mini norm problem, and for $N \to \infty$ goes back to the original optimal control problem. In any case, for all the finite N, a stable solution can be chosen, and, if the conditions in [36] are respected, the optimal controller is recovered even for the case of a finite time horizon.

Once the controller $\mathbf{h}$ is computed, it is better to rearrange the constraint (20) in a way in which the associated limits on the control action clearly emerge:

$$\frac{\partial V_L(\mathbf{z})}{\partial \mathbf{z}}(\mathbf{f}(\mathbf{z}) + \mathbf{g}_1(\mathbf{z})u_1 + \mathbf{g}_2(\mathbf{z})u_2) \le \frac{\partial V_L(\mathbf{z})}{\partial \mathbf{z}}(\mathbf{f}(\mathbf{z}) + \mathbf{g}_1(\mathbf{z})h_1 + \mathbf{g}_2(\mathbf{z})h_2). \tag{21}$$

The existence of a controller $\mathbf{h}(\mathbf{z}) = [h_1(\mathbf{z}), h_2(\mathbf{z})]^T$ that provides the stability of an equilibrium point (*i.e.*, the origin), while satisfying the system input constraints for all the states inside a given stability region, is not generally guaranteed, and it is associated to the property of stabilizability of the system [46]. According to Artstein's theorem [47], a dynamical system has a differentiable Control Lyapunov Function (CLF) if and only if there exists a regular global asymptotically stabilizing feedback. Thus, given a set of class K functions $\alpha_i(\cdot),\ i = 1, 2, 3, 4$, and a CLF $V_L(\mathbf{z})$, the inverse optimal control problem can be surely solved.

Definition 2.1. A class K function is a continuous function $\alpha : [0; a) \to R_+$, with $a > 0,\ \alpha(0) = 0$ and α strictly monotonically increasing (denoted $\alpha \in K$). If $a = \infty$ and $\lim_{r\to\infty} \alpha(r) = \infty$, then α is said to be a class K_∞ function [37].

Definition 2.2. A CLF is a continuously differentiable function $V_L(\mathbf{z})$ for which the following inequalities are valid [48]:

$$\alpha_1(|\mathbf{z}|) \le V_L(\mathbf{z}) \le \alpha_2(|\mathbf{z}|), \tag{22}$$

$$\inf_{\mathbf{h}} \frac{\partial V_L(\mathbf{z})}{\partial \mathbf{z}}(\ \mathbf{f}(\mathbf{z}) + \mathbf{g}_1(\mathbf{z})h_1 + \mathbf{g}_2(\mathbf{z})h_2\ ) \le -\alpha_3(|\mathbf{z}|), \tag{23}$$

$$\left|\frac{\partial V_L(\mathbf{z})}{\partial \mathbf{z}}\right| \le \alpha_4(|\mathbf{z}|), \tag{24}$$

for all $\mathbf{z} \in O_z \subseteq \mathbb{R}^2$, where O_z is an open neighborhood of the origin. We will say that V_L is a weak CLF when the inequality (23) is non-strict, namely, when:

$$\inf_{\mathbf{h}} \frac{\partial V_L(\mathbf{z})}{\partial \mathbf{z}}(\ \mathbf{f}(\mathbf{z}) + \mathbf{g}_1(\mathbf{z})h_1 + \mathbf{g}_2(\mathbf{z})h_2\ ) \le 0. \tag{25}$$

The existence of a weak CLF is not contemplated in Artstein's theorem, and it doesn't guarantee the global stabilizability, as it does with the existence of a CLF. Nevertheless, in many cases, a weak CLF can, indeed, be used to design a globally stabilizing control law, which is the case of the present work. To define the inverse optimal control problem specified by (15)-(16), it is necessary to choose a CLF V_L and a function $\sigma(\mathbf{z})$. The latter can be selected in order to obtain the point-wise mini-norm Sontag controller [49], which provides a smooth, real and analytic feedback stabilizer for a generic affine system. Denoting the Lie derivatives of V_L as:

$$a(\mathbf{z}) := \nabla V_L(\mathbf{z}) \cdot \mathbf{f}(\mathbf{z}), \tag{26}$$

$$b_1(\mathbf{z}) := \nabla V_L(\mathbf{z}) \cdot \mathbf{g}_1(\mathbf{z}), \tag{27}$$

$$b_2(\mathbf{z}) := \nabla V_L(\mathbf{z}) \cdot \mathbf{g}_2(\mathbf{z}), \tag{28}$$

$$\beta(\mathbf{z}) = \sum_{i=1}^{2} b_i^2(\mathbf{z}). \tag{29}$$

$\sigma(\mathbf{z})$ has to be chosen as:

$$\sigma_s(\mathbf{z}) = \sqrt{(a^2 + q(\beta)\beta^2}, \tag{30}$$

to yield the resulting control law:

$$\begin{cases} h_i = -b_i \frac{a + \sqrt{a^2 + \beta^2\ q(\beta)}}{\beta} & \text{for } \beta \ne 0, \\ h_i = 0 & \text{for } \beta = 0. \end{cases} \tag{31}$$

If, at this point, the CLF V_L is chosen to have the same shape lines of the optimal value function V^*, all the technical conditions to obtain an inverse optimal controller are satisfied. Thus, V_L corresponds to the solution of the HJB equation, and it is globally optimal [50]. The most obvious candidate for the CLF is a quadratic function, which includes the fundamental variables to measure the distance from the researched point. So, it can be defined as:

$$V_L = \frac{1}{2}\dot{x}P_1\dot{x}^T + \frac{1}{2}f_h P_2 f_h^T, \tag{32}$$

where P_1 and P_2 are the weights used to select the significance of each term. For this choice, the previously defined Lie derivatives can be expressed as:

$$\begin{aligned} a(\mathbf{z}) &:= \nabla V_L(\mathbf{z}) \cdot \mathbf{f}(\mathbf{z}) = P_1\dot{x}\cdot \mathbf{f}[1] + P_2\, f_h \cdot \mathbf{f}[3] \\ &= P_1\dot{x}\cdot\left(-\frac{D_h}{M_{tot}}\dot{x} - \frac{K_{tot}}{M_{tot}}x + \frac{f_h}{M_{tot}}\right) - P_2\, f_h \cdot K_h\dot{x}, \end{aligned} \tag{33}$$

$$\begin{aligned} b_1(\mathbf{z}) &:= \nabla V_L(\mathbf{z}) \cdot \mathbf{g}_1(\mathbf{z}) = P_1\dot{x}\cdot \mathbf{g}_1[1] + P_2\, f_h \cdot \mathbf{g}_1[3] \\ &= P_1\dot{x}\cdot\frac{K_{tot}}{M_{tot}}, \end{aligned} \tag{34}$$

$$\begin{aligned} b_2(\mathbf{z}) &:= \nabla V_L(\mathbf{z}) \cdot \mathbf{g}_2(\mathbf{z}) = P_1\dot{x}\cdot \mathbf{g}_2[1] + P_2\, f_h \cdot \mathbf{g}_2[3] \\ &= P_1\dot{x}\cdot\frac{-\dot{x}}{M_{tot}}, \end{aligned} \tag{35}$$

$$\beta(\mathbf{z}) := \sum_{i=1}^{2} b_i^2(\mathbf{z}) = P_1^2\dot{x}^2\cdot\left(\frac{K_{tot}}{M_{tot}}^2 + \frac{\dot{x}}{M_{tot}}^2\right). \tag{36}$$

However, in such a way, the resultant control law $\mathbf{h}$ is not continuous. In fact, for any shape of $q(\beta)$, that can be for example set to β^2 as in [49], the system does not respect the small control property. Thus, continuity cannot be proven in general, and it becomes necessary to look at the conditions in which $\beta \to 0$ (*i.e.*, $\dot{x} \to 0$) to understand how the stabilizer behaves. Considering the unknown human parameters as constant, for $\mathbf{z} \to \mathbf{0}$, it can be noticed that:

$$\begin{aligned} &a(\mathbf{z}) \to 0 \quad \text{as } z_i^2, \quad b_1(\mathbf{z}) \to 0 \quad \text{as } z_i, \\ &b_2(\mathbf{z}) \to 0 \quad \text{as } z_i^2, \quad \beta(\mathbf{z}) \to 0 \quad \text{as } z_i^2, \end{aligned}$$

thus:

$$\begin{aligned} h_1(\mathbf{z}) &= -b_1\frac{a+\sqrt{a^2+\beta^4}}{\beta} \to 0 \quad \text{as } z_i, \\ h_2(\mathbf{z}) &= -b_2\frac{a+\sqrt{a^2+\beta^4}}{\beta} \to 0 \quad \text{as } z_i^2. \end{aligned}$$

Indeed, with this assumption, the Lyapunov-based controller results continuous for $\mathbf{z} = \mathbf{0}$. However, it is known that the human impedance parameters M_h, D_h, K_h are actually nonlinear functions of the state and, to make the previous statement valid, it is necessary that $\frac{D_h}{M_h}$, $\frac{K_{tot}}{M_h}$ and $\frac{1}{M_h}$ do not diverge to infinity when $\mathbf{z} \to \mathbf{0}$, or at least diverge slower than $\frac{1}{z_i}$. Since for the considered HRC scenario the latter assumption is reasonable, the controller can still be considered continuous in this point. Instead, when $\dot{x} \to 0$, therefore $\beta \to 0$, but x, $f_h \neq 0$, assuming again that impedance parameters do not vary too much, and considering that $\breve{x}_d = 0 \Rightarrow f_h = -K_h x$ and $P_2 \gg P_1$ (as will be actually implemented), it can be noticed that:

$\dot{x} \to 0^+ \quad x < 0$	$\dot{x} \to 0^+ \quad x > 0$
$a(\mathbf{z}) \to 0^-$ as $\dot{x}$	$a(\mathbf{z}) \to 0^+$ as $\dot{x}$
$b_1(\mathbf{z}) \to 0^+$ as $\dot{x}$	$b_1(\mathbf{z}) \to 0^+$ as $\dot{x}$
$b_2(\mathbf{z}) \to 0^-$ as $\dot{x}^2$	$b_2(\mathbf{z}) \to 0^-$ as $\dot{x}^2$
$\beta(\mathbf{z}) \to 0^+$ as $\dot{x}^2$	$\beta(\mathbf{z}) \to 0^+$ as $\dot{x}^2$
$\dot{x} \to 0^- \quad x < 0$	$\dot{x} \to 0^- \quad x > 0$
$a(\mathbf{z}) \to 0^+$ as $\dot{x}$	$a(\mathbf{z}) \to 0^-$ as $\dot{x}$
$b_1(\mathbf{z}) \to 0^-$ as $\dot{x}$	$b_1(\mathbf{z}) \to 0^-$ as $\dot{x}$
$b_2(\mathbf{z}) \to 0^-$ as $\dot{x}^2$	$b_2(\mathbf{z}) \to 0^-$ as $\dot{x}^2$
$\beta(\mathbf{z}) \to 0^+$ as $\dot{x}^2$	$\beta(\mathbf{z}) \to 0^+$ as $\dot{x}^2$

$\dot{x} \to 0^+ \quad x < 0$
$h_1(\mathbf{z}) = -b_1 \frac{a+\sqrt{a^2+\beta^4}}{\beta} \to 0$ quicker than $\dot{x}$ $h_2(\mathbf{z}) = -b_2 \frac{a+\sqrt{a^2+\beta^4}}{\beta} \to 0$ quicker than $\dot{x}$
$\dot{x} \to 0^+ \quad x > 0$
$h_1(\mathbf{z}) = -b_1 \frac{a+\sqrt{a^2+\beta^4}}{\beta} \to -c$ $h_2(\mathbf{z}) = -b_2 \frac{a+\sqrt{a^2+\beta^4}}{\beta} \to 0$ as $\dot{x}$
$\dot{x} \to 0^- \quad x < 0$
$h_1(\mathbf{z}) = -b_1 \frac{a+\sqrt{a^2+\beta^4}}{\beta} \to c$ $h_2(\mathbf{z}) = -b_2 \frac{a+\sqrt{a^2+\beta^4}}{\beta} \to 0$ as $\dot{x}$
$\dot{x} \to 0^- \quad x > 0$
$h_1(\mathbf{z}) = -b_1 \frac{a+\sqrt{a^2+\beta^4}}{\beta} \to 0$ quicker than $\dot{x}$ $h_2(\mathbf{z}) = -b_2 \frac{a+\sqrt{a^2+\beta^4}}{\beta} \to 0$ quicker than $\dot{x}$

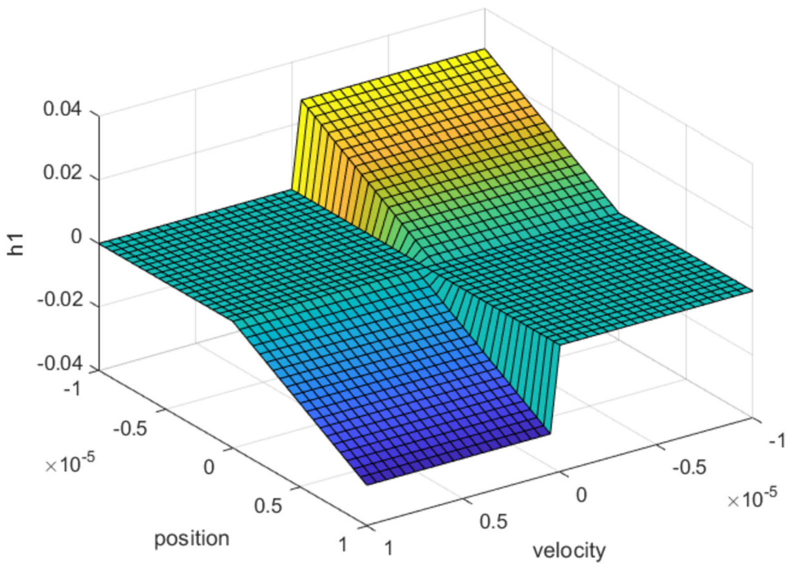


Fig. 2. First component of the Lyapunov-based controller h_1, found for constant values of the human parameters. The Figure underlines the discontinuity of the control action for $\dot{x} = 0$, with the exception of the origin.

c is a finite value. For this reason, in these points, the Lyapunov-based controller is not continuous. However, it remains piece-wise continuous, still inside the family of the feasible input actions for the proposed system. Fig. 2 and Fig. 3 shows the two components of the Sontag's controller, considering feasible values of the human parameters [51].

Considering the above analysis, the resulting closed loop system obtained from the application of the Sontag's stabilizer shows a non-smooth dynamics, and it cannot be handled with classical existence theorems for ordinary differential equations, requiring vector fields which are at least Lipschitz continuous. Instead, to analyze this system, a reference to the work of Filippov [52] needs to be made, who developed a solution concept for differential equations where right-hand sides were only required to be Lebesgue measurable in the state and time variables.

Definition 2.3 *(Filippov).* Considering a vector differential equation $\dot{x} = f(x,t)$ measurable and locally bounded, a vector function $x(\cdot)$ is called a solution on $[t_0,\ t_1]$ if $x(\cdot)$ is absolutely continuous on $[t_0,\ t_1]$ and for almost all $t \in [t_0,\ t_1]$:

$$\dot{x} \in K[f](x,t), \tag{37}$$

where:

$$K[f](x,t) = \bigcap_{\delta>0} \bigcap_{\mu N=0} \bar{co} f\{B(x,\delta) \setminus N\}, \tag{38}$$

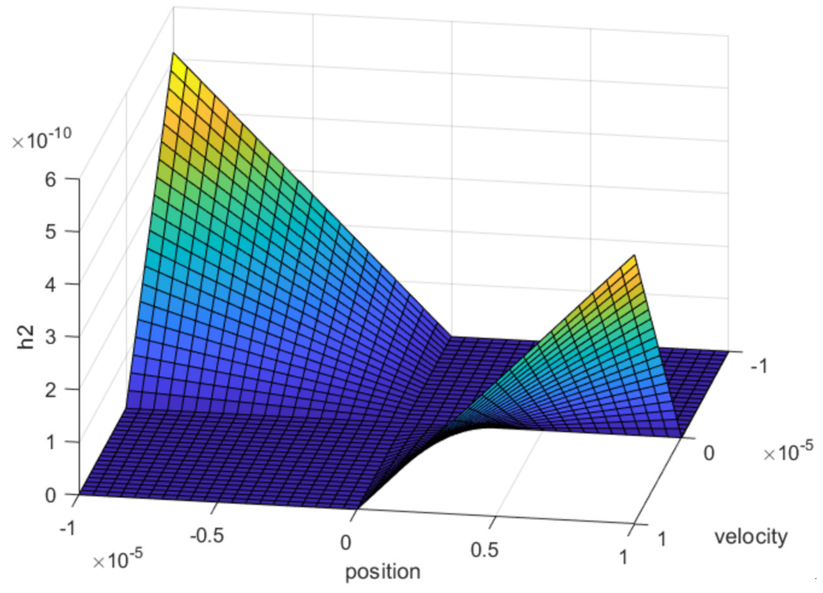


Fig. 3. Second component of the Lyapunov-based controller h_2, found for constant values of the human parameters.

$\bigcap_{\mu N=0}$ denotes the intersection over all sets N of Lebesgue measure zero, $\bar{co}$ represents the convex closure of a set, while $B(x,\delta)$ is the ball of radius δ centered at x.

$K[f](x,t)$ is a multivalued map able to associate multiple values to the same couple (x,t). Through its introduction, the above differential equation is generalized in a differential inclusion, and it becomes conceptually possible to deal with the discontinuous $\dot{x}$. In the above definition, it is important that the sets of measure zero are discarded. This technical detail allows the obtained solutions to be defined at points even where the vector field itself is not defined, *e.g.* at the interface of two regions in a piece wise defined vector field.

Theorem 2.1 *(Chain rule). Let $x(\cdot)$ be a Filippov solution to $\dot{x} = f(x,t)$ on an interval containing t and $V : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ be a smooth function, generally dependent on time. Then $V(x(t),t)$ is absolutely continuous and:*

$$\frac{\mathrm{d}}{\mathrm{d}t} V(x(t),t) \in \dot{\tilde{V}}(x,t), \tag{39}$$

where:

$$\dot{\tilde{V}}(x,t) = \begin{bmatrix} \nabla V_x \\ \partial V_t \end{bmatrix} \begin{bmatrix} K[f](x,t) \\ 1 \end{bmatrix}. \tag{40}$$

∇V_x represents the gradient of function V respect to vector x, while ∂V_t stands for the partial derivative of V respect to t.

The demonstration is provided in [53]. The definition of the set value map $\dot{\tilde{V}}(x,t)$ is necessary to study the equilibria of differential inclusions through the generalized Lyapunov criterion, defined and demonstrated in [54].

Theorem 2.2 *(Lyapunov stability). Let $\dot{x} = f(x,t)$ be essentially locally bounded and $0 \in K[f](x,t)$ in a region $D \supset \{x \in \mathbb{R}^n \parallel x \parallel < \delta\} \times \{t | t_0 \leq t < \infty\}$. Also, let $V : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ be a regular function satisfying:*

$$V(0,t) = 0, \tag{41}$$

$$0 < \alpha_1(\parallel x \parallel) \leq V(x,t) \leq \alpha_2(\parallel x \parallel) \quad for\ x \neq 0, \tag{42}$$

in D for some $\alpha_1, \alpha_2 \in$ class K. Then:

1. *$\dot{\tilde{V}}(x,t) \leq 0$ in D implies $x(t) \equiv 0$ is a uniformly stable solution and V is a weak CLF;*
2. *if in addition, there is a class K function α_3 in D with the property:*

$$\dot{\tilde{V}}(x,t) \leq -\alpha_3(\parallel x \parallel) < 0, \tag{43}$$

then the solution $x(t) \equiv 0$ is uniformly asymptotically stable and V is a CLF.

For the purposes of the present work, the set $\dot{\tilde{V}}_L(x,t) = \nabla V_L(\mathbf{z}) \cdot (\mathbf{f}(\mathbf{z}) + \mathbf{g}_1(\mathbf{z})h_1 + \mathbf{g}_2(\mathbf{z})h_2)$ is defined as:

$$\dot{\tilde{V}}_L(x,t) = \begin{cases} a - a - \sqrt{a^2 + \beta^2\, q(\beta)} < 0 & \text{for } \beta \neq 0, \\ a = 0 & \text{for } \beta = 0, \end{cases} \tag{44}$$

therefore, the chosen V_L is only a weak CLF and it doesn't guarantee the existence of a stabilizing controller. Nevertheless, the solution can be still accepted, proceeding with the inverse optimal design, since the global asymptotic stability can be proven from an extension of the LaSalle's theorem, originally demonstrated only for continuous differential equations.

Theorem 2.3 *(LaSalle). Let Ω be a compact set such that every Filippov solution to the autonomous system $\dot{x} = f(x)$, $x(0) = x(t_0)$ starting in Ω is unique and remains in Ω for all $t \geq t_0$. Let $V : \Omega \rightarrow \mathbb{R}$ be a regular function such that $v \leq 0$ for all $v \in \dot{\tilde{V}}$. Define $\Xi = \{x \in \Omega | 0 \in \dot{\tilde{V}}\}$. Then, every trajectory in Ω converges to the largest invariant set M in the closure of Ξ.*

Proof. Let $\mathbf{z}(t)$ be a Filippov solution of the considered system starting in Ω. Since $V_L(\mathbf{z}(t))$ is absolutely continuous, $\dot{\tilde{V}}_L$ is bounded below zero and V_L is bounded above zero, $V_L(\mathbf{z})$ tends to a constant a as $t \rightarrow \infty$. Uniqueness of the trajectories implies continuous dependence on initial conditions, so that the positive limit set L^+ of $\mathbf{z}(t)$ is an invariant set. Moreover, $L^+ \subset \Omega$ because Ω is closed. By the continuity of V_L, $V_L(p) = a$ for all $p \in L^+$. Since L^+ is invariant, $0 \in \dot{\tilde{V}}_L$ in L^+. Given that all the solutions inside L^+ are absolutely continuous and characterized by $\frac{\mathrm{d}}{\mathrm{d}t} V_L = 0$, it follows that $L^+ \subset \Xi$. Since L^+ is contained in the largest invariant set in Ξ, the theorem is proved, and these solutions are attracted towards M, that is the largest invariant set for which $\dot{x} = 0$ [53]. $\square$

Now, if in this set M there are no trajectories consistent with the differential inclusion $K[f](\mathbf{z},t)$, solutions different from $\mathbf{z} \equiv \mathbf{0}$ cannot remain in M, and this equilibrium position can be considered asymptotically stable.

$$M \cap K[f](\mathbf{m}) = \varnothing \quad \forall \mathbf{m} \neq \mathbf{0} \in M. \tag{45}$$

In the considered system, M is an invariant subset characterized by $\dot{x} = 0$, and general x and f_h, whereas $K[f](\mathbf{m})$ is obtained evaluating $K[f](\mathbf{z},t)$ at these points:

$$M = Span \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}, \quad K[f](\mathbf{m}) = \begin{bmatrix} -\frac{K_{tot}+K_h}{M_{tot}} \\ 0 \\ 0 \end{bmatrix}. \tag{46}$$

Since $\mathbf{z} = \mathbf{0}$ is the only point of intersection of these two sets, it is possible to conclude that the system remains in M only for $\mathbf{z} \equiv \mathbf{0}$. Thus, the designed Lyapunov-based controller $\mathbf{h}$ is able to asymptotically stabilize the system and to attract the state to the equilibrium point, as graphically shown in Fig. 4. However, the latter doesn't guarantee the stability when it is implemented in a zero-order hold fashion. In this case, the result is a function of the sampling period Δ and of the weights P_1 and P_2. At this point, there is the possibility to replace the receding horizon constraint derived in the continuous-time implementation (21) with its variant adapted for a sampled system. However, this substitution leads to a degradation of the performance [37], thus, we will keep the constraint (21) in the MPC to inherit the properties of the Lyapunov-based controller.

2.5. Q-Learning

Reinforcement Learning (RL) refers to the actor (or agent) that modifies its action based on the stimuli received by interacting with the environment in order to maximize the notion of cumulative reward [55]. It is an efficient technique to realize optimal control for systems with high complexity, such as industrial process control and chemical process control [55–57]. Among the RL algorithms, Q-learning is a typical method developed by Watkins and Dayan [58], known as action-dependent heuristic dynamic programming. One of the advantages of Q-learning is that it can obtain the optimal control policy directly by minimizing the Q-function relying on the information of system states and inputs without any prior knowledge of the system dynamics. An example of RL algorithm used to approximate a quasi-infinite-horizon nonlinear MPC for a discrete-time nonlinear system is developed in [59]. Based on that, in [38] a predictive controller for nonlinear continuous-time systems is designed, taking into consideration pHRC and its modeling for control purposes, as in the here presented method. Due to the presence of several sources of disturbance and due to the large amount of unmodeled variables, the pHRC system to be controlled is modeled as a Markov decision process (MDP). The system is defined as a 4-tuple $\eta(X, U, F_M, R_M)$, where X is the state space, U is the action space, $F_M : X \times U \rightarrow X$ is the state transition function that describes the process to be controlled, and R_M represents the reward function. The function F_M, modeling the evolution of the state depending on the actual state and input, is stochastic. Thus, it associates at all the states in X a certain probability to be acquired by the system in the actual conditions. The goal of the RL is to find an optimal control policy $\pi : X \rightarrow U$

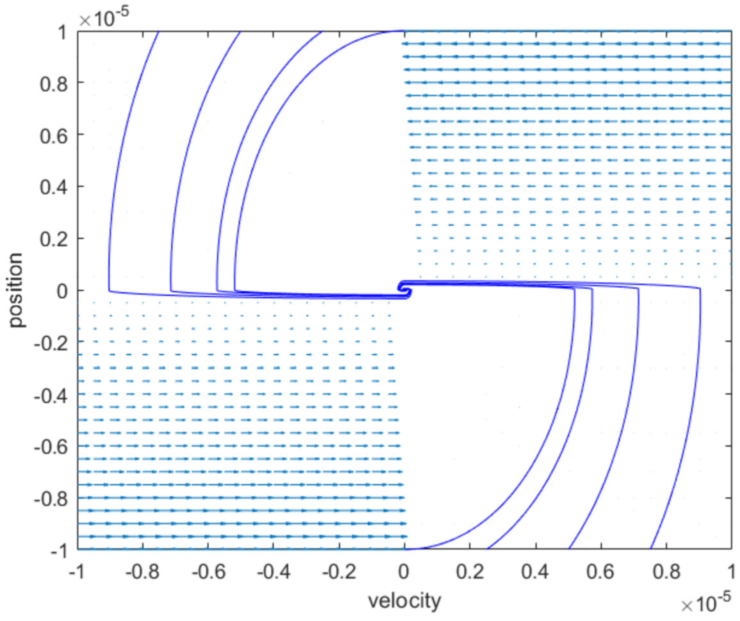


Fig. 4. Representation of the vector field of the differential inclusion with the associated streamlines, which represent the trajectory solution for different initial conditions.

by optimizing the cumulative or total reward function [60]. In this paper, the proposed cost function to be minimized is defined as the integral of the utility function from time instant t_k to infinity:

$$R_M = V(\mathbf{z}(t_k)) = \int_{t_k}^{t_\infty} r(\mathbf{z}(\tau), \mathbf{u}(\tau))d\tau, \tag{47}$$

where the utility function $r(\cdot)$ is equal to the previously defined quadratic cost function in (17). The control action $\mathbf{u}$ belongs to the piece-wise constant functions $\mathbf{u}(t) = \mathbf{u}(t_k)\ \forall\ t \in [t_k, t_{k+1})$, and it is chosen by the control policy π depending on the state $\mathbf{z}$. For this reason, V is function of the state only.

The goal of the RL is the resolution of an optimization problem, therefore, to define a series of input actions that minimizes the value function:

$$V^*(\mathbf{z}(t_k)) = \min_{\mathbf{u}(\cdot)} \int_{t_k}^{t_\infty} \psi^n r(\mathbf{z}(\tau), \mathbf{u}(\tau))d\tau, \tag{48}$$

where ψ is the discounted factor of the Bellman function. According to the Bellman principle, it can be reformulated as it follows:

$$V^*(\mathbf{z}(t_k)) = \min_{\mathbf{u}(\cdot)} \int_{t_k}^{t_{k+1}} \psi^n r(\mathbf{z}(\tau), \mathbf{u}(\tau))d\tau + V^*(\mathbf{z}(t_{k+1})). \tag{49}$$

In the framework of the Q-learning, the reward function to be optimized can be expressed as the function Q, dependent on the present state and input action. Thus, the previous expression can be rewritten as:

$$Q^*(\mathbf{z}(t_k), \mathbf{u}(t_k)) = \min_{\mathbf{u}(\cdot)} \int_{t_k}^{t_{k+1}} r(\mathbf{z}(\tau), \mathbf{u}(\tau))d\tau + Q^*(\mathbf{z}(t_{k+1}), \mathbf{u}(t_{k+1})), \tag{50}$$

and the optimal control law $\mathbf{u}^*(t_k)$ results in:

$$\mathbf{u}^*(t_k) = \arg\min_{\mathbf{u}(\cdot)} Q^*(\mathbf{z}(t_k), \mathbf{u}(t_k)), \tag{51}$$

$$\text{s.t.} \quad \mathbf{z}(t_{k+1}) = F_M(\mathbf{z}(t_k), \mathbf{u}(t_k)). \tag{52}$$

These equations can be used to derive a Q-learning-based algorithm to obtain the optimal control policy of the LMPC problem iteratively [38]. The algorithm implementation related to the described methodology is summarized in the pseudo-code in Algorithm 1 (Q-Learning).

The set $\bar{U}$ in which the input action is optimized is defined as $\bar{U} = U \cap L_U$, where L_U is the set of inputs that respect the Lyapunov constraint expressed in the LMPC (as described in the previous Section). Enforcing such defined Lyapunov constraint, once the algorithm is converged, the policy should inherit the stability and the robustness of the Lyapunov-based controller. The initialization of the Q-function at the step 4 in the Algorithm 1 facilitates the calculation of the initial policy in the following step 5. In any case, considering the learning process of the iterative Q-function, the initialization method does not affect its update and convergence. Theoretically, the convergence is reached for $i \to \infty$. However, this is not feasible in practical applications, due to the limited computing resources and real-time computing requirements. Indeed, in these cases, a sub-optimal control law is obtained after a finite number of iterations. Besides, by approximating the Q-function and the control policy with neural networks in online update, an additional fitting error is introduced. Nevertheless, this method still allows to reach very high accuracy in the control of the system [38].

Algorithm 1 Q-Learning.

```
1: Initialize the error threshold $\epsilon > 0$
2: for $k = k_0, k_0 + 1, \ldots, \infty$ do
3:     Measure the current state vector $\mathbf{z}(t_k)$ and set $i = 0$
4:     Initialize $Q^0(\cdot) = 0$
5:     Calculate the initial control $\mathbf{u}^0(t_k)$ by
           $$\mathbf{u}^0(t_k) = \arg\min_{\mathbf{u}(\cdot)\in\bar{U}} \int_{t_k}^{t_{k+1}} r(\mathbf{z}(\tau), \mathbf{u}(\tau))d\tau + Q^0(\cdot) \quad (53)$$
6:     while
           $$\left\| Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k)) - Q^i(\mathbf{z}(t_k), \mathbf{u}^{i-1}(\mathbf{z}(t_k))) \right\| > \epsilon \quad (54)$$
       do
7:         Update the sequence of action policies:
               $$\mathbf{u}^i(t_k) = \arg\min_{\mathbf{u}(\cdot)\in\bar{U}} \int_{t_k}^{t_{k+1}} r(\mathbf{z}(\tau), \mathbf{u}(\tau))d\tau + Q^i(\mathbf{z}(t_{k+1}), \mathbf{u}^{i-1}(t_{k+1})) \quad (55)$$
8:         Update the sequence of Q-functions:
               $$Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k)) = \int_{t_k}^{t_{k+1}} r(\mathbf{z}(\tau), \mathbf{u}^i(\tau))d\tau + Q^i(\mathbf{z}(t_{k+1}), \mathbf{u}^{i-1}(\mathbf{z}(t_{k+1}))) \quad (56)$$
9:     Apply $\mathbf{u}(t) = \mathbf{u}^i(t_k),\ t \in [t_k, t_{k+1})$ to the system
```

In contrast to the ordinary Q-Learning algorithm, in the proposed approach, the agent adopts the same action policy used for an update of the Q-function. In literature, this kind of algorithm is named as SARSA. Considering that a greedy policy is adopted, it can still be considered as a Q-learning algorithm. This peculiarity depends on the fact that, in the proposed pHRC scenario, it is not possible to identify two separate time-windows for the exploration of the environment, waiting for the function to converge, and for the optimal control of the system. Indeed, the best control available at the time is immediately applied to the system.

The convergence of the proposed Q-Learning algorithm need to be proved. Exploiting the three lemma, it will be proved that for $i \to \infty$, the sequence $\{Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k))\}$ tends to the optimal Q-function $Q^*(\mathbf{z}(t_k), \mathbf{u}^i(t_k))$, and the estimated control policy sequence $u^i(t_k)$ tends to the optimal control policy $u^*(t_k)$, where $u(t) = u(t_k),\ t \in [t_k, t_{k+1})$. A similar demonstration is also available in [38].

Lemma 2.4. *Let $\mathbf{a}^i$ be any arbitrary control sequence subjected to the control constraint $\bar{U}$. The corresponding value function $\Lambda^{i+1}(\cdot)$ is updated by:*

$$\Lambda^{i+1}(\mathbf{z}(t_k), \mathbf{a}^i(t_k)) = \int_{t_k}^{t_{k+1}} r(\mathbf{z}(\tau), \mathbf{a}^i(\tau))d\tau + \Lambda^i(\mathbf{z}(t_{k+1}), \mathbf{a}^{i-1}(\mathbf{z}(t_{k+1}))), \tag{57}$$

with $\Lambda^0(\cdot) = 0$. Then, $Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k)) \leq \Lambda^{i+1}(\mathbf{z}(t_k), \mathbf{a}^i(t_k))$ can be satisfied for any iterative number i.

Proof. Note that $Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k))$ is the result of a minimization done by using the optimal input $\mathbf{u}^i(t_k)$, while $\Lambda^{i+1}(\mathbf{z}(t_k), \mathbf{a}^i(t_k))$ is achieved under an arbitrary control input $\mathbf{a}^i(t_k)$; it can be concluded that $Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k)) \leq \Lambda^{i+1}(\mathbf{z}(t_k), \mathbf{a}^i(t_k))$. □

Lemma 2.5. *Consider the Q-function sequence* $\{Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k))\}$ *defined in the algorithm, and the corresponding system input sequence* $\{\mathbf{u}^i(t_k)\}$. *Then, it follows that* $\forall\, i \quad Q^i \leq Q^{i+1}$.

Proof. Setting the arbitrary control sequence $\mathbf{a}^i(t_k) = \mathbf{u}^{i+1}(t_k)$ for Λ^{i+1}:

$$\Lambda^{i+1}(\mathbf{z}(t_k), \mathbf{u}^{i+1}(t_k)) = \int_{t_k}^{t_{k+1}} r(\mathbf{z}(\tau), \mathbf{u}^{i+1}(\tau))d\tau + \Lambda^i(\mathbf{z}(t_{k+1}), \mathbf{u}^i(\mathbf{z}(t_{k+1}))), \tag{58}$$

$$Q^{i+2}(\mathbf{z}(t_k), \mathbf{u}^{i+1}(t_k)) = \int_{t_k}^{t_{k+1}} r(\mathbf{z}(\tau), \mathbf{u}^{i+1}(\tau))d\tau + Q^{i+1}(\mathbf{z}(t_{k+1}), \mathbf{u}^i(\mathbf{z}(t_{k+1}))). \tag{59}$$

Considering the initialized value functions $Q^0(\cdot) = \Lambda^0(\cdot) = 0$, for $i = 0$:

$$Q^1(\mathbf{z}(t_k), \mathbf{u}^0(t_k)) - \Lambda^0(\cdot) = \int_{t_k}^{t_{k+1}} r(\mathbf{z}(\tau), \mathbf{u}^0(\tau))d\tau \geq 0. \tag{60}$$

This difference is kept for all i since the $Q^{i+1}(\cdot)$ and $\Lambda^i(\cdot)$ terms are always increased of the same quantity $\int_{t_k}^{t_{k+1}} r(\mathbf{z}(\tau), \mathbf{u}^i(\tau))d\tau$. Indeed, $Q^{i+1}(\cdot) \geq \Lambda^i(\cdot)\ \forall i$. Knowing from Lemma 2.4 that $Q^i(\cdot) \leq \Lambda^i(\cdot)$, it can be concluded that:

$$Q^{i+1}(\cdot) \geq \Lambda^i(\cdot) \geq Q^i(\cdot) \Rightarrow Q^{i+1}(\cdot) \geq Q^i(\cdot)\ \ \forall\, i. \quad \square$$

Lemma 2.6. *Consider the Q-function sequence* $\{Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k))\}$ *defined in the algorithm. There is an admissible control policy that makes the control law and the corresponding system state* $\mathbf{z}$ *be* $\mathbf{0}$ *in a limited time* t_{Nf}. *There exists an upper bound* $Y(t_k)$ *such that* $\forall\, i\ Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k)) \leq Q^*(\mathbf{z}(t_k), \mathbf{u}^*(t_k)) \leq Y(t_k)$.

Proof. Let $\mu(\mathbf{z})$ be an arbitrary admissible control policy, and set $\mathbf{a}^i = \mu(\mathbf{z}(t_k))$. At this point, $\Lambda^{i+1}(\mathbf{z}(t_k), \mu(\mathbf{z}(t_k)))$ can be expressed as:

$$\begin{aligned}
&\Lambda^{i+1}(\mathbf{z}(t_k), \mu(\mathbf{z}(t_k))) = \\
&= \int_{t_k}^{t_{k+1}} r(\mathbf{z}(\tau), \mu(\mathbf{z}(\tau)))d\tau + \Lambda^i(\mathbf{z}(t_{k+1}), \mu(\mathbf{z}(t_{k+1}))) \\
&= \int_{t_k}^{t_{k+2}} r(\mathbf{z}(\tau), \mu(\mathbf{z}(\tau)))d\tau + \Lambda^{i-1}(\mathbf{z}(t_{k+2}), \mu(\mathbf{z}(t_{k+2}))) \\
&= \int_{t_k}^{t_{k+i+1}} r(\mathbf{z}(\tau), \mu(\mathbf{z}(\tau)))d\tau + \Lambda^0(\mathbf{z}(t_{k+i+1}), \mu(\mathbf{z}(t_{k+i+1}))) \\
&\leq \int_{t_k}^{t_\infty} r(\mathbf{z}(\tau), \mu(\mathbf{z}(\tau)))d\tau = V(\mathbf{z}(t_k)).
\end{aligned} \tag{61}$$

This is valid for every admissible control policy. Substituting the general μ with $\mathbf{u}^*$, it is possible to get $Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^*(\mathbf{z}(t_k))) \leq Q^*(\mathbf{z}(t_k), \mathbf{u}^*(\mathbf{z}(t_k)))$, that implies:

$$Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k)) \leq Q^*(\mathbf{z}(t_k), \mathbf{u}^*(\mathbf{z}(t_k))). \tag{62}$$

Besides, let μ_0 be the admissible control policy that makes the control law and the corresponding system state $\mathbf{z}$ be $\mathbf{0}$ in a limited time t_{Nf}. It is not optimal, thus:

$$\begin{aligned} Q^*(\mathbf{z}(t_k), \mathbf{u}^*(\mathbf{z}(t_k))) &\le \int_{t_k}^{t_\infty} r(\mathbf{z}(\tau), \mu_0(\mathbf{z}(\tau)))d\tau \\ &= \int_{t_k}^{t_{Nf}} r(\mathbf{z}(\tau), \mu_0(\mathbf{z}(\tau)))d\tau + \int_{t_{Nf+1}}^{t_\infty} r(\mathbf{z}(\tau), \mu_0(\mathbf{z}(\tau)))d\tau \\ &= \int_{t_k}^{t_{Nf}} r(\mathbf{z}(\tau), \mu_0(\mathbf{z}(\tau)))d\tau = Y(t_k). \end{aligned} \tag{63}$$

Therefore, the condition $\forall\ i\ Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k)) \le Q^*(\mathbf{z}(t_k), \mathbf{u}^*(t_k)) \le Y(t_k)$ is verified. □

Theorem 2.7. *Consider the iterative control law u^i and Q^{i+1} defined in Algorithm 1. When $i \to \infty$, one has $\mathbf{u}^i \to \mathbf{u}^*$ and $Q^{i+1} \to Q^*$.*

Proof. According to Lemma 2.5 and the definition of the optimal Q-function, if $i \to \infty$ it results:

$$Q^*(\mathbf{z}(t_k), \mathbf{u}^*(t_k)) \le \lim_{i\to\infty} Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k)). \tag{64}$$

According to Lemma 2.6, it is possible to write:

$$Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k)) \le Q^*(\mathbf{z}(t_k), \mathbf{u}^*(t_k)) \le Y(t_k) \quad \forall\, i. \tag{65}$$

Thus:

$$\lim_{i\to\infty} Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k)) = Q^*(\mathbf{z}(t_k), \mathbf{u}^*(t_k)). \tag{66}$$

The theorem is therefore demonstrated. □

2.6. Actor-Critic neural networks

The Algorithm 1 can be implemented online using a proper threshold ϵ to limit the number of iterations. However, in this way, many constrained minimization problems need to be solved at each time instant, and the result of such processes is very inefficient from the computational point of view. Therefore, to overcome this problem, Q-Learning can be implemented using a couple of neural networks to approximate the Q-function and the policy π.

The **actor** neural network is the responsible for the update of the variable impedance parameters, and it represents the action choosing the policy of the Q-Learning. It approximates the LMPC control law $u_{LMPC}(t_k, \mathbf{z}(t_k))$ by processing the vector of the measured state $\hat{\mathbf{z}} = [\dot{x}, x, f_h]^T$ through 3 hidden layers to return the two optimized impedance parameters $\tilde{\mathbf{u}} = [\tilde{x}_d, \tilde{D}_h]^T$ (*i.e.*, the setpoint and the damping parameters). The structure is the following:

$$\tilde{\mathbf{u}}_{nm} = \tanh\left(\mathbf{A}_3 \cdot \sigma_{RLU}(\mathbf{A}_2 \cdot \sigma_{RLU}(\mathbf{A}_1\hat{\mathbf{z}} + \mathbf{b}_1) + \mathbf{b}_2) + \mathbf{b}_3\right). \tag{67}$$

$\tilde{\mathbf{u}}_{nm}$ indicates the non-normalized output of the ANN. The couples (A_i, b_i) represent the three linear layers of the neural network, that transform the input vector with a matrix multiplication, adding a bias. Since 64 units per layer are used, the dimensions of the matrices are:

$$\begin{aligned} &\mathbf{A}_1 \in \mathbb{R}^{64\times 3},\ \mathbf{A}_2 \in \mathbb{R}^{64\times 64},\ \mathbf{A}_3 \in \mathbb{R}^{2\times 64}, \\ &\mathbf{b}_1 \in \mathbb{R}^{64},\ \mathbf{b}_2 \in \mathbb{R}^{64},\ \mathbf{b}_3 \in \mathbb{R}^{2}. \end{aligned}$$

The activation function σ_{RLU} is the Rectified Linear Unit, defined as it follows:

$$\sigma_{RLU}(x) = x^+ = max(0, x). \tag{68}$$

Due to the hyperbolic tangent, the output is constrained in the range $[-1, 1]$, and it should be denormalized:

$$\tilde{\mathbf{u}} = \tilde{\mathbf{u}}_{nm} \cdot \mathbf{u}_{std} + \mathbf{u}_{avr}, \tag{69}$$

$$\mathbf{u}_{avr} = (\mathbf{u}_{max} + \mathbf{u}_{min})/2, \tag{70}$$

$$\mathbf{u}_{std} = (\mathbf{u}_{max} - \mathbf{u}_{min})/2. \tag{71}$$

These constraints represent the saturation limit of the input, and this is a good method to enforce them for the implementation of the proposed controller in real applications.

During the training, the output of the actor neural network $\bar{\mathbf{u}}$ is compared with $\bar{\bar{\mathbf{u}}}$, the result of the minimization of the Q-function, approximated by the critic neural network $\bar{Q}$:

$$\bar{\bar{\mathbf{u}}}(t_k) = \arg\min_{\mathbf{u}(\cdot)\in\bar{U}} \bar{Q}\left(\mathbf{z}(t_k), \mathbf{u}(t_k)\right), \tag{72}$$

$$\mathbf{e}_a = \bar{\mathbf{u}}(t_k) - \bar{\bar{\mathbf{u}}}(t_k), \quad E_a = \frac{1}{2}\left\|\mathbf{e}_a\right\|^2. \tag{73}$$

The stand gradient descent algorithm is used to update the weight for the actor network, that is implemented as it follows:

$$\mathbf{A}_i^{j+1} = \mathbf{A}_i^j - l_a \frac{\partial E_a}{\partial \bar{\mathbf{u}}(t_k)} \frac{\partial \bar{\mathbf{u}}(t_k)}{\partial \mathbf{A}_i}, \tag{74}$$

$$\mathbf{b}_i^{j+1} = \mathbf{b}_i^j - l_a \frac{\partial E_a}{\partial \bar{\mathbf{u}}(t_k)} \frac{\partial \bar{\mathbf{u}}(t_k)}{\partial \mathbf{b}_i}, \tag{75}$$

where l_a is the learning rate of actor.

The **critic** neural network is used to approximate the Q-function, that in general is expressed as:

$$\bar{Q} = \int_{t_k}^{t_\infty} r(\mathbf{z}(\tau), \bar{\mathbf{u}}(\tau))d\tau, \tag{76}$$

where $\bar{\mathbf{u}}(t_k)$ is approximated by the actor neural network. Since $r(\cdot) = f_h f_h^T$, $\bar{\mathbf{u}}$ is not directly present in the functional. However, it determines the future states, *i.e.*, the values of f_h from t_{k+1} onward. In fact, this network needs to approximate the sum of the costs associated to the states probably covered according to a greedy control policy, and it requires as an input the whole state $(\mathbf{z}_t; \bar{\mathbf{u}}_t)^T$ to be able to estimate the future system evolution. Then, 3 hidden layers of 128 neurons each are used to elaborate the state and return an estimation of the Q-function. The adopted structure is as it follows:

$$\begin{aligned}
\bar{Q} &= \mathbf{C}_3 \cdot \sigma_{RLU}(\mathbf{C}_2 \cdot \sigma_{RLU}(\mathbf{C}_1(\mathbf{z}_t; \bar{\mathbf{u}}_t)^T + \mathbf{d}_1) + \mathbf{d}_2) + d_3, \\
&\mathbf{C}_1 \in \mathbb{R}^{128\times 5},\ \mathbf{C}_2 \in \mathbb{R}^{128\times 128},\ \mathbf{C}_3 \in \mathbb{R}^{128}, \\
&\mathbf{d}_1 \in \mathbb{R}^{128},\ \mathbf{d}_2 \in \mathbb{R}^{128},\ d_3 \in \mathbb{R}.
\end{aligned} \tag{77}$$

Then, the approximated error and squared error of the critic network are, respectively, defined as e_c and E_c:

$$e_c = \bar{Q}(t_k) - \bar{\bar{Q}}(t_k), \quad E_c = \frac{1}{2}\|e_c\|^2, \tag{78}$$

where $\bar{\bar{Q}}$ is the result of the Bellman equation:

$$\bar{\bar{Q}}(t_k) = \int_{t_k}^{t_{k+1}} r(\cdot)d\tau + \bar{Q}(t_{k+1}). \tag{79}$$

Again, a variant of the gradient descent algorithm is used to update the weight during the training of the critic network, that can be schematized as it follows:

$$\mathbf{C}_i^{j+1} = \mathbf{C}_i^j - l_c \frac{\partial E_c}{\partial \bar{Q}(t_k)} \frac{\partial \bar{Q}(t_k)}{\partial \mathbf{C}_i}, \tag{80}$$

$$\mathbf{d}_i^{j+1} = \mathbf{d}_i^j - l_c \frac{\partial E_c}{\partial \bar{Q}(t_k)} \frac{\partial \bar{Q}(t_k)}{\partial \mathbf{d}_i}, \tag{81}$$

where l_c is the learning rate of critic.

2.7. Cross entropy method

In the previous section the minimization of $\bar{Q}$ has been described, necessary during the training of actor to define the error $\mathbf{e}_a$. Thus, another optimization problem is introduced, which can be solved as a discrete optimal control problem since, for this application, the input can only be piece-wise constant and the state has already been sampled. To choose the best input action $\bar{\bar{\mathbf{u}}}(t_k)$, the evolution of the interaction force f_h suggested by the ensemble of neural networks along a time horizon long enough has to be considered. This problem can be formalized as:

$$\begin{aligned}\bar{\bar{\mathbf{u}}}_k &= \arg \min_{\mathbf{u}(\cdot)\in\bar{U}} \bar{Q}(\mathbf{z}_k, \mathbf{u}_k) \\ &= \arg \min_{\mathbf{u}(\cdot)\in\bar{U}} \sum_{k=0}^{N-1} r(\cdot) + \bar{Q}(\mathbf{z}_{N-1}, \mathbf{u}_{N-1}),\end{aligned} \tag{82}$$

$$\text{s.t.}\quad \dot{\mathbf{z}}_{k+1} = \mathbf{f}(\mathbf{z}_k) + \mathbf{g}_1(\mathbf{z}_k)u_{1_k} + \mathbf{g}_2(\mathbf{z}_k)u_{2_k}, \tag{83}$$

$$\mathbf{z}_0 = \tilde{\mathbf{z}}(t_k), \tag{84}$$

$$\bar{\bar{\mathbf{u}}}_k \in \bar{U},$$

$$k = 0, 1, ..., N-1.$$

The exact computation of the optimum is not possible due to the high non-linearity of the problem, and due to the uncertainties that come with the modeling of the system. In order to compute an approximated solution of the above finite-horizon control problem, the Cross-Entropy Method (CEM) is exploited [61]. CEM is a probabilistic optimization belonging to the field of Stochastic Optimization. The strategy of the algorithm is to sample the problem space, approximating the distribution of suitable solutions. This is achieved by assuming a distribution of the problem space (such as Gaussian), sampling from this distribution to generate possible candidate solutions. Then, the distribution is updated on the basis of the best found candidate solutions. As the algorithm progresses, the distribution becomes more refined, until it focuses on the area of optimal solutions. The pseudo-code describing its implementation is given in Algorithm 2 (Cross Entropy Method).

Algorithm 2 Cross entropy method.

```
1: Initialize the mean $\mu$ and the standard deviation $\sigma$ of the stochastic distributions
2: for $iter = 0, 1, ..., N_{it}$ do
3:     Sample the sequence of outputs $\{\bar{\bar{\mathbf{u}}}^j\}$
4:     Saturate the values out of the limit range
5:     for $k = 0, 1, ..., N-1$ do
6:         Predict the system evolution for each sample $j$ through the ensemble of neural networks $\rightarrow \bar{\mathbf{z}}_k^j$
7:     Evaluate $cost_j = \sum_{k=0}^{N-1} r(f_{h_k}^j) + \bar{Q}(\bar{\mathbf{z}}_{N-1}^j, \bar{\bar{\mathbf{u}}}_{N-1}^j)$
8:     Order the samples according to $cost_j$
9:     Choose the best $s$ elite samples.
10:    Compute their mean $\mu_{cp}$ and standard deviation $\sigma_{cp}$
11:    Use them to update the stochastic distributions, considering a smoothing factor $\alpha$:
           $\mu^{i+1} = (1-\alpha)\mu^i + \alpha\mu_{cp}$,    $\sigma^{i+1} = (1-\alpha)\sigma^i + \alpha\sigma_{cp}$
12: Return the best sample $\bar{\bar{\mathbf{u}}}_j$
```

It is important to highlight that the value function to be minimized is considered through the Bellman equation, so that the contributions closest in time are directly evaluated from the cost function. In such a way, the prediction capabilities of the model approximator are exploited to estimate the state evolution, and to obtain a final result that is more accurate than the one achieved by an algorithm that relays only on the Q-function. The limit range out of which the generated input must be cut is the previously defined set $\bar{U}$. It is the intersection of U, that includes all the input actions that respect the saturation limits, and the set L_U, that include the inputs that respect the Lyapunov constraint expressed in the LMPC. In fact, these constraints can be decoupled and transferred into the upper and lower bounds of each input.

The conditions present in the definition of the Model Predictive Control can be made stricter to be reformulated for the single inputs:

$$\left[\frac{\partial V_L(\mathbf{z}(t_k))}{\partial \mathbf{z}}\mathbf{g}_i(t_k)(\mathbf{z})\right] u_i(t_k) \le \left[\frac{\partial V_L(\mathbf{z}(t_k))}{\partial \mathbf{z}}\mathbf{g}_i(t_k)(\mathbf{z})\right] h_i(\mathbf{z}(t_k)), \tag{85}$$

where $i = 1, 2$, $\left[\frac{\partial V_L(\mathbf{z}(t_k))}{\partial \mathbf{z}}\mathbf{g}_i(t_k)(\mathbf{z})\right] \in \mathbb{R}$, and $u_i(t_k) \in \mathbb{R}$. In such a way, the stability of the system respect to each single input is imposed. Simplifying w.r.t. the common term, it is possible to write:

$$\begin{cases} u_i(t_k) \le h_i(\mathbf{z}(t_k)) & \text{if } \left[\frac{\partial V_L(\mathbf{z}(t_k))}{\partial \mathbf{z}}\mathbf{g}_i(t_k)(\mathbf{z})\right] > 0, \\ u_i(t_k) \ge h_i(\mathbf{z}(t_k)) & \text{if } \left[\frac{\partial V_L(\mathbf{z}(t_k))}{\partial \mathbf{z}}\mathbf{g}_i(t_k)(\mathbf{z})\right] < 0. \end{cases} \tag{86}$$

Combining these limits with the saturation constraints, it gets to:

$$\begin{cases} u_i^{min} \le u_i(t_k) \le h_i(\mathbf{z}(t_k)) & \text{if } \left[\frac{\partial V_L(\mathbf{z}(t_k))}{\partial \mathbf{z}}\mathbf{g}_i(t_k)(\mathbf{z})\right] > 0, \\ h_i(\mathbf{z}(t_k)) \le u_i(t_k) \le u_i^{max} & \text{if } \left[\frac{\partial V_L(\mathbf{z}(t_k))}{\partial \mathbf{z}}\mathbf{g}_i(t_k)(\mathbf{z})\right] < 0, \end{cases} \tag{87}$$

that is the explicit definition of the set $\bar{U}$. Note that in case $\dot{x} = 0$, $\left[\frac{\partial V_L(\mathbf{z}(t_k))}{\partial \mathbf{z}}\mathbf{g}_i(t_k)(\mathbf{z})\right] = 0\ \forall i$, and:

$$\frac{\partial V_L(\mathbf{z})}{\partial \mathbf{z}}(\mathbf{f}(\mathbf{z}) + \mathbf{g}_1(\mathbf{z})u_1 + \mathbf{g}_2(\mathbf{z})u_2) = \frac{\partial V_L(\mathbf{z})}{\partial \mathbf{z}}(\mathbf{f}(\mathbf{z}) + \mathbf{g}_1(\mathbf{z})h_1 + \mathbf{g}_2(\mathbf{z})h_2) = 0. \tag{88}$$

Therefore, it is not necessary to introduce further limitations on the set U.

2.8. Online updating the Q-LMPC

All the ANNs employed by the controller are trained online to provide the algorithm with the capability of dynamically adapting to new data patterns. In the RL framework, the online learning approach (as opposed to batch learning) allows the immediate knowledge transfer to the agent, which is also valuable to guide the data acquisition in the most convenient direction. This method results in higher exploitation and less sensitivity to the learning parameters for the system to converge [62]. Besides, the memory for most critical tasks is preserved by using the data shortly after their acquisition. The design and implementation of the online controller to effectively and efficiently update the impedance control parameters is proposed below, considering a nonlinear system and its unknown dynamics (as introduced in Fig. 1). The training of the neural networks described in Section 2.6 recalls the update of the sequences $\{Q^{i+1}(\mathbf{z}(t_k), \mathbf{u}^i(t_k))\}$, and $\{\mathbf{u}^i(t_k))\}$, shown in the Algorithm 1. Therefore, the proposed Q-LMPC (summarized in the pseudo-code in Algorithm 3) is a fair approximation of Q-Learning and, once the networks converged, it tends to the result of the LMPC problem.

Algorithm 3 Q-LMPC.

```
 1: Initialize the weight of the actor neural network and the critic neural network
 2: Initialize the weight of the neural networks of the model approximator ensemble
 3: for $k = k_0, k_0 + 1, ..., \infty$ do
 4:     Measure the current state vector $\mathbf{z}(t_k) = [\dot{x}, x, f_h]^T$
 5:     Store the transition $(\mathbf{z}(t_{k-1}), \mathbf{u}(t_{k-1}), \mathbf{z}(t_k))$ inside the buffer $D$
 6:     Calculate the control law $\tilde{\mathbf{u}}(t_k)$ through equations (67) and (69):

            $\tilde{\mathbf{u}}_{nm} = \tanh\left(\mathbf{A}_3 \cdot \sigma_{RLU}(\mathbf{A}_2 \cdot \sigma_{RLU}(\mathbf{A}_1\hat{\mathbf{z}} + \mathbf{b}_1) + \mathbf{b}_2) + \mathbf{b}_3\right)$

            $\tilde{\mathbf{u}} = \tilde{\mathbf{u}}_{nm} \cdot \mathbf{u}_{std} + \mathbf{u}_{avr}$

 7:     Apply $\tilde{\mathbf{u}}(t_k)$ to the system
 8:     if $k/N_h$ is integer then
 9:         Get $N_h$ transitions $(\mathbf{z}_n, \mathbf{u}_n, \mathbf{z}_{n+1})$ from $D$, where $n = 1, 2, ..., N_h$
10:         Update the model approximators
11:         for each transition $(\mathbf{z}_n, \mathbf{u}_n, \mathbf{z}_{n+1})$ in $N_h$ do
12:             Use $\mathbf{z}_n$ and $\mathbf{u}_n$ to compute $\bar{Q}_n$
13:             Use $\mathbf{z}_{n+1}$ to compute $\bar{\mathbf{u}}_{n+1}$ through the actor neural network
14:             Use $\mathbf{z}_{n+1}$ and $\bar{\mathbf{u}}_{n+1}$ to compute $\bar{Q}_{n+1}$
15:             Apply the Bellman equation to find $\bar{\bar{Q}}_n$
16:             Calculate $e_c$, $E_c$ and update the critic weight
17:         for each transition $(\mathbf{z}_n, \mathbf{u}_n, \mathbf{z}_{n+1})$ in $N_h$ do
18:             Find $\bar{\bar{u}}_{n+1}$ minimizing $\bar{Q}(\mathbf{z}_{n+1})$ with CEM
19:             Calculate $\mathbf{e}_a$, $\mathbf{E}_a$ and update the actor weight
```

The catastrophic interference issue in the actor-critic network training is addressed by adapting the learning rate according to the measured state. In this way, the operating scenarios that provide non-meaningful information are detected, reducing their relevance in the learning. As a matter of example, the learning rate of the actor-network is decreased when the operator and the robot do not interact. Due to the limited computational capability of the Linux rt kernel (in which is not possible to make use of GPUs), it is not possible to update the networks at every iteration (*i.e.*, 6 Hz), requiring to slow down the frequency to 1.2 Hz. During training, the record of the states previously covered (*i.e.*, $z_n, z_{n+1}, \ldots$) has to be performed into a buffer D of size N_h. When the buffer fills up, the training phase is activated, and the operations inside the dashed square in Fig. 1 are performed. The data stored into the buffer are provided to the training framework in the same order in which they are acquired. It is important to have a buffer of a small size, both to shorten the idle time of the training process (to benefit of the online learning properties) and to reduce the variance between the data used in two subsequent training steps. Indeed, if these data are perceived as sampled from different distributions, a sudden variation of the control policy can occur, leading to an unstable behavior. This issue has also been tackled designing a proper data normalization and the experimental results assess the successful outcome of these solutions. From the computational point of view, the training of the actor is the most expensive part of the algorithm due to the application of the CEM. Therefore, after it converges, it is possible to stop the training, increasing the update frequency of the controller.

3. Experimental results

In this Section, the experimental validation of the proposed Q-LMPVIC for pHRC tasks is described. The proposed controller has been compared w.r.t. the one in [20], previously developed by some of the authors. Both the controllers have

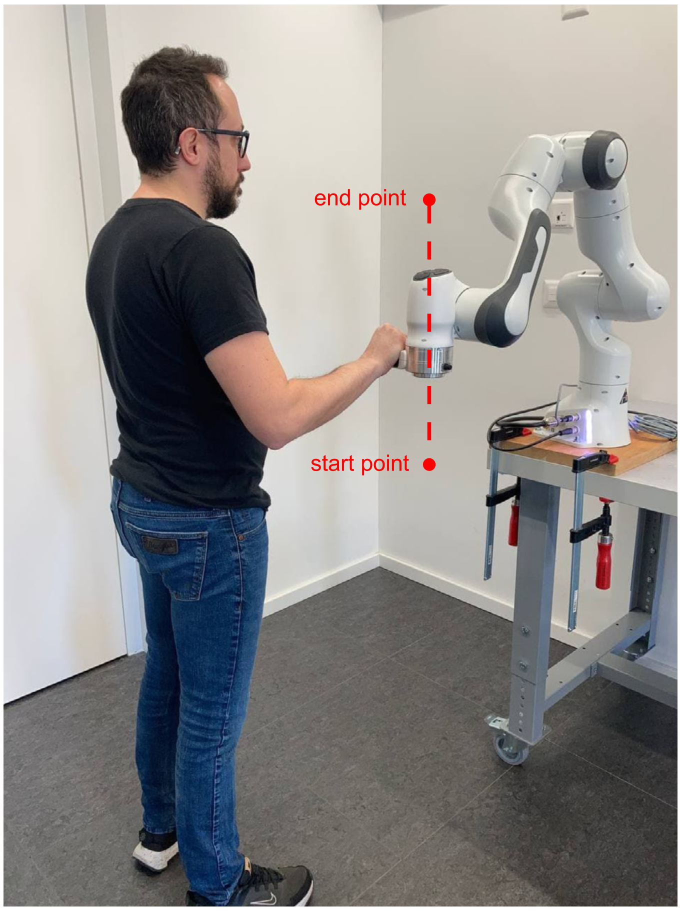


Fig. 5. Franka EMIKA panda Robot involved in the validation experiments. Considering the experimental scenario i), the user interacts with the manipulator at its end-effector. Considering the experimental scenario ii), a sealing gun is attached to the robot end-effector and the user interacts with the robot through it (see Fig. 13).

been implemented and tested on a Franka EMIKA panda robot (Fig. 5), used as a test platform for the execution of the target pHRC task. In the following, the considered target pHRC tasks for validation purposes are firstly described. Then, the evaluation protocol is introduced. The participants are later detailed. Next, the implementation details of the Q-LMPVIC are given. Finally, the experimental results are shown.

3.1. *Target pHRC tasks*

Two experimental scenarios have been considered for the test and validation of the proposed Q-LMPVIC, compared with the approach in [20]: i) interaction between the human and the robot end-effector (*i.e.*, no tool at the robot end-effector, Fig. 5); ii) interaction between the human and a sealing gun installed at the robot's end-effector. The subjects are supposed to operate the robot along the vertical direction z, where the proposed Q-LMPVIC approach and the one in [20] are activated. Two virtual walls along the z axis (that have been set at 0.7 m and 0.3 m) have been implemented in order to constraint the robot motion. Each subject performs 10 up and down complete motions between the defined virtual walls. The two experimental scenarios have been tested by adopting a pre-training for both the proposed algorithm and the one in [20]. Such pre-training has been performed by one of the authors, Andrea Testa. The running order of the control algorithms has been randomized to balance the evaluation results. In addition, the training process of the proposed Q-LMPVIC approach has been tested considering the experimental scenario i), asking to the participants to perform their own training section in order to evaluate the training procedure of the proposed methodology.

3.2. *Evaluation protocol*

The pre-trained controllers (both the Q-LMPVIC and the controller in [20]) have been evaluated on the basis of the qualitative evaluation framework used in [4,20]. Indeed, the performance of the two tested control methods has been assessed using a questionnaire at the end of the evaluation experiments for all the subjects. The following metrics have been defined to address both physical human-robot interactions (pHRI) and cognitive human-robot interactions (cHRI) to completely address human expectations and task goals [63]:

- naturalness: human's overall likability, normalcy, ease of use, convenience, non- complexity in operation and collaboration;
- smoothness: whether the movement is smooth and characterized by a limited jerk;
- effort: amount of effort, hardship or endeavor required to achieve a performance level satisfying the mental, physical and temporal demand;
- motion: nature of object velocity and acceleration felt by the human (*i.e.*, whether the velocity, acceleration is low or high compared to human expectation);
- stability: presence/absence of oscillations, sudden inactivity of the system, and their effects on manipulation, object, system structure, and environment;
- detection of intention: whether the robot follows human intention in accelerating or de-accelerating the motion;
- performance: the overall performance, *e.g.*, lifting the object to the desired position within the specified time and attempting to avoid user-unfriendly events.

Each of the proposed metrics has a ranking between 0 (minimum ranking, very negative review) to 4 (maximum ranking, very positive review). The ratings have been assigned upon completion of the experiment in one explored scenario and allow to order the feedback in a structured framework. Ratings are based on the subjective opinion of the participants, gained by the feeling with the controllers and indicative mainly of their relative performance. As among volunteers there are expert professionals as well as merely curious, the questionnaire collected a various collection of points of view. So, there is also room for reviews built on a deep know-how, which can be considered closer to an absolute independent evaluation of the controllers.

The training process has been evaluated in order to verify the usability of the proposed controller from scratch (*i.e.*, to be trained by a specific user within a real context of application). The following criteria are considered for the proposed evaluation:

- user-friendliness: whether the training is intuitive or not (*i.e.*, if the necessary actions to train the robot are complex and/or too far from the ordinary way to use the robot);
- duration: time required to achieve an acceptable performance of the controller;
- satisfaction: whether the operator is able to achieve the target performance during the test.

Each of the proposed metrics has a ranking between 0 (minimum ranking) to 4 (maximum ranking).

3.3. Participants

The following subjects have been involved in the experimental evaluation for the different experimental scenarios:

- scenario i) with pre-training: 10 healthy subjects (6 males, 4 females, with mean age = 31 ± 6 years) without any physical problem;
- scenario ii) with pre-training: 10 healthy subjects (10 males, with mean age = 24 ± 1 years) without any physical problem;
- scenario iii) without pre-training: 5 healthy subjects (3 males, 2 females, with mean age = 26 ± 3 years) without any physical problem.

Prior to the testing, all subjects have been informed about the evaluation scenario and the testing procedure.

3.4. Implementation of the Q-LMPVIC

The code required to implement and run the proposed Q-LMPVIC (with detailed instructions) is available at the following GitHub repository: https://github.com/Andrea8Testa/Laboratorio.git. The provided *readme* file contains the precise description of the hyperparameters used to define and train the ANNs for the sake of repeatability.

The proposed Q-LMPVIC has been implemented on a Franka EMIKA panda robot, exploiting the ROS framework. The implementation has been made on top of the available Franka torque control (based on libfranka functionalities), exploiting the Franka Control Interface (FCI), making it possible to send motor signals to the manipulator in real-time at 1 kHz. The end-effector interaction force, fundamental for the evaluation of the cost function, is provided by converting the measurements of the torque sensors installed by the vendor in each actuated joint of the robot employed for experimental evaluation. However, it is important to underline that a force/torque sensor installed at the end-effector of the robot can be used if joint torque sensors are not available on the employed manipulator.

3.4.1. Low-level Cartesian impedance control

The low-level Cartesian impedance control runs at 1 kHz (*i.e.*, the frequency of the Franka EMIKA panda robot torque controller), and its parameters are imposed as follows: the mass parameters have been set to 10 kg, while the inertia parameters are set to 10 kg m^2, the translational and rotational stiffness parameters have been set to 500 N/m and 50 N m/rad respectively. The damping ration parameters, for the non-optimized Cartesian impedance control DoFs, are set to 0.7. The setpoint and the damping parameters are computed online along the controlled Cartesian direction z, exploiting the proposed Q-LMPVIC for the Cartesian DoF z. The Cartesian impedance control has been implemented exploiting the measured interaction wrench $\mathbf{f}_{ext} = \mathbf{f}_h$ applied by the human operator, that is provided by the manipulator (making use of its internal joint torques measurements).

3.4.2. Lyapunov-based controller

As mentioned in Section 2.4, the parameters of the Lyapunov function are chosen empirically, based on experimental testing:

$$V_L = \frac{1}{2}\dot{x}P_1\dot{x}^T + \frac{1}{2}f_h P_2 f_h^T,$$
$$P_1 = 10^{-5}, \quad P_2 = 4 \cdot 10^{-4}.$$

3.4.3. ANNs

The architecture of the ensemble of model approximators and of the actor-critic neural networks have been discussed, respectively, in Section 2.3 and in Section 2.6. To update the weights of the critic neural network during the training, the SGD algorithm of Pytorch is used [64]. Instead, for all the others, the Adam algorithm (an algorithm for first-order gradient-based optimization of stochastic objective functions) [65] is employed. The learning rates are set to 10^{-3} for the model approximators and the critic networks, whereas it goes from 10^{-4} to $5 \cdot 10^{-5}$ in the actor network, according to the magnitude of the measured f_h. The buffer used to store the transitions required to train the ANNs has a size of $N_h = 5$.

3.4.4. CEM

The CEM used to solve the discrete optimal control problem in section 2.7 is characterized by a smoothing rate $\alpha = 0.9$, a prediction horizon $N = 7$, $N_{it} = 4$, $N_{samples} = 15$, and $s = 4$ elite samples.

The saturation limits on input are set to:

$$\mathbf{u}_{max} = \begin{bmatrix} 0.7 \\ 5.0 \end{bmatrix}, \quad \mathbf{u}_{min} = \begin{bmatrix} 0.3 \\ 0.1 \end{bmatrix}.$$

The control frequency of the high-level Q-LMPC is set equal to 6 Hz. This value is mainly determined by the number of ANNs and the settings of the CEM.

3.5. Experimental results

The experimental results related to the human-robot interaction dynamics modeling, to the comparison between the proposed Q-LMPVIC and the MBRLC in [20], and to the evaluation of the training process of the proposed controller are shown in this Section.

3.5.1. Human-robot interaction dynamic modeling evaluation

The accuracy of the learned human-robot interaction dynamic model is fundamental for the effectiveness of the proposed Q-LMPVIC. In fact, on the basis of the learned model, the MPC computes the optimized impedance control parameters to be applied during the human-robot collaboration task. Therefore, such model affects the task dynamics and the human perception of the collaboration with the robot. It is therefore mandatory that the proposed approach is capable to model the complex human-robot interaction dynamics to succeed.

In order to validate the proposed modeling, Fig. 6 shows the training loss and the test loss for the two employed ANNs. The behavior of the pre-trained model approximators has been recorded in the experimental scenario i), extracting random training and testing subsets from the collected data. Two new ANNs are then trained on the first subset, and evaluated on the second one. Such plots highlight that the proposed couple of ANNs is capable to model the human-robot interaction dynamics. Fig. 7 shows the comparison between the measured and the predicted state variables at the end of the aforementioned training, highlighting the capabilities of the modeling to predict the interaction dynamics for control purposes. In fact, as it can be seen especially considering the force estimation, the estimation error is in the same range of model-based state of the art observers [66–69], and, more importantly, as it is shown in [20].

3.5.2. Q-LMPVIC vs. MBRLC within the experimental scenario i)

Considering the experimental scenario i) defined in Section 3.1, the obtained results related to the first six performance indicators detailed in Section 3.2 are shown in Fig. 8 for both the tested controllers. The overall performance indicator is

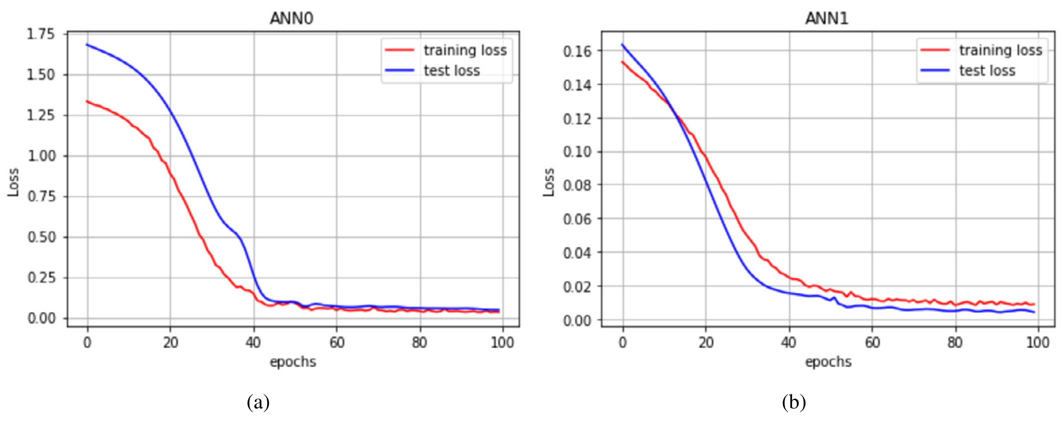


Fig. 6. Training loss (red line) and test loss (blue line) for the 2 ANNs. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

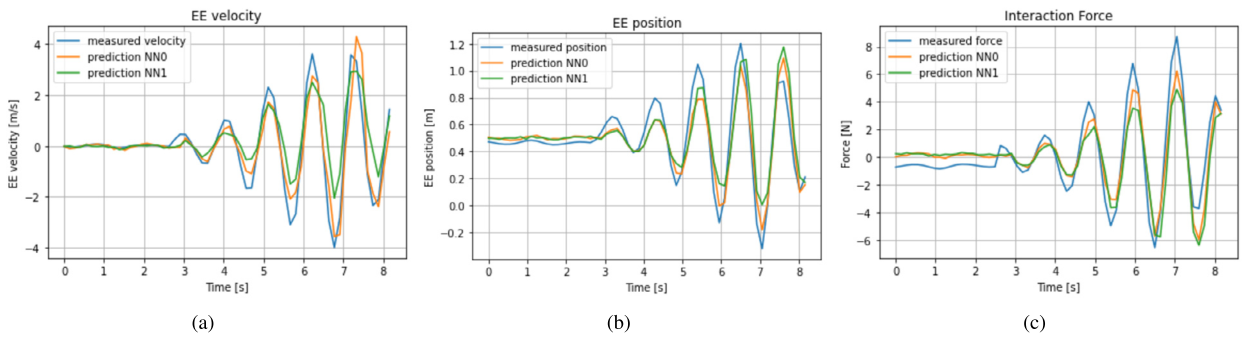


Fig. 7. Comparison between the measured and the predicted state variables.

instead shown in Fig. 9 for both the tested controllers. The controllers achieved a similar result w.r.t. the stability criterion as shown in Fig. 8(e), whereas in all the other metrics the Q-LMPC controller is performing better. The most important improvements are achieved w.r.t. the effort criterion as shown in Fig. 8(c), the motion criterion as shown in Fig. 8(d), and the naturalness criterion as shown in Fig. 8(a). In fact, the novel Q-LMPVIC has been perceived as much lighter and more responsive than the MBRLC. This result can be highlighted by overlapping the evolution of the interaction force and the optimized setpoint. Fig. 10 shows such evolutions, to highlight the reactivity of the proposed controller, being able to quickly adapt the Cartesian impedance setpoint during the human-robot interaction.

3.5.3. *Q-LMPVIC vs. MBRLC within the experimental scenario ii)*

Considering the experimental scenario ii) defined in Section 3.1, the obtained results related to the first six performance indicators detailed in Section 3.2 are shown in Fig. 11 for both the tested controllers. The overall performance indicator is instead shown in Fig. 12 for both the tested controllers. The achieved results in the considered scenario ii) are aligned to the ones achieved in the scenario i): the proposed Q-LMPVIC outperforms the MBRLC. In particular, considering the extra payload attached to the robot end-effector (*i.e.*, the sealing gun), the effort criterion highlights the better performance of the proposed Q-LMPVIC much more w.r.t. the ones of the MBRLC (as shown in Fig. 11(c)). In fact, the proposed Q-LMPVIC is demonstrated to be very reactive, promptly following the intended motion of an operator, *i.e.*, quickly adapting the setpoint so that the applied force decays within few time instants due to the prompt displacement of the end-effector.

3.5.4. *Q-LMPVIC training process validation*

Considering the validation procedure of the Q-LMPVIC training process defined in Section 3.1, the obtained results are shown in Fig. 14. The involved subjects have been globally satisfied by the training process of the Q-LMPVIC. While three subjects considered the first minute of training boring, the controller proved to learn quickly, improving their feedback along with the training. All the subjects found the procedure smooth and simple, and, in the most of the cases, 5 minutes have been enough to achieve acceptable performance. A completely satisfactory controller behavior has been commonly reached after 10 minutes of training.

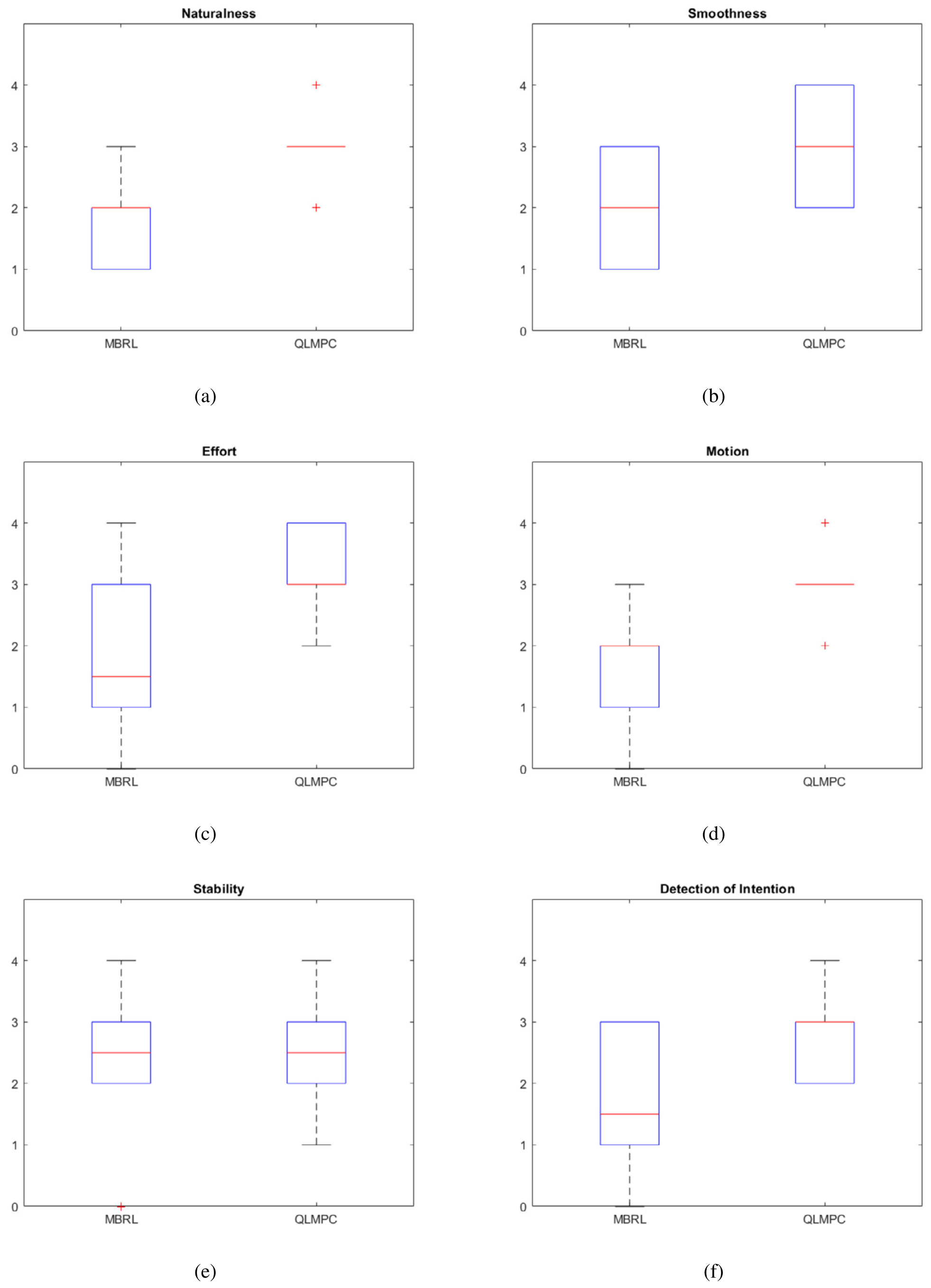


Fig. 8. (a) *Naturalness*, (b) *Smoothness*, (c) *Effort*, (d) *Motion*, (e) *Stability*, (f) *Detection of Intention* criteria for the two tested controllers within the scenario i).

3.6. Validation use-cases

Besides the performance comparison above, showing the improved achieved performance of the proposed approach w.r.t. the one in [20], the developed controller has been employed in two industrial tasks to provide more insights on its usage in real production scenarios. In particular, (i) a collaborative assembly task of a gear into its shaft [5] (Fig. 13 (a)) and (ii) a collaborative deposition task [6] (Fig. 13 (b)) have been considered as target use-cases. (i) considers the robot equipped with a gripper manipulating a gear to be collaboratively assembled. (ii) considers the robot equipped with a sealing gun operated by the human along the deposition path, strictly related to the activities developed in the H2020 CS2 ASSASSINN

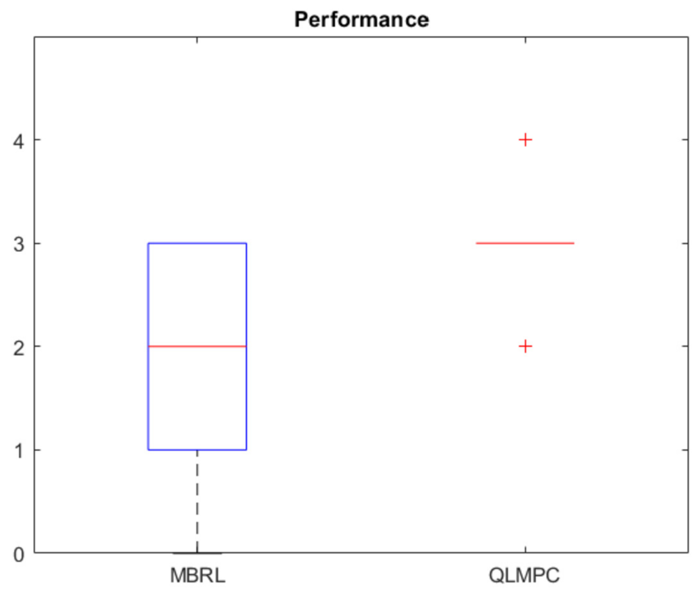


Fig. 9. *Overall performance* criterion for the two tested controllers within the scenario i).

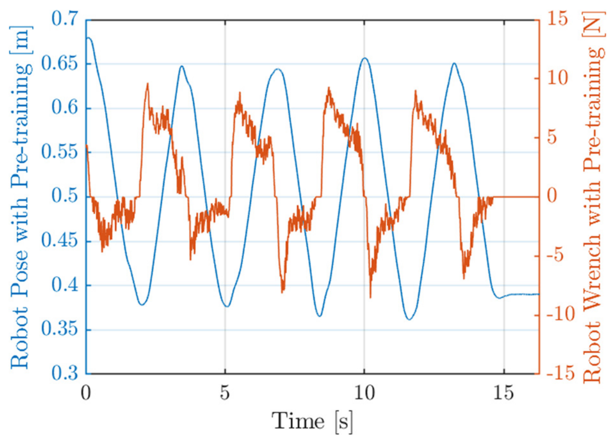


Fig. 10. Interaction force and the optimized setpoint evolutions in the considered scenario i) for a specific subject. The delay between the interaction force and the setpoint represents the reaction time of the manipulator.

project. Both the tasks can be executed to demonstrate the application to the robot (*e.g.*, in the context of programming-by-demonstration) or to collaboratively perform it.

The selected tasks allow the assessment of the adoption of the proposed controller in real human-robot collaborative industrial tasks. In fact, such tasks require a smooth human-robot interaction, having the robot capable of quickly reacting to the intention of motion of the human in order to properly execute the target task, guaranteeing the stability of the interaction. In fact, instabilities, vibrations, or delays in the controller reaction might cause task failures or unacceptable output quality. Indeed, the proposed use-cases allow the evaluation of the effectiveness of the developed controller in complex human-robot collaborative applications.

A video showing the performed assembly task is available at the link https://youtube.com/shorts/Mc-VMzbGfpA?feature=share, while a video showing the performed deposition task is available at the link https://youtube.com/shorts/QQ1hLD9Va6o?feature=share. As it can be highlighted, the proposed controller allows the implementation of a smooth and reactive robot behavior (reflecting the performance evaluation provided in the previous Sections), making it possible to perform complex human-robot collaborative tasks. Thus, the proposed controller is proven to be applicable to real industrial tasks.

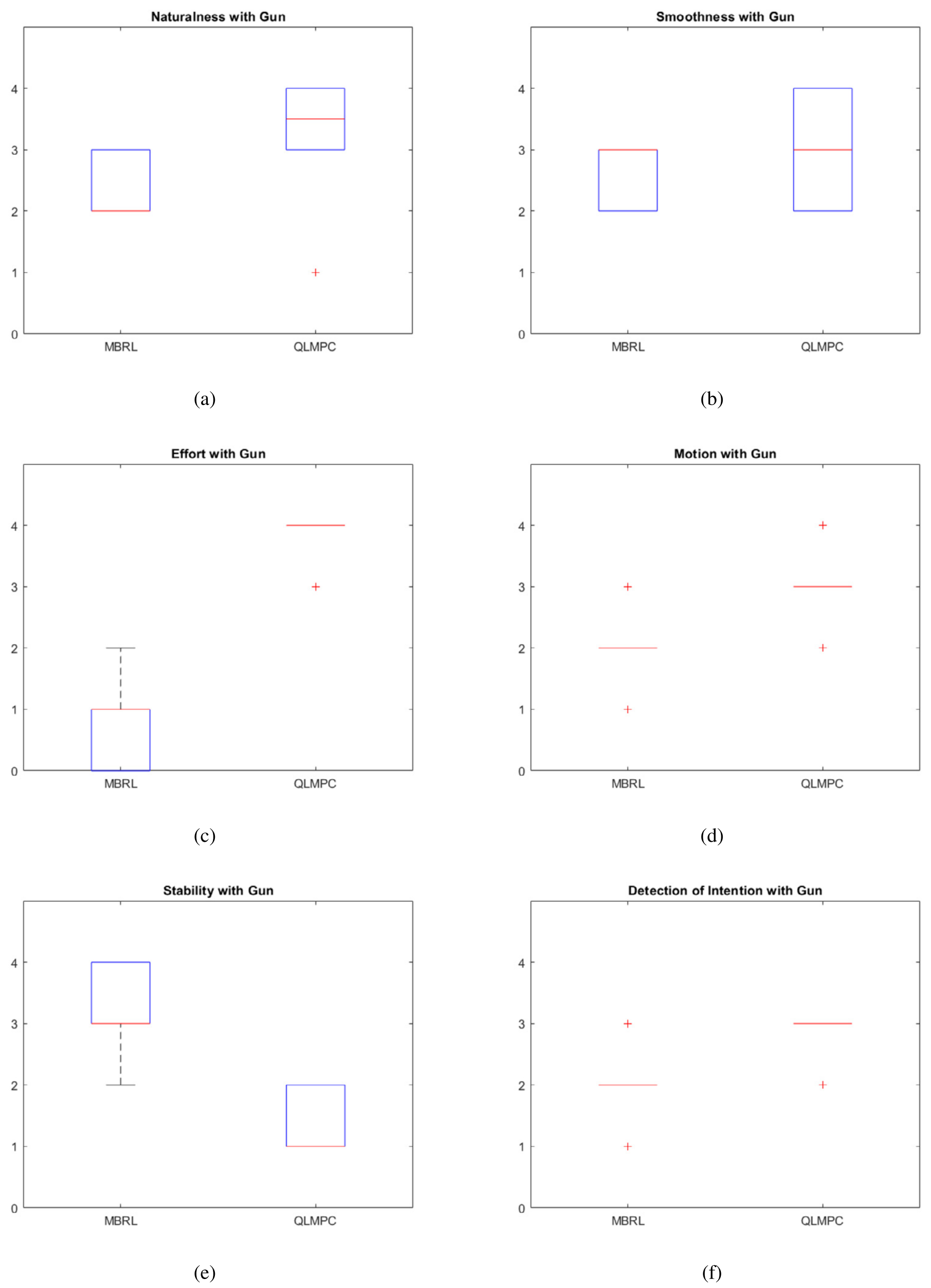


Fig. 11. (a) *Naturalness*, (b) *Smoothness*, (c) *Effort*, (d) *Motion*, (e) *Stability*, (f) *Detection of Intention* criteria for the two tested controllers with Makita Cordless Caulking Gun mounted on the manipulator within the scenario ii).

3.7. Discussion

3.7.1. Pros

The Lyapunov-based approach addressed the lack of stability issue and performance-weight sensitivity present in the previous related works (such as in [20]). As a consequence, the proposed Q-LMPVIC outperforms the approach in [20] with a minimal tuning effort. The ensemble of ANNs solves the challenges related to the difficulty and the effort required for obtaining an accurate model of a non-linear system by providing a reliable estimation of the human-robot interaction behavior. The integration of the CEM and the Q-learning for LMPC problem resolution combines the prediction capabilities

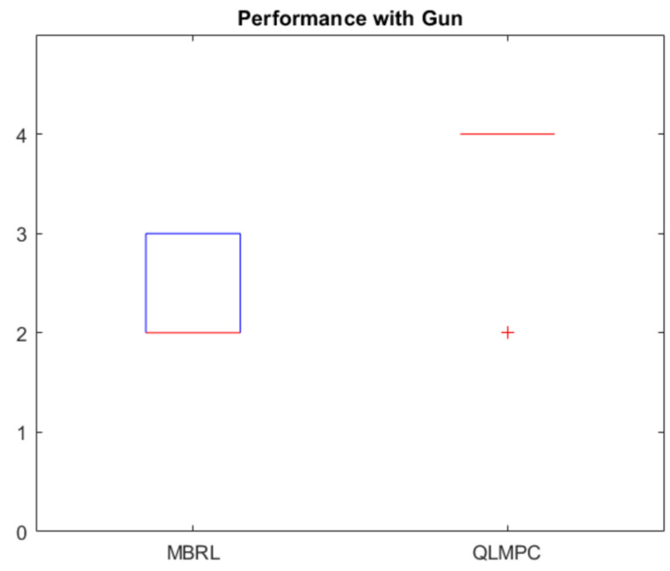
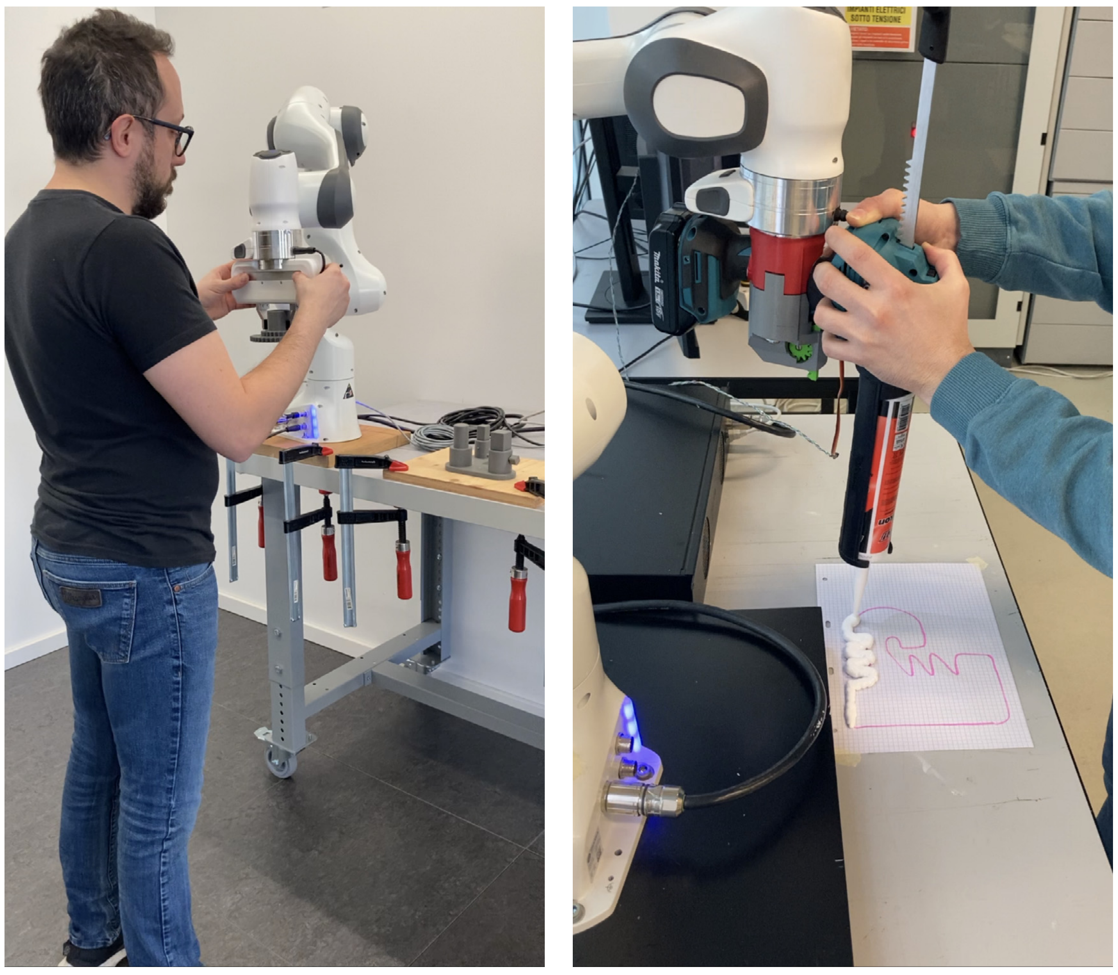


Fig. 12. *Overall performance* criterion for the two tested controllers within the scenario ii).



(a) Assembly task

(b) Deposition task

Fig. 13. The tested (a) assembly task (where the gear has to be assembled into its shaft) and (b) deposition task (where the material has to be deposited along the highlighted path) are shown.

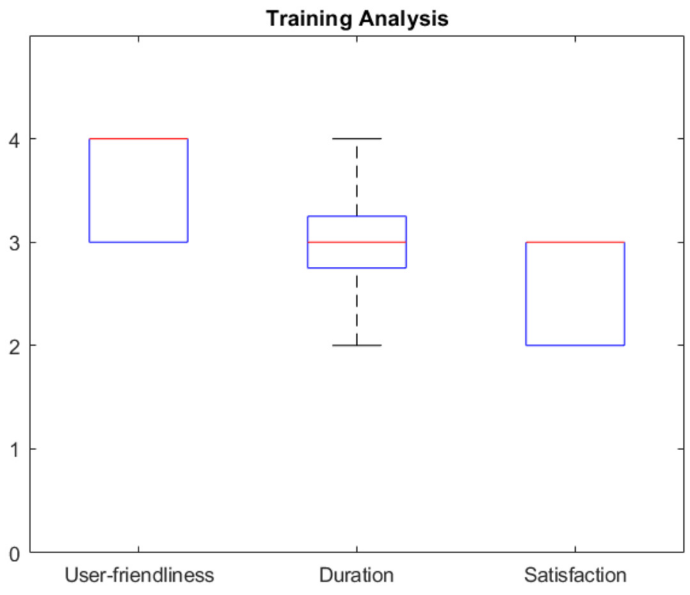


Fig. 14. Results of the training analysis.

offered by the ensemble with the limited computational burden associated with the RL techniques. In fact, the controller can run on the Linux rt kernel (which does not allow the usage of) at an acceptable frequency.

3.7.2. Cons

The hardware limitations and the constraints on the computational time generated by the necessity of online learning limit the number of ANNs that the algorithm can handle. By reducing the dimension of the model approximator ensemble, a higher risk of over-fitting is shown in the regions with few available data, together with a performance deterioration at the beginning of the training. Besides, only the interaction force is used for the operator's effort evaluation during the task, and other ergonomics analyses on the collaboration performance have not been carried out.

4. Conclusions

The presented work in this paper proposes a Q-Learning-based Model Predictive Variable Impedance Control Q-LMPVIC to assist the operators in physical human-robot collaboration (pHRC) tasks. The proposed methodology is composed of two control loops, a low-level Cartesian impedance controller, and a high-level methodology for the online optimization of impedance control parameters (*i.e.*, the setpoint and damping parameters), allowing to minimize the human-robot interaction force during the collaboration. The outer high-level controller is composed of an actor and a critic ANN, which implement a Q-Learning algorithm for the resolution of a nonlinear optimal control problem. An ensemble of ANNs is exploited to estimate the model of the system. The MPC enhanced with stability guarantees (by means of Lyapunov constraints) is, indeed, implemented to compute the low-level impedance control parameters online.

The proposed Q-LMPVIC has been tested to assess its performance, comparing it w.r.t. a MBRL variable impedance control implemented in [20]. The proposed Q-LMPVIC has been proven to improve the pHRC performance. The training process of the proposed approach has been also validated, showing high-quality performance and a limited training time. Two complex use-cases (a collaborative assembly task and a collaborative deposition task) have been additionally setup to show the applicability of the proposed approach to real industrial tasks.

Future work is devoted to increase the high-level control loop frequency (imposed equal to 6 Hz in this paper) in order to further improve the pHRC performance. To solve this issue, parallelization with modern GPUs can be exploited. In addition, external sensors (such as EMGs) can be exploited to better capture the human-robot interaction modeling and collaboration performance. The proposed strategy will also be adapted (in terms of modeling and control objectives) and applied to an exoskeleton device.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

The work has been developed within the project ASSASSINN, funded from H2020 CleanSky 2 under grant agreement n. 886977.

The work has been developed within the project ExoRescue, funded from EUREKA Eurostars under grant agreement n. E!115182.

References

[1] G. Fragapane, D. Ivanov, M. Peron, F. Sgarbossa, J.O. Strandhagen, Increasing flexibility and productivity in industry 4.0 production networks with autonomous mobile robots and smart intralogistics, Ann. Oper. Res. (2020) 1–19.
[2] S. Makris, Cooperating Robots for Flexible Manufacturing, Springer, 2021.
[3] L. Roveda, N. Castaman, S. Ghidoni, P. Franceschi, N. Boscolo, E. Pagello, N. Pedrocchi, Human-robot cooperative interaction control for the installation of heavy and bulky components, in: 2018 IEEE International Conference on Systems, Man, and Cybernetics, SMC, IEEE, 2018, pp. 339–344.
[4] L. Roveda, S. Haghshenas, M. Caimmi, N. Pedrocchi, L. Molinari Tosatti, Assisting operators in heavy industrial tasks: on the design of an optimized cooperative impedance fuzzy-controller with embedded safety rules, Frontiers in Robotics and AI 6 (2019) 75.
[5] L. Roveda, M. Magni, M. Cantoni, D. Piga, G. Bucca, Human–robot collaboration in sensorless assembly task learning enhanced by uncertainties adaptation via bayesian optimization, Robot. Auton. Syst. 136 (2021) 103711.
[6] L. Roveda, B. Maggioni, E. Marescotti, A. Shahid, A.M. Zanchettin, A. Bemporad, D. Piga, Pairwise preferences-based optimization of a path-based velocity planner in robotic sealing tasks, IEEE Robot. Autom. Lett. (2021).
[7] F. Vicentini, N. Pedrocchi, M. Beschi, M. Giussani, N. Iannacci, P. Magnoni, S. Pellegrinelli, L. Roveda, E. Villagrossi, M. Askarpour, et al., Piros: cooperative, safe and reconfigurable robotic companion for cnc pallets load/unload stations, in: Bringing Innovative Robotic Technologies from Research Labs to Industrial End-Users, Springer, 2020, pp. 57–96.
[8] R.R. Galin, R.V. Meshcheryakov, Human-robot interaction efficiency and human-robot collaboration, in: Robotics: Industry 4.0 Issues & New Intelligent Control Paradigms, Springer, 2020, pp. 55–63.
[9] L. Roveda, S. Haghshenas, A. Prini, T. Dinon, N. Pedrocchi, F. Braghin, L.M. Tosatti, Fuzzy impedance control for enhancing capabilities of humans in onerous tasks execution, in: 2018 15th International Conference on Ubiquitous Robots, UR, IEEE, 2018, pp. 406–411.
[10] A. Mauri, J. Lettori, G. Fusi, D. Fausti, M. Mor, F. Braghin, G. Legnani, L. Roveda, Mechanical and control design of an industrial exoskeleton for advanced human empowering in heavy parts manipulation tasks, Robotics 8 (3) (2019) 65.
[11] E. Magrini, A. De Luca, Hybrid force/velocity control for physical human-robot collaboration tasks, in: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2016, pp. 857–863.
[12] A. Martinez, B. Lawson, C. Durrough, M. Goldfarb, A velocity-field-based controller for assisting leg movement during walking with a bilateral hip and knee lower limb exoskeleton, IEEE Trans. Robot. 35 (2) (2018) 307–316.
[13] N. Hogan, Impedance control: an approach to manipulation, in: 1984 American Control Conference, IEEE, 1984, pp. 304–313.
[14] L. Roveda, A user-intention based adaptive manual guidance with force-tracking capabilities applied to walk-through programming for industrial robots, in: 2018 15th International Conference on Ubiquitous Robots (UR), IEEE, 2018, pp. 369–376.
[15] S.G. Khan, G. Herrmann, M. Al Grafi, T. Pipe, C. Melhuish, Compliance control and human–robot interaction: Part 1—survey, Int. J. Humanoid Robot. 11 (03) (2014) 1430001.
[16] P. Liang, C. Yang, N. Wang, Z. Li, R. Li, E. Burdet, Implementation and test of human-operated and human-like adaptive impedance controls on Baxter robot, in: Conference Towards Autonomous Robotic Systems, Springer, 2014, pp. 109–119.
[17] C. Yang, C. Zeng, C. Fang, W. He, Z. Li, A dmps-based framework for robot learning and generalization of humanlike variable impedance skills, IEEE/ASME Trans. Mechatron. 23 (3) (2018) 1193–1203.
[18] W. Kim, L. Peternel, M. Lorenzini, J. Babič, A. Ajoudani, A human-robot collaboration framework for improving ergonomics during dexterous operation of power tools, Robot. Comput.-Integr. Manuf. 68 (2021) 102084.
[19] L. Roveda, N. Pedrocchi, L.M. Tosatti, Exploiting impedance shaping approaches to overcome force overshoots in delicate interaction tasks, Int. J. Adv. Robot. Syst. 13 (5) (2016) 1729881416662771.
[20] L. Roveda, J. Maskani, P. Franceschi, A. Abdi, F. Braghin, L.M. Tosatti, N. Pedrocchi, Model-based reinforcement learning variable impedance control for human-robot collaboration, J. Intell. Robot. Syst. 100 (2) (2020) 417–433.
[21] S. Cremer, S.K. Das, I.B. Wijayasinghe, D.O. Popa, F.L. Lewis, Model-free online neuroadaptive controller with intent estimation for physical human–robot interaction, IEEE Trans. Robot. 36 (1) (2019) 240–253.
[22] C. Gaz, E. Magrini, A. De Luca, A model-based residual approach for human-robot collaboration during manual polishing operations, Mechatronics 55 (2018) 234–247.
[23] F. Dimeas, N. Aspragathos, Reinforcement learning of variable admittance control for human-robot co-manipulation, in: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2015, pp. 1011–1016.
[24] A. Kukker, R. Sharma, Stochastic genetic algorithm-assisted fuzzy q-learning for robotic manipulators, Arab. J. Sci. Eng. (2021) 1–13.
[25] C. Li, Z. Zhang, G. Xia, X. Xie, Q. Zhu, Efficient force control learning system for industrial robots based on variable impedance control, Sensors 18 (8) (2018) 2539.
[26] J.R. Medina, H. Börner, S. Endo, S. Hirche, Impedance-based gaussian processes for modeling human motor behavior in physical and non-physical interaction, IEEE Trans. Biomed. Eng. 66 (9) (2019) 2499–2511.
[27] H. Gomi, R. Osu, Task-dependent viscoelasticity of human multijoint arm and its spatial characteristics for interaction with environments, J. Neurosci. 18 (21) (1998) 8965–8978.
[28] E. Noohi, M. Žefran, J.L. Patton, A model for human–human collaborative object manipulation and its application to human–robot interaction, IEEE Trans. Robot. 32 (4) (2016) 880–896.
[29] L. Peternel, N. Tsagarakis, A. Ajoudani, A human–robot co-manipulation approach based on human sensorimotor information, IEEE Trans. Neural Syst. Rehabil. Eng. 25 (7) (2017) 811–822.
[30] Y. Li, S.S. Ge, Human–robot collaboration based on motion intention estimation, IEEE/ASME Trans. Mechatron. 19 (3) (2013) 1007–1014.
[31] L. Grüne, J. Pannek, Nonlinear model predictive control, in: Nonlinear Model Predictive Control, Springer, 2017, pp. 45–69.
[32] W.-L. Ma, S. Kolathaya, E.R. Ambrose, C.M. Hubicki, A.D. Ames, Bipedal robotic running with durus-2d: bridging the gap between theory and experiment, in: Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control, 2017, pp. 265–274.
[33] R.A. Freeman, P.V. Kokotovic, Inverse optimality in robust stabilization, SIAM J. Control Optim. 34 (4) (1996) 1365–1391.
[34] A. Jadbabaie, J. Yu, J. Hauser, Unconstrained receding-horizon control of nonlinear systems, IEEE Trans. Autom. Control 46 (5) (2001) 776–783.
[35] A. Jadbabaie, J. Hauser, On the stability of receding horizon control with a general terminal cost, IEEE Trans. Autom. Control 50 (2005) 674–678.

[36] J.A. Primbs, V. Nevistic, J.C. Doyle, A receding horizon generalization of pointwise min-norm controllers, IEEE Trans. Autom. Control 45 (5) (2000) 898–909.
[37] R. Grandia, A.J. Taylor, A. Singletary, M. Hutter, A.D. Ames, Nonlinear model predictive control of robotic systems with control Lyapunov functions, preprint, arXiv:2006.01229, 2020.
[38] H. Zhang, S. Li, Y. Zheng, Q-learning-based model predictive control for nonlinear continuous-time systems, Ind. Eng. Chem. Res. 59 (40) (2020) 17987–17999.
[39] T. Binazadeh, M.A. Rahgoshay, Robust output tracking of a class of non-affine systems, Syst. Sci. Control Eng. 5 (1) (2017) 426–433.
[40] F. Caccavale, C. Natale, B. Siciliano, L. Villani, Six-dof impedance control based on angle/axis representations, IEEE Trans. Robot. Autom. 15 (2) (1999) 289–300.
[41] L. Sciavicco, B. Siciliano, Modelling and Control of Robot Manipulators, Springer Science & Business Media, 2012.
[42] K. Chua, R. Calandra, R. McAllister, S. Levine, Deep reinforcement learning in a handful of trials using probabilistic dynamics models, preprint, arXiv:1805.12114, 2018.
[43] A. Nagabandi, G. Kahn, R.S. Fearing, S. Levine, Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning, in: 2018 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2018, pp. 7559–7566.
[44] P. Mhaskar, N.H. El-Farra, P.D. Christofides, Predictive control of switched nonlinear systems with scheduled mode transitions, IEEE Trans. Autom. Control 50 (11) (2005) 1670–1680.
[45] P. Mhaskar, N.H. El-Farra, P.D. Christofides, Stabilization of nonlinear systems with state and control constraints using Lyapunov-based predictive control, Syst. Control Lett. 55 (8) (2006) 650–659.
[46] M. Heidarinejad, J. Liu, P.D. Christofides, Economic model predictive control of nonlinear process systems using Lyapunov techniques, AIChE J. 58 (3) (2012) 855–870.
[47] Z. Artstein, Stabilization with relaxed controls, Nonlinear Anal., Theory Methods Appl. 7 (11) (1983) 1163–1173.
[48] D. Munoz de la Pena, P.D. Christofides, Lyapunov-based model predictive control of nonlinear systems subject to data losses, IEEE Trans. Autom. Control 53 (9) (2008) 2076–2089.
[49] E.D. Sontag, A 'universal' construction of Artstein's theorem on nonlinear stabilization, Syst. Control Lett. 13 (2) (1989) 117–123.
[50] R.A. Freeman, J.A. Primbs, Control Lyapunov functions: new ideas from an old source, in: Proceedings of 35th IEEE Conference on Decision and Control, vol. 4, IEEE, 1996, pp. 3926–3931.
[51] D. Lakatos, F. Petit, P. Van Der Smagt, Conditioning vs. excitation time for estimating impedance parameters of the human arm, in: 2011 11th IEEE-RAS International Conference on Humanoid Robots, IEEE, 2011, pp. 636–642.
[52] A.F. Filippov, Differential Equations with Discontinuous Righthand Sides: Control Systems, vol. 18, Springer Science & Business Media, 2013.
[53] D. Shevitz, B. Paden, Lyapunov stability theory of nonsmooth systems, IEEE Trans. Autom. Control 39 (9) (1994) 1910–1914.
[54] M. Vidyasagar, Nonlinear Systems Analysis, SIAM, 2002.
[55] F.L. Lewis, D. Vrabie, Reinforcement learning and adaptive dynamic programming for feedback control, IEEE Circuits Syst. Mag. 9 (3) (2009) 32–50.
[56] Q. Wei, D. Liu, G. Shi, A novel dual iterative q-learning method for optimal battery management in smart residential environments, IEEE Trans. Ind. Electron. 62 (4) (2014) 2509–2518.
[57] R. Padhi, N. Unnikrishnan, X. Wang, S. Balakrishnan, A single network adaptive critic (snac) architecture for optimal control synthesis for a class of nonlinear systems, Neural Netw. 19 (10) (2006) 1648–1660.
[58] C.J.C.H. Watkins, P. Dayan, Q-Learning, Mach. Learn. 8 (1992) 279–292.
[59] X. Xu, H. Chen, C. Lian, D. Li, Learning-based predictive control for discrete-time nonlinear systems with stochastic disturbances, IEEE Trans. Neural Netw. Learn. Syst. 29 (12) (2018) 6202–6213.
[60] O. Sprangers, R. Babuška, S.P. Nageshrao, G.A. Lopes, Reinforcement learning for port-Hamiltonian systems, IEEE Trans. Cybern. 45 (5) (2014) 1017–1027.
[61] J. Brownlee, Clever algorithms: nature-inspired programming recipes, Jason Brownlee, 2011.
[62] G.A. Rummery, M. Niranjan, On-Line Q-Learning Using Connectionist Systems, vol. 37, Citeseer, 1994.
[63] S. Mizanoor Rahman, R. Ikeura, Cognition-based control and optimization algorithms for optimizing human-robot interactions in power-assisted object manipulation, J. Inf. Sci. Eng. 32 (5) (2016).
[64] L. Bottou, Stochastic gradient descent tricks, in: Neural Networks: Tricks of the Trade, Springer, 2012, pp. 421–436.
[65] D.P. Kingma, J. Ba, Adam: a method for stochastic optimization, preprint, arXiv:1412.6980, 2014.
[66] L. Roveda, A. Bussolan, F. Braghin, D. Piga, 6d virtual sensor for wrench estimation in robotized interaction tasks exploiting extended Kalman filter, Machines 8 (4) (2020) 67.
[67] L. Roveda, D. Piga, Sensorless environment stiffness and interaction force estimation for impedance control tuning in robotized interaction tasks, Auton. Robots 45 (3) (2021) 371–388.
[68] L. Roveda, A.A. Shahid, N. Iannacci, D. Piga, Sensorless optimal interaction control exploiting environment stiffness estimation, IEEE Trans. Control Syst. Technol. 30 (1) (2021) 218–233.
[69] L. Roveda, A. Bussolan, F. Braghin, D. Piga, Robot joint friction compensation learning enhanced by 6d virtual sensor, Int. J. Robust Nonlinear Control (2022).