Experimental Assessment of Deep Reinforcement Learning for Robot Obstacle Avoidance: A LPV Control Perspective

Gian Paolo Incremona * **Nikolas Sacchi** **
Bianca Sangiovanni *** **Antonella Ferrara** ***

* *Politecnico di Milano, Piazza Leonardo da Vinci, 32, 20133 Milano, Italy (e-mail: gianpaolo.incremona@polimi.it)*
** *University of Genova, via alla Opera Pia 13, 16100, Genova, Italy (e-mail: nikolas.sacchi@dibris.unige.it)*
*** *University of Pavia, Via Ferrata 5, 27100 Pavia, Italy (e-mail: bianca.sangiovanni01@universitadipavia.it, antonella.ferrara@unipv.it).*

Abstract: This work presents the experimental assessment of a hybrid control scheme based on Deep Reinforcement Learning (DRL) for obstacle avoidance in robot manipulators. More precisely, relying on an equivalent Linear Parameter Varying (LPV) state-space representation of the system, two operative modes, one based on both joint positions and velocities, one only based on velocity inputs, are activated depending on the measurement of the distance between the robot and the obstacle. Therefore, when the obstacle is close to the robot, a switching mechanism is introduced to enable the DRL algorithm instead of the basic motion planner, thus giving rise to a self-configuring architecture to cope with objects randomly moving in the workspace. The experimental tests of the DRL based collision avoidance hybrid strategy are carried out on a physical EPSON VT6 robot manipulator with satisfactory results.

Keywords: deep reinforcement learning, robot control, collision avoidance, LPV

1. INTRODUCTION

Learning control is a topic with a long history, which is very much evolving over time, thus giving rise to new methods in systems and control. Indeed, the number of applications that have been experimentally validated on laboratory test-beds and in some industrial context is increasing. In this work, in particular, we focus on the robotic applications where the use of learning techniques is very promising, above all in cases where robots need to cooperate with human operators [Bicchi et al., 2008; De Santis et al., 2008]. More specifically, two aspects deserve attention and require efficient and safety-oriented control strategies. On one hand, although the computing power of current devices is growing, in certain working conditions lighter computational solutions might be alternated with more computationally onerous methods needed only when a high performance is required. On the other hand, the presence of human operators makes an efficient motion planner and collision avoidance mechanism mandatory [De Luca and Flacco, 2012].

Among the possible approaches to cope with the previous issues, recent promising alternatives are given by data-driven optimal control, or by Reinforcement Learning (RL) approaches [Brüggemann and Possieri, 2021; Possieri et al., 2021; Kober et al., 2014]. Differently from optimal control problems, RL techniques rely on measured data and instant feedback by interacting with the environment, thus enabling the considered plant (the robot in our case) to autonomously discover an optimal behaviour through optimizing a reward function. In order to alleviate the limitations related to practical RL training times and the possible high number of involved data, general-purpose Deep Neural Networks (DNN) are adopted [LeCun et al., 2015].

As for robotics applications, RL has been successfully applied, for instance, in [Levine et al., 2017; Lenz et al., 2015], where robots have been trained to grasp sparse objects through convolutional neural networks for pose estimation, in [Deisenroth et al., 2011] for dexterity operations, or in [Haarnoja et al., 2018] to accomplish a stacking task by using soft Q-learning. The main idea in [Sangiovanni et al., 2018a; Ferrara et al., 2019] is instead that of introducing a switch between pre-specified control approaches to enable the robot to solve a given task in an unpredictable environment. To this end, a DRL based decision mechanism is introduced to select the most suitable control architecture between a decentralized control scheme and a centralized one based on the inverse dynamics. Collision avoidance approaches based on end-to-end DRL and on a hybrid learning method are proposed in [Sangiovanni et al., 2018b], and [Sangiovanni et al., 2021], respectively.

⋆ This is the final version of the accepted paper submitted for inclusion in the Proceedings of the 4th IFAC Workshop on Linear Parameter Varying Systems LPVS, Milan, Italy, Jul. 2021.

1.1 Contributions

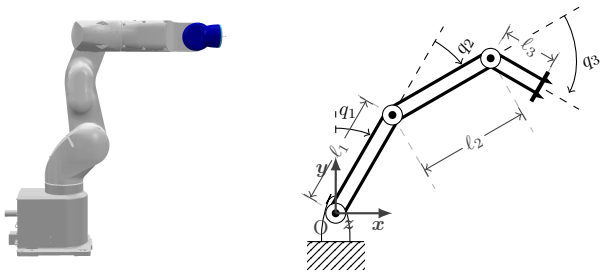
Making reference to [Sangiovanni et al., 2021], in this paper we present the experimental assessment of the hybrid DRL based collision avoidance approach therein introduced and validated only in simulation. A LPV perspective of the overall control scheme is also discussed in this work. The main idea is that of suitably selecting the best motion control approach in order to reach the desired target in an operative space invaded by moving obstacles [Sangiovanni et al., 2018b; Sangiovanni et al., 2021]. Since a switching mechanism is adopted, this determines a time varying closed-loop robot control scheme, thus resulting equivalent to a LPV system whose dependency is related to the joint variables, its derivatives, and the measurement of the distance between the obstacles and the robot full body. Successful experimental results carried out on an anthropomorphic robot manipulator EPSON VT6 are finally illustrated.

1.2 Organization of the paper

This paper is organized as follows. The model of the robot environment and some preliminaries on DRL are introduced in Section 2. The adopted hybrid DRL based approach is recalled in Section 3, together with a discussion on the LPV nature of the considered scheme. In Section 4 the EPSON VT6 anthropomorphic robot manipulator setup is described, and experimental results are illustrated and commented. Some concluding remarks are reported in Section 5.

2. MODELING AND PROBLEM FORMULATION

In this section the model of the considered robot manipulator is introduced and the problem to solve is described. Then, having in mind to adopt the DRL based hybrid approach in [Sangiovanni et al., 2021], some preliminaries on RL are recalled.



(a) (b)

Fig. 1. EPSON VT6. (a) Physical setup. (b) Schematic view of the robot with three joints

2.1 Model of the robotic environment

In this work we consider the robotic environment composed of the robot manipulator and the obstacles, possibly of different shape, randomly invading the workspace. More specifically, let $\mathcal{O} \in \mathbb{R}^{n_\mathcal{O}}$ represent the information about the position of such obstacles moving around the robot workspace. Note that, the scalar $n_\mathcal{O}$ is related to the obstacle representation, which can be for instance the number of coordinates corresponding to the vertices of a polygonal object.

Consider Figure 1, and let $q \in \mathbb{R}^{n_q}$ be the joint variables, with n_q being the number of degrees of freedom. The kinematics of the robot between the joint variables q and the end-effector vector $x_\mathrm{e} \in \mathbb{R}^{6\times 1}$, is therefore defined as

$$x_\mathrm{e} = k(q) \ , \tag{1}$$

where $k(q) \in \mathbb{R}^6$.

Relying instead on the Lagrangian equations [Siciliano et al., 2009], the dynamics of the robot is captured by the following model

$$B(q)\ddot{q} + C(q,\dot{q})\dot{q} + F_\mathrm{v}\dot{q} + F_\mathrm{s}\,\mathrm{sgn}(\dot{q}) + g(q) = \tau \ , \tag{2}$$

where $B(q) \in \mathbb{R}^{n_q\times n_q}$ is the inertia matrix, $C(q,\dot{q}) \in \mathbb{R}^{n_q\times n_q}$ represents centripetal and Coriolis torques, $F_\mathrm{v} \in \mathbb{R}^{n_q\times n_q}$ is the viscous friction matrix, $F_\mathrm{s} \in \mathbb{R}^{n_q\times n_q}$ is the static friction matrix, $g(q) \in \mathbb{R}^{n_q}$ is the vector of gravitational torques, and $\tau \in \mathbb{R}^{n_q}$ represents the motors torques.

We assume that the joint positions q and velocities $\dot{q}$ can be measured, and a vision system is adopted to observe the robot environment. In particular, the vision sensor is assumed to provide the end-effector position $p_\mathrm{e} \in \mathbb{R}^3$, as well as position and velocity of the obstacles, namely $p_\mathrm{O} \in \mathbb{R}^3$ and $\dot{p}_\mathrm{O} \in \mathbb{R}^3$. This allows to achieve instant by instant the minimum distance, namely $d_\mathrm{O} \in \mathbb{R}$, between the obstacles position and the robot full body.

2.2 Collision avoidance control problem

Given the model (2), where the joints are also subject to input and state constraints defined as $h(q,\dot{q}) \leq 0$, with $h : \mathbb{R}^{n_q}\times\mathbb{R}^{n_q} \mapsto \mathbb{R}^\ell$, and ℓ being the number of constraints, the objective is to make the robot move towards a target or follow a desired trajectory (e.g., to execute an industrial task such as spot welding, pick and place, or point-to-point motions), while avoiding obstacles invading the workspace. Precisely, this latter requirement can be formulated as

$$\mathcal{W}(q(t)) \cap \mathcal{O} \equiv \varnothing \ , \tag{3}$$

where $\mathcal{W}(q(t)) \subset \mathbb{R}^6$ is the space occupied by the robot at the instant t. In other words, our goal becomes to find a suitable velocity input sequence $\{\dot{q}^\star\}$, which makes the robot end-effector perfectly reach a target point $x^\star$ from the initial state $x_{\mathrm{e}0}$, while avoiding the obstacles $\mathcal{O} \subset \mathbb{R}^{n_\mathcal{O}}$.

As shown in [Sangiovanni et al., 2021], the previous problem can be written as a finite-horizon Collision Avoidance Optimal Control Problem (CAOCP), whose solving difficulties are however given by the presence of the collision avoidance constraint (3), which in principle could be non-convex and non-differentiable. The control objective can be therefore achieved relying on a learning model-free method.

2.3 Preliminaries on DRL

For the readers' convenience, some preliminaries on DRL are hereafter recalled.

RL is based on the fact that the involved *agent* can understand from past experience which *actions* will lead

to maximize a function, namely the *cumulative reward*, in a finite or even infinite temporal horizon given a current *state*. In other words, a RL approach aims to maximize not only the best current reward, but also the best future ones.

Let us introduce the main notation and elements of RL. Given a Markov Decision Process (MDP), let $s_t \in \mathcal{S}$ be the *state* of the agent at the time t with $\mathcal{S}$ being the state space, and let $a_t \in \mathcal{A}$ be the *action* by the agent given a state, with $\mathcal{A}$ being the action space. Then, let $P(s_{t+1} \mid a_t, s_t)$ be the *transition probability* function (also called *model*), which rules the state changes through the actions operated by the agent. A transition to state s_{t+1} triggered by an action a_t determines a *reward* $r_t \in \mathcal{R}$ to the agent given by a function $r_t = r(a_t, s_t)$, which says "how well" an agent has done at step t. For episodic tasks, involving a *sequence* of actions, the cumulative reward R_t is instead given by

$$R_t = \sum_{k=0}^{T} \gamma^k r_{t+k+1} \,, \tag{4}$$

where $0 \leq \gamma \leq 1$ is called *discount rate* and is used to prioritize earlier rewards over later ones. The mapping from states to action is given by a *policy* $\pi : \mathcal{S} \mapsto \mathcal{A}$, that can be either deterministic or probabilistic. The goal is to find the optimal policy $\pi^\star$ that maximizes the *expected* cumulative reward. More specifically, given a certain policy π, the *value function* to optimize $V^\pi : \mathcal{S} \mapsto \mathbb{R}$ is the expected cumulative reward accumulated by the agent by adopting π from a given state on. During the training process, for each *episode* the agent interacts with the environment for either a complete attempt to perform a goal task or a fixed number of time-steps before being reset.

Since the robotic application at hand, which involves structural variations of the robot and/or a rapidly changing environment, has a model that cannot be completely known in advance, the so-called Q-learning approach is applied. Specifically, let $Q^\pi(s, a)$ be the so-called *action value function* or *Q-function*. This learning method allows the convergent, incremental updating of a suitable approximation $\tilde{Q}$ towards the optimal $Q^\star$, as experience progresses. Since computing $Q^\star$ can become computationally expensive in case of continuous action problems with a large number of states, a possible solution is the use of a parametric approximator $\tilde{Q}$, such as a DNN, that is

$$\tilde{Q}\left(s, a|\theta^Q\right) = \tilde{A}\left(s, a|\theta^A\right) + \tilde{V}\left(s|\theta^V\right) \,, \tag{5}$$

where $\tilde{A}$ and $\tilde{V}$ are parametric approximators of the *advantage function* $A^\pi(s_t, a_t) = Q^\pi(s_t, a_t) - V^\pi(s_t)$ and the value function $V^\pi(s_t)$, respectively, and θ^A and θ^V the corresponding vectors of parameters. The learning process aims to minimize a *loss function* given by

$$L(\theta^Q) = \mathbb{E}_{r,\nu^\beta,\beta}\left[\left(\tilde{Q}\left(s_t, a_t|\theta^Q\right) - y_t\right)^2\right] \,, \tag{6}$$

where $y_t = r(s_t, a_t) + \gamma\tilde{Q}\left(s_{t+1}, \mu(s_{t+1})|\theta^Q\right)$ is the target, θ^Q are parameters of the action-value function, β is a stochastic behaviour policy such that $a_t = \beta(s_t)$, and ν^β is the state visitation frequency with policy β. Finally, in order to achieve the policy

$$\tilde{\mu}(s_t, a_t) = \arg\max_a \tilde{Q}(s_t, a_t) \,, \tag{7}$$

the Normalized Advantage Functions (NAF) algorithm is adopted (see [Gu et al., 2016] and [Possieri et al., 2021] for further details). The latter approximates the value $\tilde{Q}$ as a quadratic function for which the optimal output can be directly computed.

3. THE ADOPTED APPROACH

In this section the used hybrid obstacle avoidance strategy is recalled and a LPV control perspective is also provided.

3.1 Hybrid DRL approach

The collision avoidance approach experimentally assessed for the first time in this paper is that in [Sangiovanni et al., 2021]. The strategy consists of a switching algorithm implementing both off-line motion planning and DRL-based control, in order to enable a robot manipulator to perform a reaching task whilst avoiding moving obstacles. More in detail, given an initial robot configuration and a target point in space, a feasible trajectory $(q^\star, \dot{q}^\star)$ is computed off-line, using standard motion planning algorithms, not considering the presence of obstacles. This allows for a generally faster trajectory generation, as the target is defined inside the robot's operative space. In the case of the paper the so-called Single-Query Bi-Directional Probabilistic Roadmap Planner with Lazy Collision Checking (SBL) algorithm is used.

Then, as the robot moves according to the position, velocity and acceleration profiles generated during the planning phase, the environment is sampled continuously. In case an obstacle is detected within a certain critical distance from the robot body, or according to any other suitable switching metric, the motion is halted and the control is switched to the end-to-end DRL control module. In the considered case, the switching rule is given by

$$d_{\mathrm{O}} \leq \epsilon \,, \tag{8}$$

where ϵ is the minimum distance threshold. The learning control is performed by deploying the policy computed by an agent trained with a model-free DRL algorithm. This approach, instead of performing trajectory planning and generation, directly outputs low-level joint control commands at each time step, given the environment observations used at training time. Specifically, the so-called state space is selected as

$$\mathcal{S} = \{q,\, \dot{q},\, p_{\mathrm{e}},\, p^\star,\, p_{\mathrm{O}},\, \dot{p}_{\mathrm{O}}\} \,, \tag{9}$$

with $p^\star$ being the target position, while the action space is given by the reference velocities for each joint, i.e.,

$$\mathcal{A} = \{\dot{q}^\star\} \,. \tag{10}$$

Therefore, the agent is trained using the objective (*reward*) function defined as

$$r = -(c_1 r_{\mathrm{T}} + c_2 r_{\mathrm{O}} + c_3 r_{\mathrm{A}}) \,, \tag{11}$$

i.e., the weighted sum of three terms. The first one takes into account the Euclidean distance d between the end-effector and the target (the so-called *Huber-Loss function*)

$$r_{\mathrm{T}} = \begin{cases} \frac{1}{2}d^2 & \text{for } d < \delta \\ \delta\left(d - \frac{1}{2}\delta\right) & \text{otherwise} \end{cases} \,, \tag{12}$$

with $\delta \in \mathbb{R}$ being a suitable selected scalar. The second one is the distance between the obstacle and the robot,

$$r_{\mathrm{O}} = \min\left(k_r \frac{r_{\mathrm{T}}}{c_2}; \left(\frac{d^\star}{d_{\mathrm{O}} + d^\star}\right)^g\right) \,, \tag{13}$$

where $d^\star$ is a constant parameter ensuring that $0 < r_\mathrm{O} < 1$, g is a hyperparameter [Sangiovanni et al., 2018b], and k_r is a positive integer that ensures a minimum, fixed proportion between r_O and r_T. Finally, the third one is the magnitude of the actions performed, i.e.,

$$r_\mathrm{A} = \|a\|^2 \,, \tag{14}$$

which has the scope to achieve smoother inputs. The value of the weights c_1, c_2 and c_3 are instead defined through repeated tests.

This hybrid switched methodology allows exploiting the advantages of both classical algorithms and DRL based control, in order to ensure a higher degree of stability, such as when following a pre-defined path, and at the same time reducing the hand-engineering for complex operation, such as full-body collision avoidance. In the presented case study, indeed, it has been observed that using a classical motion planner in combination with DRL strategy ensures better performances in terms of tracking precision with respect to the genuine end-to-end DRL controller. In fact, the DRL policy introduces some randomicity elements to the actions performed, thus causing jittering when little movement is required. On the other hand, the activation of the DRL approach allows avoiding the recomputation of the classical motion planner every time the obstacle closely invades the workspace of the robot.

3.2 LPV perspective

After the description of the adopted strategy, in this paper we would like to highlight the LPV nature of the considered controlled robotic system. Indeed, referring to [Halalchi et al., 2010], letting $z = [\tilde{q}^\top, \dot{\tilde{q}}^\top]^\top$, with $\tilde{q} = q^\star - q$ and $\dot{\tilde{q}} = \dot{q}^\star - \dot{q}$, the considered system (2) can be written as the following state-space LPV form

$$\begin{cases} \dot{z}(t) = A(z(t))z(t) + B(z(t))u(t) \\ y(t) = z(t) \end{cases} , \tag{15}$$

where $A(z(t))$ are $B(z(t))$ are suitable defined matrices (see e.g., [Halalchi et al., 2010]) with the dependence on the measured joint positions and velocities. In our case, we can

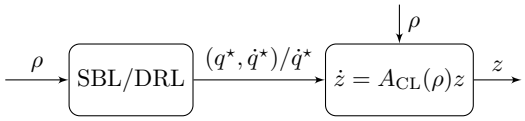


Fig. 2. LPV control representation of the hybrid obstacle avoidance strategy

further rearrange the previous system, assuming first that local joints controllers are capable to track the desired joint path. This assumption is realistic as we have verified in the experimental assessment described in the next section, where the inner controllers of the robot perfectly track the desired input trajectories. According to the adopted hybrid strategy, the reference inputs of the closed-loop system are different, and its linear dynamics changes due to the fact that in case of standard motion planning a cascade control scheme with velocity and position control loops is activated, otherwise only the velocity control loop is used for the DRL based mode. Therefore, letting $\rho = [z^\top, d_\mathrm{O}]^\top$ be the vector of measured parameters, and $u = K(\rho)z$ be the whole dynamic controller of the robot, one has that

$$\begin{cases} \dot{z}(t) = (A(z(t)) + B(z(t))K(\rho(t)))z(t) = A_\mathrm{CL}(\rho(t))z(t) \\ y(t) = z(t) \,, \end{cases} \tag{16}$$

where $A_\mathrm{CL}(\rho(t))$ is the closed-loop matrix of the equivalent LPV system, for which the parametric dependence is with respect to the time-varying parameter vector ρ. More precisely, it depends on the measured distance which determines not only the dynamics (16), but also the suitable input reference to reach the target, while avoiding obstacles (see Figure 2). In other words, the hybrid scheme with DRL gives rise to a LPV system governed by ρ, as equivalently done by the switching rule (8).

4. EXPERIMENTAL VALIDATION

In the following, the interfacing system predisposed for the deployment of the adopted methodology on an EPSON VT6, a 6-axis industrial manipulator, is described in detail. Then, a case study featuring a spot-welding task is introduced and experimental results are reported.

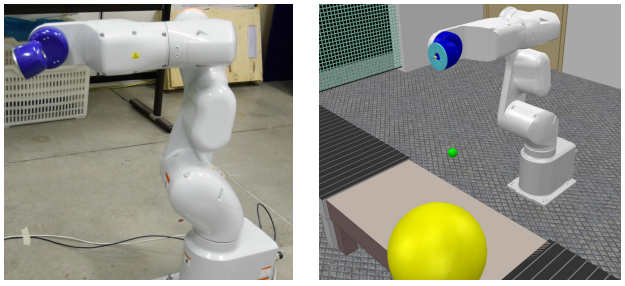
4.1 Settings

The EPSON VT6, placed in an environment together with a randomly moving obstacle, needs to perform the following two tasks:

T1) to track a desired trajectory in order to reach a pre-specified target;
T2) to avoid collisions with elements invading the robot workspace, while reaching the target.

To do this, the hybrid obstacle avoidance strategy discussed in Section 3 is used. The scenario considered reproduces a spot-welding task, that is the robot must track a target that moves along a path, on which it advances after being reached.

The robot EPSON VT6, illustrated in Figure 3(a), is designed for industrial applications and features built-in controllers. It has an operation range of 900 mm, and can support a payload up to 6 kg. Moreover, the motion has a repeatability of 0.1 mm for all joints, whose specifications are reported in Table 1.



(a) lab setup (b) simulator

Fig. 3. Working environment. (a) Lab setup. (b) Simulation scenario with spherical obstacle (yellow) and target point (green)

Table 1. EPSON VT6 joint specifications

Joint	q (deg)	$\dot{q}$ (deg s^{-1})	τ (N m^{-1})
1	[-170,170]	±166.2	50
2	[-160,65]	±122.2	50
3	[-51,190]	±118.8	50
4	[-200,200]	±271.4	12
5	[-125,125]	±296.8	12
6	[-360,360]	±293.2	7

4.2 System interface

The commands to the robot are given via EPSON RC+, which is the EPSON's proprietary software to directly interface a computer with the robot's controllers, through USB or Ethernet. Besides managing the proper setting and communications of the robotic application, the software features a proprietary IDE for high-level development of programs for the robot's controllers. As it is designed for industry applications, the language features intuitive, easy to interpret commands to define routines and robot motions, in order to facilitate the operations by employees.

For the execution of the experiment, the EPSON VT6 proprietary control software, EPSON RC+ 7.0, communicates through TCP/IP with a virtual environment reproducing the considered scenario (see Figure 3(b)), in order to sample the movements of the simulated robot and reproduce them on the physical system. The virtual environment is instead directly interfaced with the control algorithm through a dedicated API.

4.3 Experimental results

The results achieved with the real EPSON VT6 industrial manipulator are hereafter reported and compared to those of the virtual version.

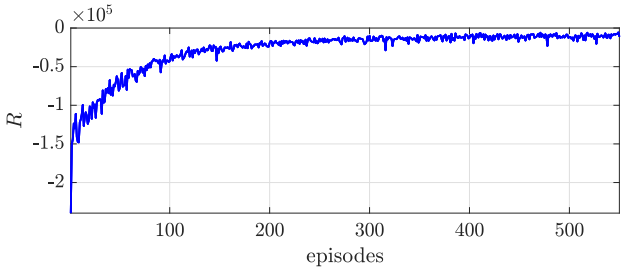


Fig. 4. Cumulative reward function over 550 episodes

The policy used for DRL control of collision avoidance is selected after 550 episodes of training on the EPSON VT6 model. Each episode has a duration of 18 s, with a time-step duration of 50 ms (i.e., a total of 360 steps per episode). The cumulative reward function is represented in Figure 4, where the convergence towards zero indicates a successful learning process.

Figure 5 shows instead the time evolution of the end-effector position for the three axes. One can observe that the position of the real robot (cyan solid line), obtained through the presented interfacing architecture, features a satisfactory level of fidelity with what is obtained in simulation (blue solid line), with a delay of about 0.5 s, thus allowing a reliable deployment of the proposed approach. A rendering of the motion of the physical EPSON

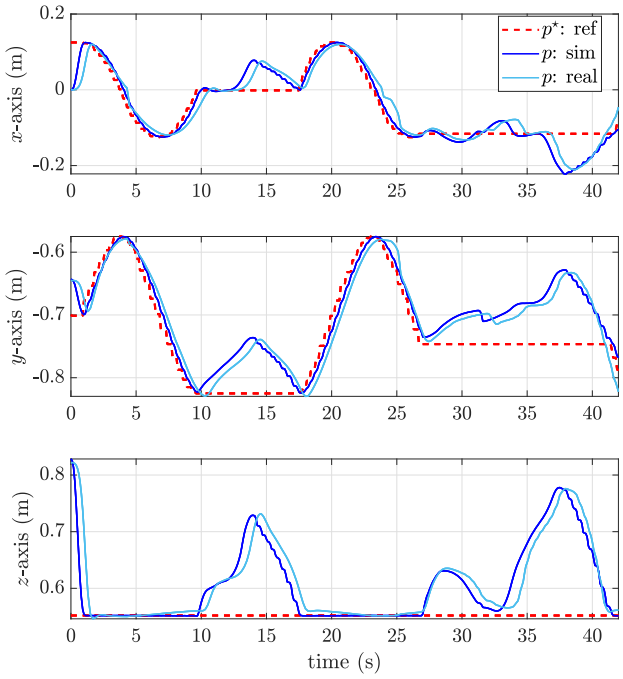


Fig. 5. Position of the end-effector of the EPSON VT6 industrial manipulator: virtual (blue solid line) and real (cyan solid line) position with respect to the reference (red dashed line)

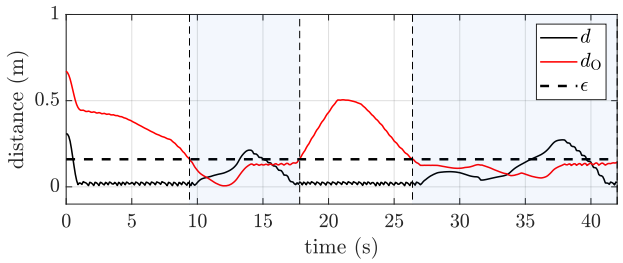


Fig. 6. Distances between end-effector and target point (black solid line), and between robot and obstacle (red solid line), in case of the hybrid approach (with DRL used in the shadow windows when $d_O < \epsilon$)

VT6 with respect to the simulated environment during the task execution is illustrated in Figure 7, where it is evident the synchronism of the real environment with the virtual one containing the obstacle. Finally, in Figure 6 the distance from the target point and the distance between robot and obstacle are reported, with DRL used in the shadow windows when $d_O < \epsilon$, and $\epsilon = 0.16$. To conclude, satisfactory results are obtained, with the robot capable to precisely reach the target by using the SBL algorithm (for instance in the intervals between 1 and 10 s, and between 17 and 27 s), while moving far from it whenever the obstacle invades the robot workspace.

5. CONCLUSIONS

The paper presented the experimental assessment of a DRL-based approach for robot obstacle avoidance. The hybrid learning approach in [Sangiovanni et al., 2021] has been recalled and described also from a LPV perspective. Therefore, a realistic industrial scenario has been repro-

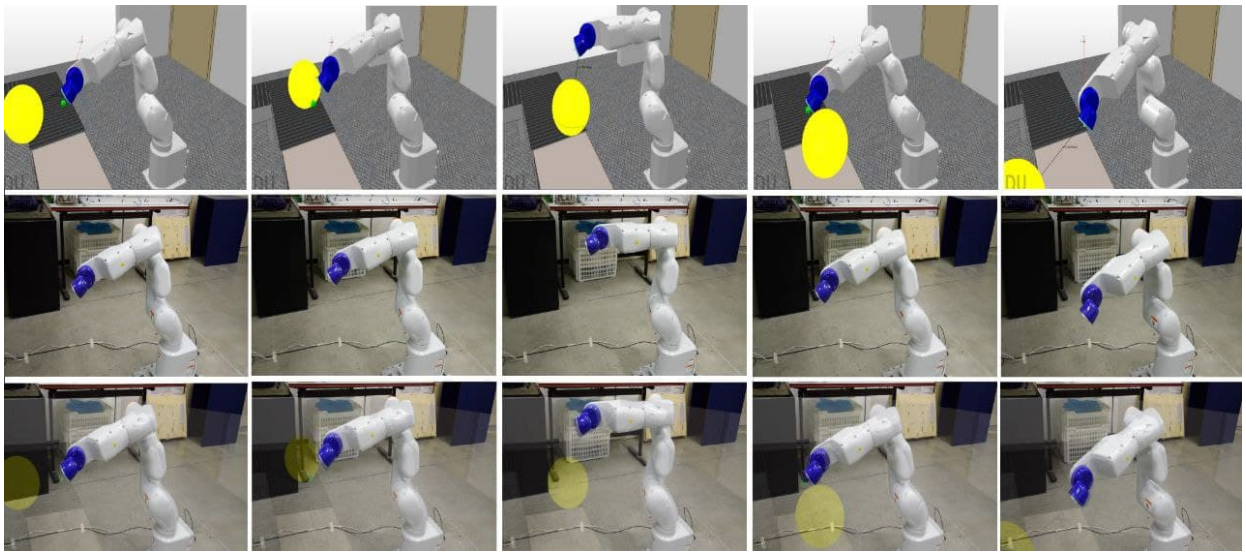


Fig. 7. Virtual environment compared with the real one. From the top: virtual scenario containing the simulated robot and the obstacle (yellow); real scenario containing the robot moving as in the simulator; overlap between the simulated robot and the physical one to show the effectiveness of the strategy with a virtual obstacle

duced in practice relying on the physical robot EPSON VT6, obtaining satisfactory results that make the proposal promising also for industrial field implementations.

REFERENCES

Bicchi, A., Peshkin, M.A., and Colgate, J.E. (2008). *Safety for Physical Human–Robot Interaction*, 1335–1348. Springer Berlin Heidelberg, Berlin, Heidelberg.

Brüggemann, S. and Possieri, C. (2021). On the use of difference of log-sum-exp neural networks to solve data-driven model predictive control tracking problems. *IEEE Control Systems Letters*, 5(4), 1267–1272.

De Luca, A. and Flacco, F. (2012). Integrated control for pHRI: Collision avoidance, detection, reaction and collaboration. In *Proc. 4th IEEE RAS EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, 288–295. Rome, Italy.

De Santis, A., Siciliano, B., De Luca, A., and Bicchi, A. (2008). An atlas of physical human–robot interaction. *Mechanism and Machine Theory*, 43(3), 253–270.

Deisenroth, M.P., Rasmussen, C.E., and Fox, D. (2011). Learning to control a low-cost manipulator using data-efficient reinforcement learning. In *Proc. of the International Conference on Robotics: Science and Systems (RSS)*. Los Angeles, USA.

Ferrara, A., Incremona, G.P., and Sangiovanni, B. (2019). Tracking control via switched integral sliding mode with application to robot manipulators. *Control Engineering Practice*, 90, 257–266.

Gu, S., Lillicrap, T.P., Sutskever, I., and Levine, S. (2016). Continuous deep Q-learning with model-based acceleration. In *Proc. 33rd International Conference on Machine Learning*. New York, NY, USA.

Haarnoja, T., Pong, V., Zhou, A., Dalal, M., Abbeel, P., and Levine, S. (2018). Composable deep reinforcement learning for robotic manipulation. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 6244–6251. St. Paul, MN, USA.

Halalchi, H., Bara, G.I., and Laroche, E. (2010). LPV controller design for robot manipulators based on augmented LMI conditions with structural constraints. *IFAC Proceedings Volumes*, 43(21), 289–295.

Kober, J., Bagnell, J.A., and Peters, J. (2014). Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11), 1238–1274.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521, 436–444.

Lenz, I., Lee, H., and Saxena, A. (2015). Deep learning for detecting robotic grasps. *The International Journal of Robotics Research*, 34(4-5), 705–724.

Levine, S., Pastor, P., Krizhevsky, A., Ibarz, J., and Quillen, D. (2017). Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, 37(4–5), 421–436.

Possieri, C., Incremona, G.P., Calafiore, G.C., and Ferrara, A. (2021). An iterative data-driven linear quadratic method to solve nonlinear discrete-time tracking problems. *IEEE Transactions on Automatic Control*. doi: 10.1109/TAC.2021.3056398, in press.

Sangiovanni, B., Incremona, G.P., Ferrara, A., and Piastra, M. (2018a). Deep reinforcement learning based self-configuring integral sliding mode control scheme for robot manipulators. In *Proc. IEEE Conference on Decision and Control (CDC)*, 5969–5974. Miami Beach, FL, USA.

Sangiovanni, B., Incremona, G.P., Piastra, M., and Ferrara, A. (2021). Self-configuring robot path planning with obstacle avoidance via deep reinforcement learning. *IEEE Control Systems Letters*, 5(2), 397–402.

Sangiovanni, B., Rendiniello, A., Incremona, G.P., Ferrara, A., and Piastra, M. (2018b). Deep reinforcement learning for collision avoidance of robotic manipulators. In *Proc. European Control Conference (ECC)*, 2063–2068. Limassol, Cyprus.

Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G. (2009). *Robotics-Modelling, Planning and Control*. Springer-Verlag, London, UK, third edition.