IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received 5 March 2023, accepted 24 March 2023, date of publication 27 March 2023, date of current version 31 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3262450

RESEARCH ARTICLE

Affordance-Based Human–Robot Interaction With Reinforcement Learning

FRANCISCO MUNGUIA-GALEANO[1], SATHEESHKUMAR VEERAMANI[2], (Member, IEEE), JUAN DAVID HERNÁNDEZ[3], (Senior Member, IEEE), QINGMENG WEN[1], AND ZE JI[1], (Member, IEEE)

[1]School of Engineering, Cardiff University, CF24 3AA Cardiff, U.K.
[2]Cooper Group, University of Liverpool, L69 7ZD Liverpool, U.K.
[3]School of Computer Science and Informatics, Cardiff University, CF24 4AG Cardiff, U.K.

Corresponding author: Francisco Munguia-Galeano (MunguiaGaleanoF@cardiff.ac.uk)

This work was supported in part by the Consejo Nacional de Ciencia y Tecnología (CONACyT).

ABSTRACT Planning precise manipulation in robotics to perform grasp and release-related operations, while interacting with humans is a challenging problem. Reinforcement learning (RL) has the potential to make robots attain this capability. In this paper, we propose an affordance-based human-robot interaction (HRI) framework, aiming to reduce the action space size that would considerably impede the exploration efficiency of the agent. The framework is based on a new algorithm called Contextual Q-learning (CQL). We first show that the proposed algorithm trains in a reduced amount of time (2.7 seconds) and reaches an 84% of success rate. This suits the robot's learning efficiency to observe the current scenario configuration and learn to solve it. Then, we empirically validate the framework for implementation in HRI real-world scenarios. During the HRI, the robot uses semantic information from the state and the optimal policy of the last training step to search for relevant changes in the environment that may trigger the generation of a new policy.

INDEX TERMS Q-learning, robotics, affordances, robot learning, human–robot interaction.

I. INTRODUCTION

Robots with human-level intelligence to plan and adapt to dynamic environments may open the door to full integration of robotics in industrial and domestic environments [1]. In the context of human-robot interaction (HRI), reinforcement learning (RL) has been applied to adapt the behavior of the robot to the user in dynamic environments, such as finding optimal parameters in a robot arm impedance model [2], biped dynamic walking [3] and allowing navigation among crowds [4], [5]. HRI is challenging and requires dealing with dynamic changes in the environment. A robot with the capacity to reprogram itself when necessary may lead to improve HRI [6], [7], [8], [9], [10].This is particularly important in grasp, and release-related tasks, where combining RL capacities to adapt to dynamic environments and HRI is crucial [11].

In the literature, several approaches exist for automating repetitive robot tasks based on HRI, such as learning from demonstration (LfD). Usually, LfD is executed employing three methods: kinesthetic, passive observation, and teleoperation [12]. In kinesthetic demonstrations, the user manually guides the robot by pulling or pushing the end effector [13], [14]. Passive observation is when the robot learns from the user through video streams [15]. With teleoperation, the user guides the robot by operating a teach pendant, a joystick, or a haptic device [16]. A limitation of LfD approaches is that these are constrained to the solution shown by the user, and more optimal solutions are often discarded. Hence, RL offers a solution to this problem by encouraging the agent to explore further and find better solutions than the user's demonstrated one.

A problem with RL is that, because of its stochastic nature during the exploration-exploitation process, the actions taken by the agent rely only on the reward function, which can be sparse for complex tasks. Hence, the agent wastes time

The associate editor coordinating the review of this manuscript and approving it for publication was Zheng H. Zhu.

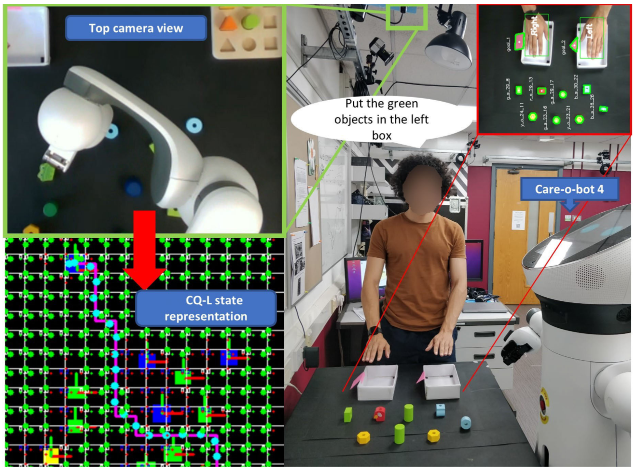

Top camera view
Put the green objects in the left box
CQ-L state representation
Care-o-bot 4
Right
Left

FIGURE 1. Experimental setup. The view of the camera and a visual representation of the state can be appreciated at the bottom left and the upper right of the image respectively.

learning what not to do while increasing the complexity of the design of reward functions [17]. This is because (i) RL relies on data that are the product of the policy through random exploration and the policy is trained through only the indirect information that is given as a reward [18], and (ii) the contextual information, such as affordances given a state, is often ignored, such that the agent must learn it from scratch.

An approach to solve this problem is based on a concept that Gibson [19] coined as affordances, which is contextual information that represents the relationships between actions and objects. Affordances give information about the effect of a given action i.e., whether a particular task affords an action or not [20], [21]. Affordances enhance the performance of the $\epsilon - greedy$ policy [22] (the $\epsilon - greedy$ policy randomly chooses actions during the exploration-exploitation process of the agent). The application of affordances in robotics is important because it encourages human-like generalization capabilities [23].

In this paper, we present an affordance-based human-robot interaction framework, whose novelty lies in its capacity to use contextual information (semantic information, affordances, and high-level goals) that enhances the exploration and learning process of the agent. The framework is based on a new algorithm called Contextual Q-learning (CQL). The algorithm aims to reduce the exploration space of the agent and the number of states required to represent it. This allows CQL to find a policy from the current observation in the real world and solve the Q-table in a short period of time. This fast learning capacity makes the framework suitable for HRI tasks. Our contributions are (i) CQL is introduced, allowing efficient learning in the context of active HRI, and (ii) a framework based on CQL that allows robots to perform HRI in the real world.

The problem domain to validate the framework empirically is an HRI scenario (Fig. 1). Here, the user first provides instructions and then actively interacts with the robot to manipulate objects. At the same time, changes that interfere intermittently with the robot's actions are produced. CQL is validated experimentally by comparing its performance against baseline algorithms such as classical Q-learning (QL) [24], deep Q-network (DQN) [25], proximal policy optimization (PPO) [26] and advantage actor-critic (A2C) [27] from the stable-baselines [28] implementations.

The rest of this article is structured as follows. We begin with Section II that introduces relevant concepts in RL. Section III summarises related works in RL and HRI. Then in Section IV, we formally introduce the framework based on CQL. Followed by Section V that describes the experimental setup. Section VI discusses the implementability of the framework in an HRI scenario and the results. Finally, we conclude this paper in Section VII.

II. PRELIMINARIES

A Markov decision process (MDP) is a 5-tuple $\langle S, A, R, T, \gamma \rangle$ where S is a set of states, A is a set of actions, $R(a, s)$ is a reward function, $T(s'|s, a)$ is a transition function equal to a probability distribution $P(s'|s, a)$, and γ is a discount factor [29]. To solve an MDP, RL agents are designed to learn a policy that maximizes the reward while mapping which action a is the best given a state s where $a \in A$ and $s \in S$. Usually, RL agents perform the following steps: observing the environment, selecting an action a, receiving a reward $R(a, s)$, transitioning into a new state s' based on $T(s'|s, a)$, and updating the policy. Moreover, RL agents utilize a value function to estimate the quality of the action-state pairs.

QL is an off-policy RL algorithm that learns a Markov decision process (MDP) in discrete environments. QL updates the quality of a state-action combination given by:

$$Q : A \times S \longrightarrow \mathbb{R}, \tag{1}$$

where Q are the Q-values. The Q-values represent the quality of the state-action pair. In other words, the higher the Q-value, the better the action for that state. Then, a Q-table is necessary to represent each Q-value such that QL updates the state-action pair by using the following:

$$Q(s, a) \longleftarrow Q(s, a) + \alpha[r + \gamma(\max_{a'} Q^{*}(s', a') - Q(s, a))], \tag{2}$$

where α is the learning rate, r is the reward and γ is the discount factor. The discount factor is usually a value between 0 and 1 ($0 \leq \gamma \leq 1$) that balances the importance the agent puts on future rewards rather than immediate rewards. According to equation (2), the state-action pair is updated based on the next state s' even when that state has not been explored, which is why QL is considered an off-policy method.

III. RELATED WORK

Based on RL approaches, many researchers have worked towards improving the autonomy of robotic manipulation for several applications, including but not limited to motion planning [30], [31], obstacle avoidance [32], welding [33], robot manipulation [34], robot-assisted rehabilitation [35], and dual-arm motion planning [36]. In these works, classical planning methods, including rapidly exploring random trees (RRT) [37] and variants of it like RRT* [38], are combined with RL approaches such as actor-critic architectures [27], and deep deterministic policy gradient (DDPG) [39]. This combination enhances the performance and quality of the robot's navigation and motion planning capabilities. However, the long training time, the necessity of simulated environments, and the challenging task of transferring the knowledge from simulation to the real world difficult the generalization of these RL approaches. Besides, the presence of humans and their interaction with them or the potential of using contextual information as part of the agent's training is omitted. Our framework trains from data obtained in the real world and generalizes into different scenarios because of its capacity to learn in a reduced amount of time.

It is essential to recognize that a robot must also be re-programmed when the environment changes [40], and perception is an important factor in achieving it. For example, Lin et al. [41] presented a framework that utilizes the objects' computer-aided design (CAD) models to match with the point cloud produced by a depth camera to find the best grasping pose for an object. Nonetheless, the generalization of the previously mentioned approach would involve having a CAD model of every object the robot needs to grasp. In [42] and [43], convolutional neural networks are used for predicting the grasping rotations and locations of several objects. Nevertheless, large datasets are required to achieve a high success rate. Zeng et al. [44] proposed using genetic algorithms to avoid using large datasets. However, the input is the point cloud and lacks contextual information that may enhance the performance of these approaches. On the other hand, our framework does not require datasets or a point cloud as input because the state is represented in the shape contextual state matrix. With this matrix, our framework can learn from it, extract affordances, and improve its decision-making capabilities.

Among RL methods, QL is a promising approach that makes robots perform manipulation tasks [45]. When hybridized with secondary optimization algorithms, it is also known that the classical QL algorithm achieves better results [46], [47]. In RL, secondary optimization algorithms are used to modify the principal optimization algorithm. Some approaches have employed secondary optimization algorithms in QL agents to set initial optimistic values, and in robotics in several works [48], [49], with promising results. For instance, integrating a novel flower pollination algorithm with classical QL to initialize the Q-table and selection of control parameters accelerate the learning process of the traditional firefly algorithm [50]. This approach establishes a balance between the exploration and exploitation of the computational agent during the search process. Splitting the task into a hierarchy of small parts is proved to be an effective method in QL [51]. Ji et al. [52] alluded to a novel QL-based approach that efficiently computes the path of the robot arm based on a hybrid path planning method, which splits the planning problem into two separate parts: active finding (finds simple actions for the robot arm) and passive finding (computes joint angles). Nevertheless, in all the approaches mentioned, contextual information is not used during the training of the agents.

A proposed solution methodology to address the problem is using affordances (two examples of affordances are: a pen affords writing and a keyboard affords typing). When implemented in a Markov decision process (MDP), the affordances make the agent choose optimal actions sooner, dramatically reducing the number of state-action pairs the robot needs to evaluate. Affordances improve the planning, control, recognition, transferability, and programming style of robots [53], [54]. In RL and HRI, the use of affordances has been studied to solve problems, such as cleaning tables [55], reacting to non-verbal user's clues [56], identifying users' behaviors [57], coordinating human and robot actions [58], [59], and learning from the user [60], [61]. Despite all these advantages, using RL based on contextual information during HRI for manipulation tasks is still underdeveloped.

To the best of our knowledge, the potential of combining an optimal policy obtained with CQL, the semantic representation of the current state, and the reward function used as a trigger to scan for changes in the environment to enhance the decision-making capacity of robots has not been explored in HRI setups. The novelty of this work is in leveraging the use of contextual information (semantic information, affordances, and high-level goals) to understand human instructions related to manipulation tasks and actively produce a series of continuous actions in a model-free based approach that allows robots to execute manipulation tasks while interacting with humans in the real world.

IV. AFFORDANCE-BASED HUMAN-ROBOT INTERACTION FRAMEWORK

In this section, we present a framework that aims to solve the problem of performing grasp and release-related operations during HRI (Fig. 2). The framework is conformed of three modules: voice-gestures, learning, and valid policy detector.

A. VOICE-GESTURES

With this module, to extract the sub-goals set Π, the robot acquires human instructions by using the Google speech-to-text API [62] and the CVZone package [63] for hand-tracking. When the instruction ι is ready, the algorithm 1 processes the string and fills Π. When the word "here" is in the instruction, the module tracks the hand position and finds the closest goal, which is stored in the set of goals G, given by:

$$G^4 = \{\langle g_i, x_i, y_i, z_i \rangle \mid x_i, y_i, z_i \in \mathbb{R}\}, \tag{3}$$

where g_i is the name of the goal. The terms x_i, y_i, and z_i are the coordinates of the *ith* goal in the x, y, z axis respectively.

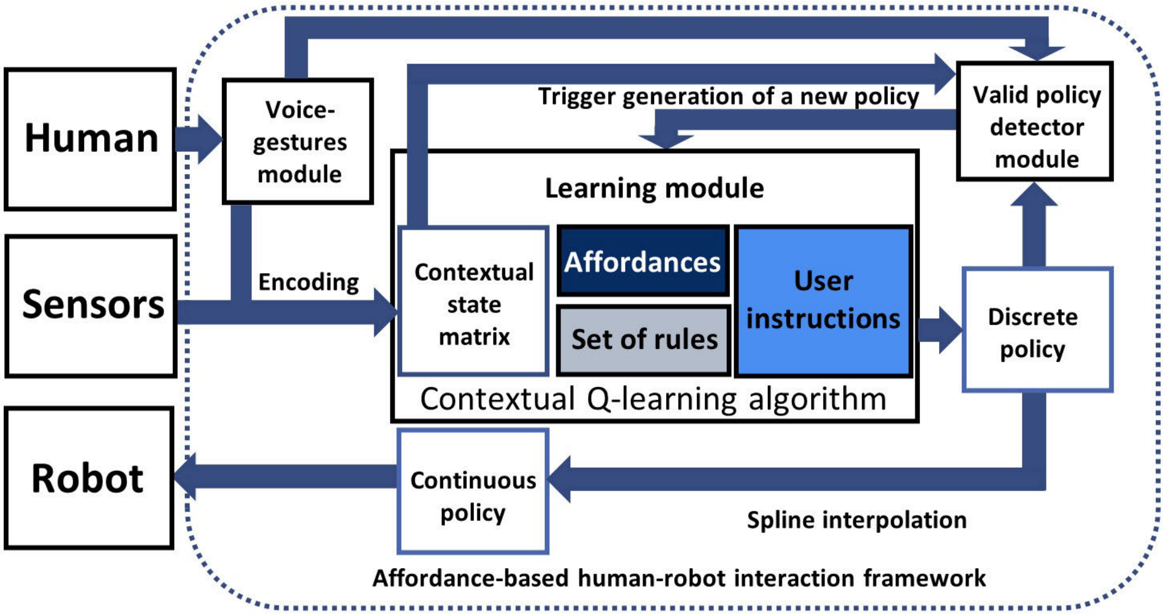


FIGURE 2. Proposed framework for HRI using RL.

Algorithm 1 Sub-Goals Extractor

```
Input : Human instruction $\iota$, Semantic set $O$, the phrase set $\rho$= { }
Result : Sub-tasks set $\Pi$
while$|\iota| > 0$
    for $word$ in $\iota$ do
        $\rho = \rho \cup \{word\}$
        if $|\rho \cap G^1| > 0$ and $|\rho \cap V| > 0$
            $goal \in \rho \cap G^1$
            if $|\rho \cap \{all, every, the\}| > 0$ and $|\rho \cap \mathbb{Z}| = 0$
                $\Pi = \Pi \cap \{\langle obj, goal\rangle \,|\, obj \in O^1 \cap \rho\}$
            if $|\rho \cap \{some, the\}| > 0$ and $|\rho \cap \mathbb{Z}| = 0$
                $\Pi = \Pi \cap \{\langle obj, goal\rangle \,|\, obj \in H\}$
            if $|\rho \cap \mathbb{Z}| > 0$
                Set $n$ to $|\rho \cap \mathbb{Z}|$;
                Fill set $H$ by randomly pick $n$ objects from $O^1 \cap \rho$;
                $\Pi = \Pi \cap \{\langle obj, goal\rangle \,|\, obj \in H\}$
            if $|\rho \cap \{a, an\}| > 0$
                $\Pi = \Pi \cap \{\langle obj, goal\rangle \,|\, obj \in_R O^1 \cap \rho\}$
        Remove $\rho$ elements from $\iota$ and set $\rho = \{\}$
    end for
if $|\Pi| = 0$
    Send error: "Not executable instruction."
end
```

This allows replacing "here" with the name of the goal. Consequently, algorithm 1 can be applied to relate the goals with the objects and split the task into sub-goals. For each sub-goal, CQL solves its respective contextual Q-table. When all the sub-goals are completed or Π is empty, the task is considered finished.

B. LEARNING

The learning module uses the proposed algorithm CQL to learn the set of actions that solve a task established by the user through the voice-gesture module. CQL uses a set of codes that numerically represents the state s of the environment. This set is used to create contextual Q-tables and to extract the affordances. Formally, a state s is a $n \times m$ matrix that contains semantic information of the state and $s_{n,m} \in \{0, 0.1, 0.2, 0.3, 0.4\}$, where 0 is an empty space, 0.1 is used to point an object as the target to manipulate, 0.2 stands for objects that become obstacles in the environment, 0.3 represents the goal, and 0.4 represents a hand. For every new object type, a number will be added to tokenize it in s. The value of the tokens is in the range between zero and one to avoid state-of-the-art approaches to learning spurious correlations due to high values. This normalization also assists with faster convergence by avoiding large input values that may mislead the agent. The objects' semantic information is represented in the object's set O, given by:

$$O^8 = \{\langle k_i, x_i, y_i, w_i, l_i, h_i, name_i\rangle \,|\, i \in (0, 1, .., n),\\ x_i, y_i, w_i, l_i, h_i \in [0, \infty)\}, \tag{4}$$

where O represents the semantic set, $i \in (0, 1, \ldots, n)$, k_i is the ith of the element, x and y are the position coordinates in pixels, w is the width, l is the length, h is the height, c is the colour and s is the shape of the ith element respectively and $name_i$ is the name of the object (e.g., "blue-square-1", "yellow-hexagon-2").

Algorithm 2 uses the semantic set O to create a matrix that represents the state s. The dimension of the states matrix s is given by $n \times m$.

Algorithm 2 s Generator

```
Input : Semantic set $O$, the number of elements $n_e$, the screen width $sw$, and the screen height $sh$
Result :$s$
$n \leftarrow \lceil \frac{sh}{h_1} \rceil$;
$m \leftarrow \lceil \frac{sw}{w_1} \rceil$;
Create an $n \times m$ matrix $s$ filled with $zeros$;
for $i = 1,\ n_e$ do
    for $r = 0,\ w_i$ do
        for $c = 0,\ h_i$ do
            if $k_i$ is the target do
                $s_{(u_i+r, v_i+c)} = 0.1$;
            if $k_i$ is an obstacle do
                $s_{(u_i+r, v_i+c)} = 0.2$;
            if $k_i$ is the goal do
                $s_{(u_i+r, v_i+c)} = 0.3$;
            if $k_i$ is a user's hand do
                $s_{(u_i+r, v_i+c)} = 0.4$;
        end for
    end for
end for
```

Algorithm 3 Contextual Q-Learning

```
Input : The goal $g$, and start position $p$
With equation (9), create a contextual Q-table
for $n$ steps do
    if $\forall a, Q(s,a) = 0$
        Randomly select an action $a_t$
    else
        With probability $\epsilon$ select a valid action $a_t$ with equation (10)
    Perform $a_t$ and get reward $r$
    if terminal state
        $Q(s,a) \longleftarrow r$
    else
        Update Q-table with equation (11)
end for
Apply spline interpolation to the series of discrete actions of the optimal policy
```

Actions are defined as follows:

$$\begin{aligned} A = \{&UP, DOWN, LEFT, RIGHT, GRASP, \\ &DROP, GOAL\}, \end{aligned} \tag{5}$$

where *UP* and *DOWN* are the displacements along the y axis given by $\pm l_i$. *RIGHT* and *LEFT* are the displacements along the x-axis given by $\pm w_i$, *GRASP* closes the gripper of the robot and controls its orientation according to l_i and w_i, *DROP* opens the gripper, and *GOAL* is the trajectory from the coordinates of the closest state of the target to the robot to the goal's position. The affordances Λ are given by:

$$\Lambda : A \times S \longrightarrow \mathbb{Z}_2^{|A|}, \tag{6}$$

Context is comprised of the affordances Λ, the semantic set O, the state s and a set of rules R. Let:

$$\begin{aligned} R = \{&\langle UP, 0, 0, 1\rangle, \langle DOWN, 0, 0, -1\rangle, \\ &\langle LEFT, 0, -1, 0\rangle, \langle RIGHT, 0, 1, 0\rangle, \\ &\langle GRASP, 0.1, 0, 0\rangle, \langle DROP, 0.3, 0, 0\rangle, \\ &\langle GOAL, 0.3, \pm 1, \pm 1\rangle\}, \end{aligned} \tag{7}$$

be the set of rules manually defined that contains which interactions among the actions and environment should not be performed by the robot. For example, picking an object when there is nothing to pick or moving to a place where the robot may collide. R^3 is the horizontal exploration range and R^4 is the vertical exploration range. The affordance ζ of each state-action pair is given by:

$$\zeta(s,a) = \begin{cases} 1, & |\{\langle a, s_{p,q}, u, v\rangle\} \cap I| > 1 \text{ and} \\ & |\{\langle a, s_{p+u,q+v}, u, v\rangle \mid a \in E\} \cap R| = 0 \\ 1, & |\{\langle a, s_{p,q}, u, v\rangle\} \cap E| > 1 \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

where $a \in A$, $p \in (0,n)$, $q \in (0,m)$, $u, v \in \{-1, 0, 1\}$, $I = \{UP, DOWN, LEFT, RIGHT\}$, and $E = \{GRASP, DROP, GOAL\}$. With the affordances equation (7) and the current state, it is possible to generate a contextual Q-table and set optimistic initial values, by:

$$Q(s,a) = \zeta(s,a) + \xi(s,a,g), \tag{9}$$

where $\xi(s,a,g) = 0.1$, when the action a points to the goal g, $\zeta > 0$ and $a \in I$, otherwise $\xi(s,a,g) = 0$. The affordances $\xi(s,a,g)$ bias the Q-values of the actions that point to the goal. Once the contextual Q-table is ready, it is necessary to find an optimal policy. To select the valid action given a state s, it is necessary to include the affordances function (8) by applying the Hadamard product operation:

$$a_t = \arg\max_a [(Q(s,a) + |\min_a Q(s,a)|) \odot \zeta(s,A)], \tag{10}$$

where a_t is the maximum possible action according to $\zeta(s,A)$, and $\forall a, s, Q(s,a) \neq 0$. Therefore, to update the Q-table by including the affordances function (8), the following equation is used:

$$\begin{aligned} Q(s,a) \longleftarrow &Q(s,a) + \alpha[r + \gamma \max_{a'}[(Q(s',a') + \\ &|\min_{a'} Q(s',a')|) \odot \zeta(s',A) - \min_{a'} Q(s',a')]] \end{aligned} \tag{11}$$

Algorithm 3 output is an optimal policy that produces a discrete set of actions D. This set is used to generate a discrete path. Since the robot is a continuous agent, it is necessary to smooth the discrete path by applying spline interpolation.

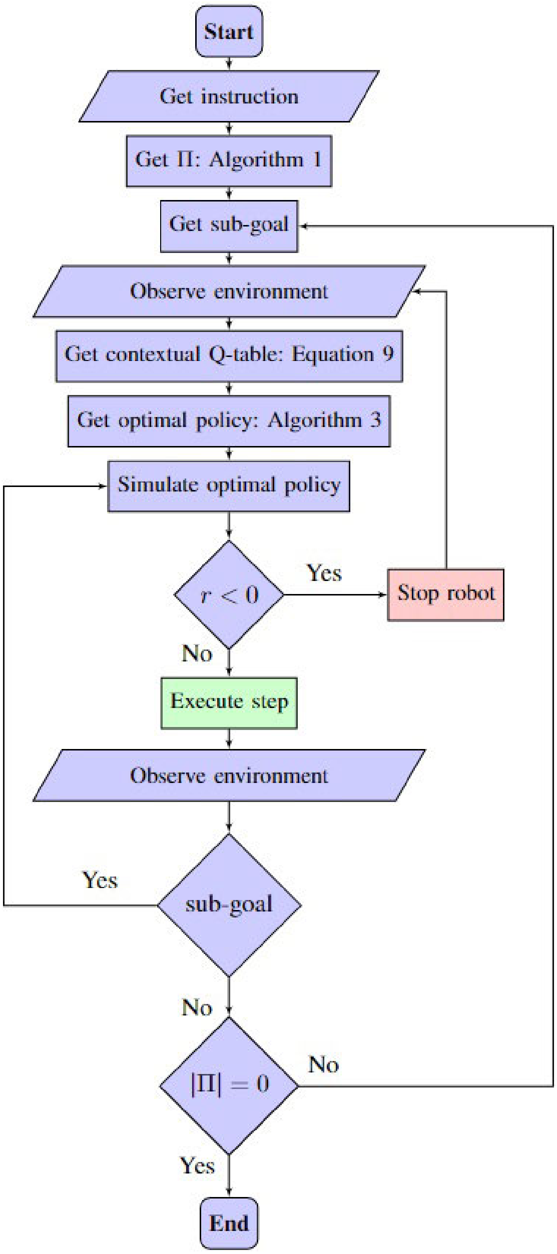


FIGURE 3. Flow chart of the HRI experiments with RL.

C. VALID POLICY DETECTOR

Within each step of execution, changes in the environment are observed to decide if re-planning is required. Since the valid policy detector module is continuously observing the environment, when a change occurs (i.e., interruption of user hand or change in goal position), a negative reward will be returned such that the robot uses it as a trigger and reprograms itself using CQL to complete the manipulation task.

V. EXPERIMENTAL SETUP

The problem domain to validate the framework empirically is an HRI scenario, where the user and robot manipulate objects. At the same time, the user provokes changes that interfere intermittently with the robot's actions. The experiment setup is comprised of a service robot (Care-O-Bot 4®), a table with a set of objects, and an Intel® RealSense camera mounted on the top of the table, as shown in Fig. 1. The learning parameters used for CQL are $\alpha = 0.99$ and $\gamma = 0.05$ (we empirically found by trial and error that CQL converges faster when using these parameters). We performed the following experiments:

1) The performance of CQL is compared against the performance of QL, DQN, PPO, and A2C by solving the MDP for 100 objects' manipulation tasks with the robot. Every manipulation task is contained in a scenario configuration, each with a different initial position, goal destination, and different obstacles.
2) The user asks the robot for a certain task and puts pieces on the table while the robot executes the task.
3) A certain type of object is set as a sensible obstacle to test the robot's capacity to establish a safety perimeter around the obstacle.
4) The user moves the goal to test the robot's capacity to recognize a change in the environment that may lead to failing the current task.
5) The user obstructs the way of the robot with a hand to test the robot's capacity to react safely to the user's movements.
6) The framework is tested in a KUKA LBR IIWA 14 ® robot.

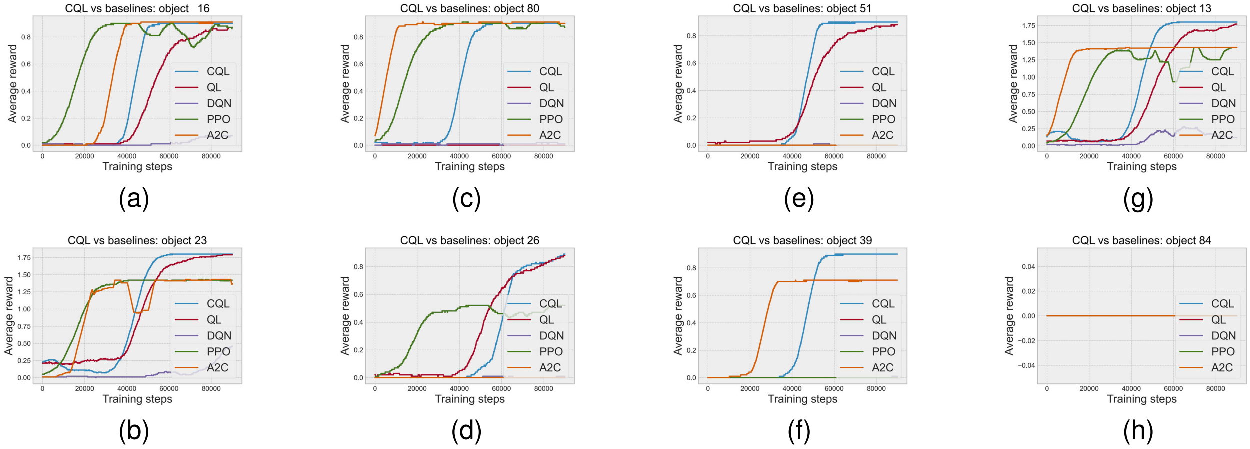
For each experiment, in turn, we aim to answer the following questions:

1) Does CQL perform better than QL, DQN, PPO, and A2C to suit the learning efficiency needed for HRI?
2) Can the robot understand the user's instructions and learn how to manipulate the objects on the table to complete the task using CQL?
3) Does CQL generate a series of optimal continuous actions that not only take an object from one place to another but also avoid collisions?
4) Can the simulation of the last optimal policy produce a negative reward trigger in real-time to re-plan the actions of the robot?
5) Can the robot safely react in real time to a dynamic obstruction from a human?
6) Is the framework suitable for a different robot configuration?

For all the algorithms, we count the number of times the agent finds a solution for the 100 hundred manipulation tasks such that it is possible to calculate the proportion of successful attempts, which we refer to as the success rate. For a better understanding of the experimental setup, see: https://youtu.be/raVeVjPv_Rc

VI. RESULTS

In the first experiment, we used CQL, QL, DQN, PPO, and A2C for training over 100 different scenarios configurations. With the current scenario configuration, in turn, a new



(a) (c) (e) (g)

(b) (d) (f) (h)

FIGURE 4. The learning curves above are eight out of 100 samples taken from the results of the experiments. All algorithms succeeded in (a) and (b), but DQN failed. In (c), QL and DQN fail to find a solution. In (d), A2C and DQN fail to find a solution while PPO and QL struggle to converge. In (e), only CQL and QL converge. In (f), CQL and A2C converge while the rest of the algorithms fail. In (g), PPO struggles to converge while CQL finds a solution. In (h), all the algorithms fail to find a solution.

contextual Q-table is generated. This table contains all possible states given the current scenario configuration. A problem with deep RL approaches is that these over-fit in simulation or the real world for a certain task, and it is difficult to make them work under different tasks or environments [64]. Hence, a change in the state related to perception or its configuration would affect the agent's performance. To avoid this issue, CQL design aims to generalize by learning from scratch given the current scenario configuration (Fig. 3). Despite changes in the environment, the algorithm always generates a new contextual Q-table for that scenario configuration. Once the contextual Q-table is solved, the set of actions can be safely transferred to the robot. In this context, we compared the performance of CQL, QL, DQN, PPO, and A2C. The CPU energy consumption of each algorithm is measured with the PyRAPL library [65].

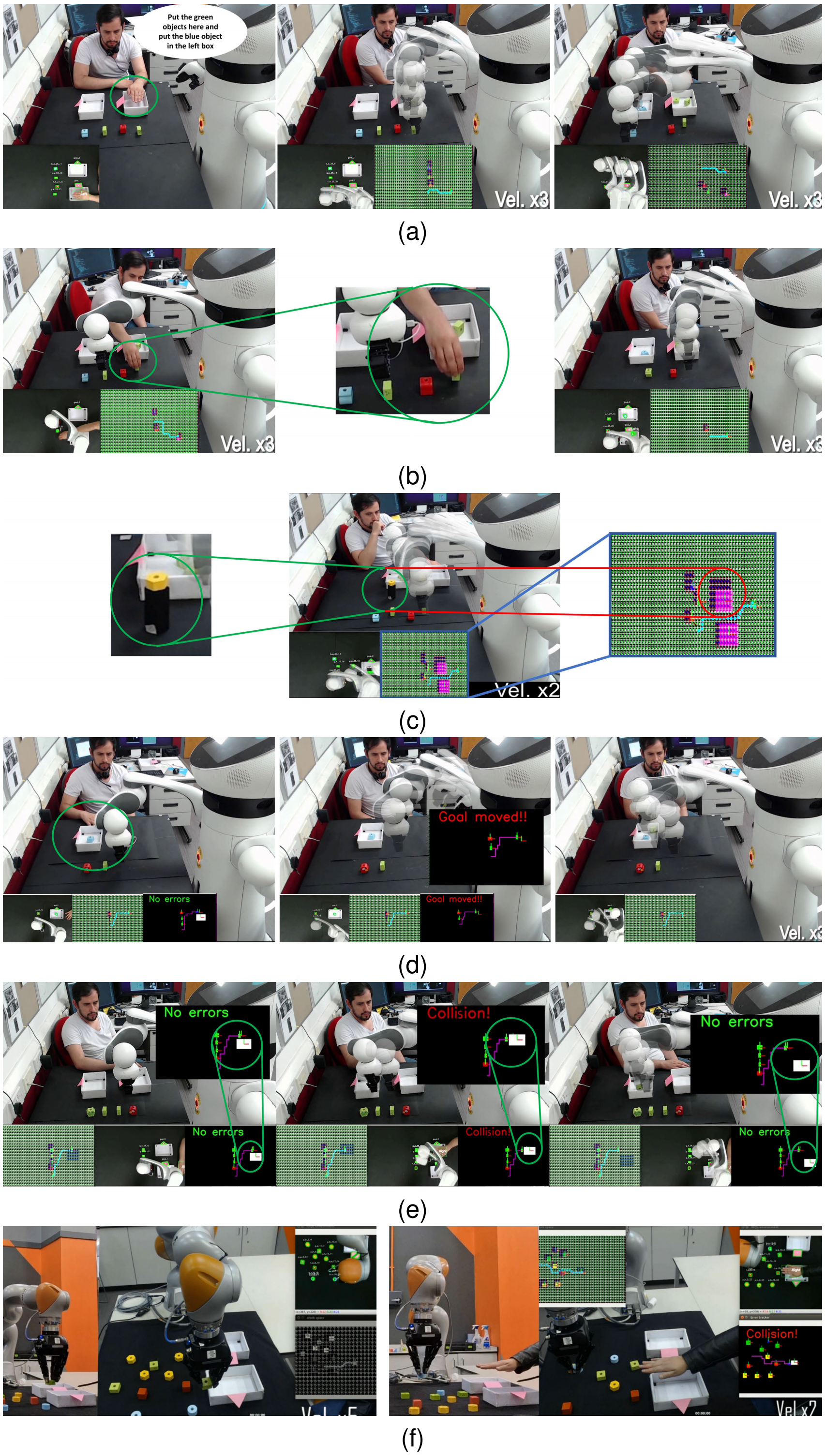
In Fig. 4, we show the learning curves of CQL, QL, DQN, PPO, and A2C after training over 100 different scenarios configurations. The results are summarised in Table 1. Where, CQL had a success rate of 84%, took 67,631 training steps on average to converge in 2.7 seconds, and expended 61.76 J of energy. QL took 1.02 seconds on average to converge and consumed 26.9 J. However, the QL success rate is 38% and takes 67,631 steps on average to converge. DQN had the slowest learning rate and it only succeeded in 17% of the scenarios with 39.9 seconds of learning time, an average number of steps to converge of 76,430, and energy consumption of 886.83 J. Among the stable-baselines algorithms, PPO showed to be the best by succeeding in 68% of the cases in 50,098 average number of steps to converge. Nevertheless, PPO learning time is about 105 seconds and consumes 2662.49 J. In terms of less number of training steps to converge (Fig. 4g), A2C spends 30,915 but its success rate is only 38%. There were cases where any of the algorithms found a solution, as shown in Fig. 4h. This experiment serves as evidence to show that CQL performs better than QL, DQN, PPO, and A2C to suit the learning efficiency needed for HRI.

TABLE 1. Results after running CQL, QL, DQN, PPO, and A2C in 100 different scenarios.

Algorithm	Success rate	Training time	Average number of steps to converge	Energy consumption
CQL	**85%**	2.7 s	60,103	61.76 J
QL	38%	**1.02** s	67,631	**26.9 J**
DQN	17%	36.9 s	76,430	886.83 J
PPO	68%	91.8 s	50,098	2122.52 J
A2C	39%	104.6 s	**30,915**	2662.49 J

The user provided visual and spoken instructions in the second experiment (Fig. 5a). The tasks were divided into sub-goals and generated a contextual Q-table for each object that met the description of the instruction. For each contextual Q-table, CQL was applied to solve the MDP by obtaining a discrete optimal policy. Once the discrete policy was available, CQL applied spline interpolation to transform the discrete series of actions into a continuous one and send it to the Care-O-Bot 4® 's controller. Between every continuous action, the robot looked for relevant changes in the environment and could identify the extra pieces the user put on the table (Fig. 5b). Therefore, the robot puts the objects into its corresponding box. This demonstration indicates that the robot understands the user's instructions and learns how to manipulate the objects on the table to complete the task using CQL.

In the third experiment (Fig. 5c), we set the yellow objects as sensible obstacles such that CQL established a virtual security perimeter. The robot executed the series of continuous actions without colliding with the sensible obstacles. This demonstrated that CQL generates a series of optimal continuous actions that take objects from one place to another while avoiding collisions.



(a)

(b)

(c)

(d)

(e)

(f)

FIGURE 5. This figure illustrates Care-O-Bot 4 ® and KUKA LBR IIWA 14 ® robots running the proposed framework. The experiments are conducted in the same HRI scenario. Each subfigure corresponds to a different experiment, wherein a user and robot manipulate objects while the first provokes intermittent changes that interfere with the robot's task. In (a), the user speaks an instruction while pointing to the right box with his hand, and the robot performs the instruction. In (b), the user puts extra objects on the table such that the robot identifies them and puts them into the box. In (c), the yellow objects are identified as sensible obstacles, and CQL adds a security perimeter while the robot successfully avoids those obstacles. In (d), the user moves the goal, and a negative reward trigger is returned such that the robot identifies that the user moved the goal and then re-planned its movements. In (e), the user puts a hand in the way of the robot's path. Consequently, a negative reward is returned, and the robot asks the user to move his hand. In (f), the framework is tested in a different robot configuration.

In the fourth experiment (Fig. 5d), the user moved the goal while the robot was holding an object. Consequently, the robot identified a relevant change in the environment by running the policy in the most recently observed environment and obtained a negative reward. In this case, the simulation of the policy dropped the object in a 0.0 (empty space) square instead of a 0.3 (goal) one. Hence, the robot could identify the type of error and react accordingly. The robot finished the task by computing a new policy with CQL and dropping the object in the new goal. This demonstration shows that the simulation of the last optimal policy can produce a negative reward trigger in real-time to re-plan the robot's actions.

In the fifth experiment (Fig. 5e), the user placed his hand in the way of the robot. The robot identified this relevant change in the environment. In this experiment, the robot found a collision while simulating the set of discrete actions against a 0.4 (user's hand). The robot asked the user to be careful with his hand, and after the user moved his hand, the robot finished the task. Overall, the robot successfully reacts safely and in real-time to dynamic obstructions from the user.

In the sixth experiment (Fig. 5f), we tested the framework in a robot with a different configuration. Even though RL performs well in simulation, its implementation on real robots experiences several shortcomings, such as incomplete perception information and physical differences between the simulation and real-world scenarios. This may difficult the implementation of RL algorithms into different robot setups. However, using our framework, the KUKA robot was able to carry out the same activities while showing the same capacities as Care-O-bot 4® during the experiments. To this end, the framework has been shown to be compatible with different robot setups.

VII. CONCLUSION

In this paper, we empirically validated our affordance-based human-robot interaction framework. The experiments showed that the framework allows robots to understand instructions and execute manipulation during HRI based on an RL approach. This shows the impact of adding contextual information, such as semantics and affordances, to set initial optimistic values in a contextual Q-table. The 2.7 seconds of learning time allows CQL to generalize in different setups. Besides, it is possible to observe the environment while the robot executes a task within every step of execution. The robot using our framework showed to be reliable while identifying relevant changes in the environment based on negative rewards and the semantic representation of the state.

We believe that our framework opens the door to robust real-time applications of RL learning in robotics and has the potential to be applied to navigation and collaborative robot problems. The current limitations of this paper include tight shapes (e.g., sticks, pens, or pencils) that may lead CQL to create irregular squares to represent the states such that the agent's exploration could be affected, and the resulting set of actions may not be optimal. In future work, we plan to implement the framework in a more complex HRI scenario, including learning from the user.

REFERENCES

[1] M. A.-M. Khan, M. R. J. Khan, A. Tooshil, N. Sikder, M. A. P. Mahmud, A. Z. Kouzani, and A.-A. Nahid, "A systematic review on reinforcement learning-based robotics within the last decade," *IEEE Access*, vol. 8, pp. 176598–176623, 2020, doi: 10.1109/ACCESS.2020.3027152.

[2] H. Modares, I. Ranatunga, F. L. Lewis, and D. O. Popa, "Optimized assistive human–robot interaction using reinforcement learning," *IEEE Trans. Cybern.*, vol. 46, no. 3, pp. 655–667, Mar. 2016, doi: 10.1109/TCYB.2015.2412554.

[3] J. L. Lin, K.-S. Hwang, W.-C. Jiang, and Y.-J. Chen, "Gait balance and acceleration of a biped robot based on Q-learning," *IEEE Access*, vol. 4, pp. 2439–2449, 2016, doi: 10.1109/ACCESS.2016.2570255.

[4] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 6015–6022, doi: 10.1109/ICRA.2019.8794134.

[5] M. Everett, Y. F. Chen, and J. P. How, "Collision avoidance in pedestrian-rich environments with deep reinforcement learning," *IEEE Access*, vol. 9, pp. 10357–10377, 2021, doi: 10.1109/ACCESS.2021.3050338.

[6] F. Cruz, G. I. Parisi, J. Twiefel, and S. Wermter, "Multi-modal integration of dynamic audiovisual patterns for an interactive reinforcement learning scenario," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Daejeon, South Korea, Oct. 2016, pp. 759–766, doi: 10.1109/IROS.2016.7759137.

[7] P.-H. Ciou, Y.-T. Hsiao, Z.-Z. Wu, S.-H. Tseng, and L.-C. Fu, "Composite reinforcement learning for social robot navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Madrid, Spain, Oct. 2018, pp. 2553–2558, doi: 10.1109/IROS.2018.8593410.

[8] S. Lathuiliere, B. Masse, P. Mesejo, and R. Horaud, "Deep reinforcement learning for audio-visual gaze control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Madrid, Spain, Oct. 2018, pp. 1555–1562, doi: 10.1109/IROS.2018.8594327.

[9] Y. Gao, E. Sibirtseva, G. Castellano, and D. Kragic, "Fast adaptation with meta-reinforcement learning for trust modelling in human–robot interaction," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Macau, China, Nov. 2019, pp. 305–312, doi: 10.1109/IROS40897.2019.8967924.

[10] W. Wang, C. Du, W. Wang, and Z. Du, "A PSO-optimized fuzzy reinforcement learning method for making the minimally invasive surgical arm cleverer," *IEEE Access*, vol. 7, pp. 48655–48670, 2019, doi: 10.1109/ACCESS.2019.2910016.

[11] A. Andriella, C. Torras, and G. Alenyà, "Short-term human–robot interaction adaptability in real-world environments," *Int. J. Social Robot.*, vol. 12, no. 3, pp. 639–657, Jul. 2020.

[12] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annu. Rev. Control, Robot., Auto. Syst.*, vol. 3, pp. 297–330, May 2020.

[13] X. Yu, W. He, Q. Li, Y. Li, and B. Li, "Human–robot co-carrying using visual and force sensing," *IEEE Trans. Ind. Electron.*, vol. 68, no. 9, pp. 8657–8666, Sep. 2021, doi: 10.1109/TIE.2020.3016271.

[14] S. Stavridis, D. Papageorgiou, and Z. Doulgeri, "Kinesthetic teaching of bi-manual tasks with known relative constraints," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Kyoto, Japan, Oct. 2022, pp. 11796–11801, doi: 10.1109/IROS47612.2022.9981196.

[15] P.-J. Hwang, C.-C. Hsu, and W.-Y. Wang, "Development of a mimic robot—Learning from demonstration incorporating object detection and multiaction recognition," *IEEE Consum. Electron. Mag.*, vol. 9, no. 3, pp. 79–87, May 2020, doi: 10.1109/MCE.2019.2956202.

[16] S. Pareek and T. Kesavadas, "IART: Learning from demonstration for assisted robotic therapy using LSTM," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 477–484, Apr. 2020, doi: 10.1109/LRA.2019.2961845.

[17] C. C. Millán-Arias, B. J. T. Fernandes, F. Cruz, R. Dazeley, and S. Fernandes, "A robust approach for continuous interactive actor-critic algorithms," *IEEE Access*, vol. 9, pp. 104242–104260, 2021.

[18] M. Naeem, S. T. H. Rizvi, and A. Coronato, "A gentle introduction to reinforcement learning and its application in different fields," *IEEE Access*, vol. 8, pp. 209320–209344, 2020, doi: 10.1109/ACCESS.2020.3038605.

[19] J. J. Gibson, R. Shaw, and J. Bransford, "Perceiving, acting, and knowing: Toward an ecological psychology," *IEEE Trans. Syst., Man, Cybern.*, vol. 7, no. 1, pp. 67–82, Jan. 1977.

[20] A. Zeng, "Learning visual affordances for robotic manipulation," Ph.D. dissertation, Princeton Univ., Princeton, NJ, USA, 2019.

[21] C. Wang, K. V. Hindriks, and R. Babuska, "Robot learning and use of affordances in goal-directed tasks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 2288–2294, doi: 10.1109/IROS.2013.6696676.

[22] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[23] P. Ardon, E. Pairet, R. P. A. Petrick, S. Ramamoorthy, and K. S. Lohan, "Learning grasp affordance reasoning through semantic relations," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 4571–4578, Oct. 2019, doi: 10.1109/LRA.2019.2933815.

[24] P. Dayan, "Technical note Q-learning," *Mach. Learn.*, vol. 292, pp. 279–292, May 1992.

[25] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing Atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.

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

[27] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. 33rd Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 1928–1937.

[28] A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Y. Sidor, and Wu Stable Baselines. (2018). *GitHub Repository*. [Online]. Available: https://github.com/hill-a/stable-baselines

[29] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*. Cambridge, MA, USA: MIT Press, 1998.

[30] X. Li, H. Liu, and M. Dong, "A general framework of motion planning for redundant robot manipulator based on deep reinforcement learning," *IEEE Trans. Ind. Informat.*, vol. 18, no. 8, pp. 5253–5263, Aug. 2022, doi: 10.1109/TII.2021.3125447.

[31] S. Wen, J. Chen, S. Wang, H. Zhang, and X. Hu, "Path planning of humanoid arm based on deep deterministic policy gradient," in *Proc. IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, Dec. 2018, pp. 1755–1760, doi: 10.1109/ROBIO.2018.8665248.

[32] G. P. Kontoudis and K. G. Vamvoudakis, "Kinodynamic motion planning with continuous-time Q-learning: An online, model-free, and safe navigation framework," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 12, pp. 3803–3817, Dec. 2019.

[33] J. Zhong, T. Wang, and L. Cheng, "Collision-free path planning for welding manipulator via hybrid algorithm of deep reinforcement learning and inverse kinematics," *Complex Intell. Syst.*, vol. 8, pp. 1899–1912, Apr. 2021.

[34] D. Liu, Z. Wang, B. Lu, M. Cong, H. Yu, and Q. Zou, "A reinforcement learning-based framework for robot manipulation skill acquisition," *IEEE Access*, vol. 8, pp. 108429–108437, 2020, doi: 10.1109/ACCESS.2020.3001130.

[35] Y. Zhang, S. Li, K. J. Nolan, and D. Zanotto, "Adaptive assist-as-needed control based on actor-critic reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Macau, China, Nov. 2019, pp. 4066–4071, doi: 10.1109/IROS40897.2019.8968464.

[36] C.-C. Wong, S.-Y. Chien, H.-M. Feng, and H. Aoyama, "Motion planning for dual-arm robot based on soft actor-critic," *IEEE Access*, vol. 9, pp. 26871–26885, 2021, doi: 10.1109/ACCESS.2021.3056903.

[37] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *IEEE Trans. Robot. Autom.*, vol. 10, no. 5, pp. 98–110, Oct. 1998.

[38] S. Karaman and E. Frazzoli, "Incremental sampling-based algorithms for optimal motion planning," *Robot. Sci. Syst.*, vol. 104, pp. 1–8, May 2010.

[39] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *Proc. 4th Int. Conf. Learn. Represent. (ICLR)*, May 2016, pp. 1–14. [Online]. Available: https://iclr.cc/archive/www/doku.php%3Fid=iclr2016:accepted-main.html

[40] K. Zhao, Y. Wang, Y. Zuo, and C. Zhang, "Palletizing robot positioning bolt detection based on improved YOLO-V3," *J. Intell. Robotic Syst.*, vol. 104, no. 3, p. 41, Feb. 2022, doi: 10.1007/s10846-022-01580-w.

[41] H.-Y. Lin, S.-C. Liang, and Y.-K. Chen, "Robotic grasping with multi-view image acquisition and model-based pose estimation," *IEEE Sensors J.*, vol. 21, no. 10, pp. 11870–11878, May 2021, doi: 10.1109/JSEN.2020.3030791.

[42] H. Cheng, Y. Wang, and M. Q.-H. Meng, "A vision-based robot grasping system," *IEEE Sensors J.*, vol. 22, no. 10, pp. 9610–9620, May 2022, doi: 10.1109/JSEN.2022.3163730.

[43] H. Cheng and M. Q.-H. Meng, "A grasp pose detection scheme with an end-to-end CNN regression approach," in *Proc. IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, Dec. 2018, pp. 544–549, doi: 10.1109/ROBIO.2018.8665219.

[44] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, and M. Bauza, "Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 3750–3757, doi: 10.1109/ICRA.2018.8461044.

[45] S. Jain, P. Sharma, J. Bhoiwala, S. Gupta, P. Dutta, K. K. Gotewal, N. Rastogi, and D. Raju, "Deep Q-learning for navigation of robotic arm for Tokamak inspection," in *Proc. Int. Conf. Algorithms Archit. Parallel Process.* Cham, Switzerland: Springer, 2018, pp. 62–71.

[46] A. Konar, I. G. Chakraborty, S. J. Singh, L. C. Jain, and A. K. Nagar, "A deterministic improved Q-learning for path planning of a mobile robot," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 43, no. 5, pp. 1141–1153, Sep. 2013, doi: 10.1109/TSMCA.2012.2227719.

[47] A. Maoudj and A. Hentout, "Optimal path planning approach based on Q-learning algorithm for mobile robots," *Appl. Soft Comput.*, vol. 97, Dec. 2020, Art. no. 106796.

[48] S. Li, X. Xu, and L. Zuo, "Dynamic path planning of a mobile robot with improved Q-learning algorithm," in *Proc. IEEE Int. Conf. Inf. Autom.*, Aug. 2015, pp. 409–414, doi: 10.1109/ICInfA.2015.7279322.

[49] C. Yan and X. Xiang, "A path planning algorithm for UAV based on improved Q-learning," in *Proc. 2nd Int. Conf. Robot. Autom. Sci. (ICRAS)*, Jun. 2018, pp. 1–5, doi: 10.1109/ICRAS.2018.8443226.

[50] E. S. Low, P. Ong, and K. C. Cheah, "Solving the optimal path planning of a mobile robot using improved Q-learning," *Robot. Auto. Syst.*, vol. 115, pp. 143–161, May 2019.

[51] S. Veeramani and S. Muthuswamy, "Hybrid type multi-robot path planning of a serial manipulator and SwarmItFIX robots in sheet metal milling process," *Complex Intell. Syst.*, vol. 8, pp. 2937–2954, Aug. 2022, doi: 10.1007/s40747-021-00499-3.

[52] M. Ji, L. Zhang, and S. Wang, "A path planning approach based on Q-learning for robot arm," in *Proc. 3rd Int. Conf. Robot. Autom. Sci. (ICRAS)*, Jun. 2019, pp. 15–19, doi: 10.1109/ICRAS.2019.8809005.

[53] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser, "Learning synergies between pushing and grasping with self-supervised deep reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 4238–4245.

[54] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "DeepDriving: Learning affordance for direct perception in autonomous driving," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 2722–2730.

[55] F. Cruz, S. Magg, C. Weber, and S. Wermter, "Training agents with interactive reinforcement learning and contextual affordances," *IEEE Trans. Cognit. Develop. Syst.*, vol. 8, no. 4, pp. 271–284, Dec. 2016, doi: 10.1109/TCDS.2016.2543839.

[56] M. Khamassi, G. Velentzas, T. Tsitsimis, and C. Tzafestas, "Robot fast adaptation to changes in human engagement during simulated dynamic social interaction with active exploration in parameterized reinforcement learning," *IEEE Trans. Cogn. Develop. Syst.*, vol. 10, no. 4, pp. 881–893, Jul. 2018, doi: 10.1109/TCDS.2018.2843122.

[57] A. Tabrez and B. Hayes, "Improving human–robot interaction through explainable reinforcement learning," in *Proc. 14th ACM/IEEE Int. Conf. Human-Robot Interact. (HRI)*, Mar. 2019, pp. 751–753, doi: 10.1109/HRI.2019.8673198.

[58] A. Ghadirzadeh, X. Chen, W. Yin, Z. Yi, M. Bjorkman, and D. Kragic, "Human-centered collaborative robots with deep reinforcement learning," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 566–571, Apr. 2021, doi: 10.1109/LRA.2020.3047730.

[59] S. Roy, E. Kieson, C. Abramson, and C. Crick, "Mutual reinforcement learning with robot trainers," in *Proc. 14th ACM/IEEE Int. Conf. Human-Robot Interact. (HRI)*, Mar. 2019, pp. 572–573, doi: 10.1109/HRI.2019.8673284.

[60] R. Dromnelle, B. Girard, E. Renaudo, R. Chatila, and M. Khamassi, "Coping with the variability in humans reward during simulated human–robot interactions through the coordination of multiple learning strategies," in *Proc. 29th IEEE Int. Conf. Robot Human Interact. Commun. (RO-MAN)*, Aug. 2020, pp. 612–617, doi: 10.1109/RO-MAN47096.2020.9223451.

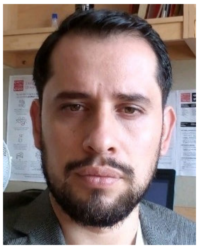
[61] R. Lowe, A. Almer, P. Gander, and C. Balkenius, "Vicarious value learning and inference in human-human and human–robot interaction," in *Proc. 8th Int. Conf. Affect. Comput. Intell. Interact. Workshops Demos (ACIIW)*, Sep. 2019, pp. 395–400, doi: 10.1109/ACIIW.2019.8925235.

[62] Google. (2022). *Google Cloud Speech-to-Text API*. [Online]. Available: https://cloud.google.com/speech-to-text

[63] CVZone, GitHub. *CVZone Library*. Accessed: Mar. 2, 2023. [Online]. Available: https://github.com/cvzone/cvzone

[64] H. Nguyen and H. La, "Review of deep reinforcement learning for robot manipulation," in *Proc. 3rd IEEE Int. Conf. Robotic Comput. (IRC)*, Naples, Italy, Feb. 2019, pp. 590–595, doi: 10.1109/IRC.2019.00120.

[65] B. Rountree and D. Tsafrir, pyRAPL: A Python Interface for RAPL. GitHub. Accessed: Mar. 2, 2023. [Online]. Available: https://github.com/berkeley-abc/pyRAPL



FRANCISCO MUNGUIA-GALEANO received the B.S. degree in robotics engineering and the master's degree (Hons.) in computing technology from Instituto Politécnico Nacional, Mexico, in 2015 and 2018, respectively. He is currently pursuing the Ph.D. degree in engineering with Cardiff University, U.K.

Before joining Cardiff University as a Research Assistant in 2021, he was an Automation Engineer with the Metal Mechanics Sector and, more recently, as a software developer. His research interests include reinforcement learning and robotics.

SATHEESHKUMAR VEERAMANI (Member, IEEE) received the M.Tech. degree in robotics and the Ph.D. degree in multirobot coordination from the Indian Institute of Information Technology (IIITDM), India, in 2017 and 2021, respectively.

He is currently a Postdoctoral Research Associate with the Cooper Group, University of Liverpool, U.K. Previously, he was a Research Associate with the Center for AI, Robotics and Human–Machine Systems, Cardiff University, U.K. During his Ph.D. studies, his research was jointly sponsored by the Ministry of Education, Government of India, under HTRA Fellowship, and the University of Genova, Italy. His works were published in reputed conferences and top-tier journals. He has hands-on working experience with several humanoid, mobile, and industrial robots. His research interest includes reinforcement learning approaches applied to robotics.

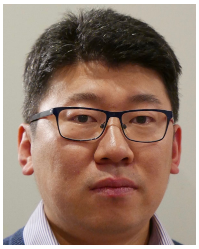
JUAN DAVID HERNÁNDEZ (Senior Member, IEEE) received the B.Sc. degree in electronic engineering from Pontifical Xavierian University, Bogotá, Colombia, in 2009, the M.Sc. degree in robotics and automation from the Technical University of Madrid, Madrid, Spain, in 2012, and the Ph.D. degree in technology (robotics) from the University of Girona, Girona, Spain, in 2017.

He was a Robotics Research Engineer with the Netherlands Organization for Applied Scientific Research (TNO), The Hague, The Netherlands, from 2017 to 2018. He was a Postdoctoral Research Associate with Rice University, Houston, TX, USA, from 2018 to 2019. He was a Senior Engineer for the simulation of autonomous systems with Apple Inc., Sunnyvale, CA, USA, from 2019 to 2020. He is currently a Lecturer (an Assistant Professor) with the School of Computer Science and Informatics, Cardiff University, Cardiff, U.K. His research interests include motion planning algorithms and human–robot collaborations.

Dr. Hernández is a Senior Member of the IEEE Robotics and Automation Society.

QINGMENG WEN received the B.S. degree in automation engineering from Huazhong Agricultural University, Wuhan, China, in 2021. He is currently pursuing the Ph.D. degree with Cardiff University, U.K.

He is a Teaching Assistant with Cardiff University. His research interests include robotics and the skeletal modeling of objects.



ZE JI (Member, IEEE) received the B.Eng. degree from Jilin University, Changchun, China, in 2001, the M.Sc. degree from the University of Birmingham, Birmingham, U.K., in 2003, and the Ph.D. degree from Cardiff University, Cardiff, U.K., in 2007.

He is a Senior Lecturer (an Associate Professor) with the School of Engineering, Cardiff University. Before this, he was working in the industry (Dyson and Lenovo) on autonomous robotics. His research interests include cross-disciplinary, including autonomous robot navigation, robot manipulation, robot learning, computer vision, simultaneous localization and mapping (SLAM), acoustic localization, and tactile sensing.