Artificial Intelligence Review (2022) 55:945–990
https://doi.org/10.1007/s10462-021-09997-9

Reinforcement learning in robotic applications: a comprehensive survey

Bharat Singh[1] · **Rajesh Kumar**[1] · **Vinay Pratap Singh**[1]

Accepted: 29 March 2021 / Published online: 20 April 2021

Abstract
In recent trends, artificial intelligence (AI) is used for the creation of complex automated control systems. Still, researchers are trying to make a completely autonomous system that resembles human beings. Researchers working in AI think that there is a strong connection present between the learning pattern of human and AI. They have analyzed that machine learning (ML) algorithms can effectively make self-learning systems. ML algorithms are a sub-field of AI in which reinforcement learning (RL) is the only available methodology that resembles the learning mechanism of the human brain. Therefore, RL must take a key role in the creation of autonomous robotic systems. In recent years, RL has been applied on many platforms of the robotic systems like an air-based, under-water, land-based, etc., and got a lot of success in solving complex tasks. In this paper, a brief overview of the application of reinforcement algorithms in robotic science is presented. This survey offered a comprehensive review based on segments as (1) development of RL (2) types of RL algorithm like; Actor-Critic, DeepRL, multi-agent RL and Human-centered algorithm (3) various applications of RL in robotics based on their usage platforms such as land-based, water-based and air-based, (4) RL algorithms/mechanism used in robotic applications. Finally, an open discussion is provided that potentially raises a range of future research directions in robotics. The objective of this survey is to present a guidance point for future research in a more meaningful direction.

Keywords Reinforcement learning · Robotics · DeepRL · Multi-agent RL · Actor-critic methods · Human–robot interaction · Neuro-evolution

✉ Bharat Singh
bharatsingh1993@yahoo.com

Rajesh Kumar
rkumar.ee@mnit.ac.in

Vinay Pratap Singh
vinay.ee@mnit.ac.in

[1] Department of Electrical Engineering, MNIT, Jaipur, India

1 Introduction

In recent times, robotics has seen a rising trend in various areas such as disaster management, healthcare, logistics warehouse, space, etc. However, robots used for current applications have a limit in both intelligence and self-learning abilities. Thus, they have failed to achieve the same level of accuracy as humans possess. Therefore, to address this limitation, researchers in robotics have integrated artificial intelligence (AI) with robots. An AI-enabled robot can learn and gain new knowledge from interaction with the environment. This helps in the development of a self-learning automated robot. It improves the overall performance of the robot in the completion of tasks.

Reinforcement learning (RL) is a framework that helps in the development of self-learning capability in robots. Basically, RL is a sub-field of machine learning (ML) which is a part of AI. Broadly, machine learning is classified into three parts, namely, (1) supervised learning: it is a learning mechanism that maps the input to output based on training data-set and data is labeled (2) unsupervised learning: it is a learning mechanism where agent finds the hidden patterns in data-set and data is not labeled (3) reinforcement learning: it is a learning mechanism where an agent learns through interaction with the environment. Figure 1a, b shows the framework of machine learning mechanism and agent-environment interaction in standard RL framework respectively (Sutton 1992; Böhmer et al. 2015; Rylatt et al. 1998; Sutton and Barto 2018; Ribeiro 2002). Comparative analysis of machine learning is presented by Wang et al. (2012).

RL framework helps in learning of agents through the interaction with the environment. At the beginning of the learning process, the initial policy opted by an agent will direct the agent to take action in the present state. The agent-environment interaction provides a reward signal and the agent transit into the next state. Here, the reward signal is predesigned by the domain expert. Basically, the reward signal quantifies how good is the action in that state. The policy is updated based on the obtained reward signal. This agent-environment interaction generates a trajectory of the current state, execution of action in that state, receiving of reward signal, a transition of an agent into the next state, and policy update. This whole process is repeated in a cyclic manner until the learning process is completed as shown in Fig. 1b.

In a more general way, the goal of any RL algorithm is to maximize the cumulative reward for finding the optimal policy. In the beginning, an agent follows a random policy

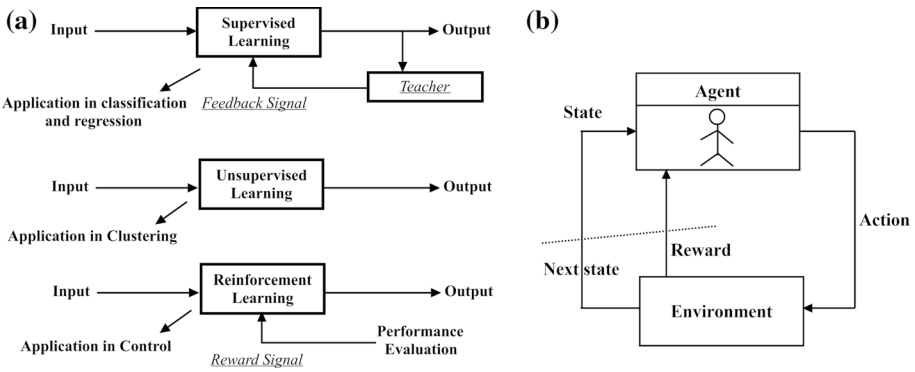

Fig. 1 **a** Framework of machine learning, **b** agent-environment interaction in standard RL framework

or any pre-defined policy. Then, two tasks are performed for finding the optimal policy, namely, policy evaluation and policy improvement. First, policy evaluation means that an agent needs to evaluate the value function for the currently adopted policy. Second, based on the evaluation of value function the agent modifies its policy known as policy improvement. These two tasks are performed in a cyclic manner until the optimal policy is obtained. Thus, the obtained policy is used by agents/robots for the accomplishment of tasks. In the literature (Zhang et al. 2016; Zhou et al. 2016; Bu et al. 2008; Arulkumaran et al. 2017; Sallab et al. 2017; Luo et al. 2018; Nguyen et al. 2017; Li et al. 2019; He et al. 2017; Luo et al. 2017; Littman 2015; Yang and Gu 2004; Luo et al. 2019; Le et al. 2019; Lin et al. 2011), obtained policy by RL shows its effectiveness and capability for multi-agent system, deep learning, human–robot interaction, target-based search, tracking control, output regulation, adaptive learning etc. A survey on policy search methods specifically in robotics is found in Deisenroth et al. (2013b).

A multi-agent system is a group of autonomous agents. They have a common interactive environment. Multi-agent RL (MARL) algorithms are developed for speeding the learning process of multiple agents in a common environment. These algorithms improve the coordination between agents. This makes the overall system robust because if some agents fail to reach the desired goal, then the remaining agents take over the same task and try to solve it. Effectiveness of MARL algorithm has been tested on applications like resource management, automated marketing, robotics, distributed control, robot soccer game, telecommunications, etc (Zhang et al. 2016; Zhou et al. 2016; Bu et al. 2008; Yang and Gu 2004; Duan et al. 2012; Madden and Howley 2004). In literature (Modares et al. 2017), the authors have applied the learning of optimal policy in synchronization with other agents without knowing the dynamics of each other. In some cases when the environment is stochastic, the transition of an agent to the next state is not known. So, the bayesian non-parametric statistic approaches with partially observable RL is used for the determination of transition probability for an agent (Doshi-Velez et al. 2013).

In practice, robotic systems have higher degrees of freedom (DoFs). Thus, the number of states and actions required by RL algorithms see exponential growth. Therefore, RL faces a critical problem of higher dimensionality. This degrades the overall performance of the agent. In literature (Schaul et al. 2015), various function approximators are proposed to tackle the dimensionality problem. Here, we discuss the integration of deep learning with RL. It provides an approximated solution. Literature shows that an approximated solution also provides the same accuracy as other methods (Nguyen et al. 2017). Briefly, the integration of deep learning with RL yields deep reinforcement learning (DeepRL). DeepRL shows its ability in solving the application of robotics such as navigation and path tracking of the robot, object picking task, under-water target-based search, etc., (Arulkumaran et al. 2017; Sallab et al. 2017). Since most DeepRL techniques are not much effective for robotics because they required more interaction time to learn control policies. This problem arises because the state-space representation is needed to learn as a part of the control policy, which is based on observed rewards. However, the reward function quantifies how good the state is, which means it does not tell how to find a good state representation from the sensory observations. So, state-space representation learning can be useful. There are different topologies are provided for integrating the state-space representation into RL algorithms (de Bruin et al. 2018). It reduces the overall dimension which yields the least time-consuming.

In most cases, the neural network architecture in deep learning is trained using back-propagation. Thus, the training of deep learning architecture is tedious. As an alternative approach, which is based on the evolution of the human brain known as “neuro-evolution”

is presented in Stanley et al. (2019). Integration of neuro-evolution with DeepRL leads to further advancement in DeepRL. Its application in robotics can lead to a more advanced intelligent system like humans.

As the usage of robots in the real-world is rising. Robots are interacting morewith the environment and humans. Some authors in literature have exploited this interaction. So, Human-centered learning algorithms are developed. In human-centered algorithms, human feedback is used to improve the learning process. Basically, algorithms evaluate the human feedback that improves the learning efficiency of agents significantly. It leads to increasing its applicability in solving real-time applications. Some survey papers have reported on the human-centered learning process (Li et al. 2019; He et al. 2017; Littman 2015).

Due to an increase in computational power and the growing popularity of RL in various fields, it deserves a focused survey. Although, some survey papers on the RL is reported in literature (Kaelbling et al. 1996; Dayan and Niv 2008; Neftci and Averbeck 2002; Kiumarsi et al. 2017; Bertsekas 2018; Gosavi 2009). However, most of them only focus on the development of RL algorithms. A key survey on "reinforcement learning in robotics" is presented in literature (Kober et al. 2013). This survey establishes a link between robotics and RL. Particularly, it mainly focused on value-function based and policy search methods. A case study on "ball-in cup" with various RL approaches is accomplished. The survey provides an insight that the implementation of RL methods in robotics is not straightforward. Instead, it requires some set of skills like reward function shaping, parameter sensitivity, dealing with high dimensional continuous actions. In the current survey, firstly author provides a basic overview of RL methods such as Actor-Critic, DeepRL, Multi-agent, and Human-centered that are implemented in robotics. Secondly, provide a comprehensive survey on the application of RL in robotics. Thirdly, a discussion on various RL algorithms/mechanisms presented in the literature on various robotic applications is provided. The main focus of this paper is on the wide range of RL applications in robotics that is based on air, land, and water-based usage. It makes this survey unique in nature. The main contributions of this paper are:

1. A basic overview of RL approaches based on Actor-Critic methods, DeepRL, Multi-agent, and Human-centered that applied in robotic science is presented.
2. Detailed survey of robotic applications in air, land, and water-based on the RL is discussed.
3. Various learning mechanisms developed by authors in the literature are presented.
4. Key challenges faced by RL and some open problems are provided.

Paper is organized as: Sect. 2 presents the overview of reinforcement learning, which is the cornerstone of robotic application. Section 3 describes an application of various RL approaches in robotics. Section 4 discussed the survey on some exciting RL mechanisms adopted by authors to find the optimal policy in different applications as found in the literature. The general outlook is provided in Sect. 5. Finally, Sect. 6 provides a conclusion.

2 Overview of RL

2.1 Development of RL

The concept of RL is derived from the early work of Bush & Moseller and Rescorla & Wagner in rhesus monkey striatum. In order to perform a voluntary movement, the striatum present in the mid-brain of the monkey takes an action. The selection of action is based on signals received from the cortex. The signal generated by the cortex is based on the choice selected from a different set of choices. After the accomplishment of the movement, an output signal is received. Dopamine neurons present in brain evaluate a reward prediction error i.e., RPE = $(r(t) - w_i(t))$. RPE is the difference between experience output and the value obtained from the striatum. This complete process is captured by the Rescorla-Wagner equation (Rescorla et al. 1972; Averbeck and Costa 2017), and it is given by (1),

$$w_i(t+1) = w_i(t) + \gamma(r(t) - w_i(t)) \tag{1}$$

Equation (1) summarizes the learning process, where the update is controlled by the learning parameter γ. Extension of Rescorla-Wagner equation leads to temporal-difference (TD) rule, where agent decides an action based on the state it encounters. Thus, TD rule at action i is represented by (2),

$$w_i(s_t) \leftarrow w_i(s_t) + \gamma(r(t) - w_i(s_t) + \lambda w_i(s_{t+1})) \tag{2}$$

where variable s_t is state at t, and λ is the discount rate for future estimates of values. This generalization is very successful in finding the optimal policy in robotics. Comparative analysis/parallelism of RL in the biological and artificial system is presented in a survey paper (Neftci and Averbeck 2019).

2.2 Background of RL

RL is a sequential methodology, where learning process of agent takes place in stochastic environments. Basically, the agent learns about the optimal policy. The optimal policy defines which action is needed in current state. In order to generate an optimal policy, agents interact with the environment. Agent-environment interaction generates a trajectory of state-action-reward-next state. The complete process is working in discretization manner and it is modeled by Markov decision process (MDP), denoted by $< S, A, P, R, \lambda >$. At time t, the agent chooses an action from action-space A and execution of the action results agent to reach in next-state $S_{t+1} \epsilon \{S\}$. The transition of agent for current state S_t to next state S_{t+1} is governed by transition probabilities. Then, the agent receives a scalar reward, i.e., r_{t+1}, from an environment. The reward is defined by $R : S \times A \times S \rightarrow \mathbb{R}$. The goal of the agent is to maximize the total accumulated reward received at time t, represented as $\sum_{j=0}^{\infty} \lambda^j r_{t+j+1}$, where $\lambda \epsilon [0, 1]$ is the discount rate. Therefore, Overall return over policy is given by $\sum_{j=0}^{\infty} \lambda^j R(s_{t+j}, \pi(s_{t+j}), s_{t+j+1})$. Thus, expected return over policy is given by Eq. (3),

$$Q^{\pi}(s, a) = E\left\{ \sum_{j=0}^{\infty} \lambda^j r_{t+j+1} | s_t = s, a_t = a \right\} \tag{3}$$

Like most RL algorithms (Bertsekas 1995; Puterman 2014; Watkins and Dayan 1992; Sutton 1988), in temporal difference (such as Q-learning, SARSA etc.), agent learns the action-value function defined as $Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a)$. TD error δ_t is given by (4),

$$\delta_t = r_t + \lambda Q(s',a') - Q(s,a) \tag{4}$$

where $Q(s',a')$ & $Q(s,a)$ is expected value of state s' at time step $t+1$ and expected value of state s at time step t respectively. Here, TD error is used to update the iterative relation as given by (5),

$$Q(s,a) \leftarrow Q(s,a) + \gamma * \delta_t \tag{5}$$

As per literature, Q-learning uses $max(Q(s',a'))$ instead of $Q(s',a')$ as compared to SARSA algorithm. It makes Q-learning and SARSA working as offline and online algorithms respectively. The major disadvantage of these algorithms is that the expected future return is not exact because these returns are truncated to zero after some time steps. So, these algorithms may not yield good outcomes, especially for highly sensitive robotic applications. Thus, Deep learning techniques are integrated with RL for better estimation.

2.3 Basic RL algorithms

In real-life applications like robotics, it is impossible to store the exact value function for every state-action pairs separately. Because the learning process is in a continuous state and action space. So, most RL algorithms practically use function approximation to cover all states and actions. In literature, we found some parametrized RL algorithms (Grondman et al. 2012). Here these algorithms are classified into three groups: Actor, critic & Actor-critic methods. The meaning of actor and critic are policy and value functions respectively.

2.3.1 Critic-methods

Variance is present in expected returns because of estimation. So, in order to reduce the variance in the estimation of expected returns, the critic method uses TD learning (Boyan 2002). They select an action that provides the highest reward known as greedy actions. Then it is used for deriving the policy (Sutton and Barto 2018). However, in order to find the best action, there is a need for optimization in every state that the agent encounters. This is a significant drawback because it requires an intensive computation, especially when the action space is continuous. Thus, the critic method uses the discretization of the action space. However, it undermines the ability of continuous action for finding the true optimum. Basically, critic-methods like Q-learning (Watkins 1989; Bradtke et al. 1994) and SARSA algorithm (Rummery and Niranjan 1994), uses a state-action value function only and no explicit function for policy. Thus, the optimal value function is obtained by approximating the solution of bellman equation (Sutton and Barto 2018). A deterministic policy based on the greedy search is obtained using optimization over state-action value function given by (6),

$$\pi(s,a) = arg\max_{a} Q(s,a) \tag{6}$$

The resulted policy will not provide a guaranteed near-optimal for the approximated value function. As per literature, Q learning and SARSA doesn't converge to an optimal solution even for simple MDPs with approximator (Baird 1995; Gordon 1995; Tsitsiklis and Van Roy 1996). However, in some articles(Melo et al. 2008), authors provide the guarantee of convergence if trajectories are sampled according to on-policy distribution for some parameters function approximators. Conditions for convergence and bound on error is presented in (Tsitsiklis and Van Roy 1997).

2.3.2 Actor-methods

Actor-methods are primarily applied with parameterized policy (the advantage of these policies is that it can generate continuous actions). Thus, the optimization techniques can be directly applied over parameter space. However, optimization methods such as policy gradient suffer from slow learning (because it has a high variance in estimates of a gradient). Policy gradient techniques are generally known as actor-methods that don't use any stored value function (Gullapalli 1990; Williams 1992). Here, the goal of RL is to maximize the total return. Reward obtained can be presented by discounted reward and average reward (Bertsekas 1995).

In the Discounted reward setting, Cost function C is defined as the expected total rewards that start from initial state s_0 over some policy $\pi(s,a)$ as given by Eq. (7),

$$C(\pi(s,a)) = E\left\{ \sum_{j=0}^{\infty} \lambda^j r_{j+1} | s_0, \pi(s,a) \right\} \tag{7}$$

Average reward per time step over some policy $\pi(s,a)$ is given by Eq. (8),

$$C(\pi(s,a)) = \lim_{m\to\infty} \frac{1}{m} E\left\{ \sum_{j=0}^{m-1} r_{j+1} | \pi(s,a) \right\} \tag{8}$$

In policy gradient method, policy $\pi(s,a)$ is parameterized by parameter $l \,\epsilon\, R^k$. Consider, cost function given by (7) and (8) are function of l. Thus, gradient of the cost function (assume the cost function is continuous) w.r.t. l is given by (9),

$$\nabla_l C = \frac{\partial C}{\partial \pi_l} \frac{\partial \pi_l}{\partial l} \tag{9}$$

Thus, the local optimal solution obtained by updating the rule iteratively as (10),

$$C_{j+1} = C_j + \gamma \nabla_l C_j \tag{10}$$

The advantage of the actor method is that it allows the policy evaluation in continuous action space. And also provides a strong convergence property (Sutton and Barto 2018; Peters and Schaal 2008). The learning rate γ must satisfy the equation (11) for making the gradient free from bias.

$$\sum_{j=0}^{\infty} \gamma = \infty \quad \& \quad \sum_{j=0}^{\infty} \gamma^2 < \infty \tag{11}$$

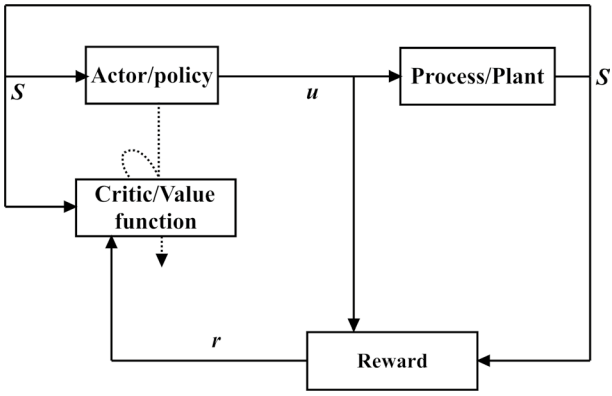

Fig. 2 Framework of the Actor-critic Method. Here, the dashed line represents Critic is responsible for updating of actor and itself (Grondman et al. 2012)

Table 1 Salient features of critic, actor, actor-critic method

Methodology → features ↓	Critic-method	Actor-method	Actor-critic method
Algorithm	Q and SARSA	Gradient	Q, SARSA and Gradient
State-action value function	Stored Function use	No stored Function use	Stored Function use
Explicit policy function	No Policy	Parametrized Policy	Parametrized Policy
Action space	Discrete	Continuous	Continuous
Convergence	Weak	Strong	Strong
Variance in Estimated grad.	-Na-	Large	Low

However, the estimated gradient has larger variance and calculated gradient have no knowledge of past estimates, a major disadvantage of actor-methods (Riedmiller et al. 2007; Peters et al. 2010; Sutton et al. 2000).

2.3.3 Actor-critic-methods

The actor-critic method combines the advantages of both actor and critic methods. It brings parametrized policy for the advantage of computation in continuous action space from actor method and low variance in the estimation of return from the critic-method. Thus, the critic's estimate allows the actor to update gradients with lower variance. It also improves the learning process. In comparison to the other two methods, actor-critic provides a good convergence property (Konda and Tsitsiklis 2000). Here, the policy is updated in the gradient direction with a small learning rate. Thus, the small change in value function leads to less oscillatory behavior of policy (Baird III and Moore 1999). The schematic framework of actor-critic is shown in Fig. 2. Learning framework has two parts: (a) actor (policy) (b) critic (value function). At a given state s, actor-part generates a control input u. Whereas, critic-part process the obtained reward. The actor is

updated after some steps of policy evaluation, using the information gained from critic. In-depth knowledge of actor-critic methods can be obtained from (Konda and Tsitsiklis 2000; Witten 1977; Barto et al. 1983). Comparative analysis of actor-method, critic-method, and actor-critic method are given in Table 1.

2.4 DeepRL algorithms

In the past, RL is limited to the lower-dimensional problem because of complexity in memory and computation. In recent years, the power of deep neural network enables us to use new tools, such as *function approximation* and *representation learning*, to overcome the limitation of RL. Therefore RL is integrated with deep learning. Resulted DeepRL helps in scaling-up the RL computation into higher-dimensional problems. In literature, Deep Q-Network (DQN) is developed by google's deep-mind Mnih et al. (2015a). The salient contribution of DQN is: (1) experience replay and target network stabilize the training of value function approximation (2) only minimum knowledge is required to design end-to-end RL. The major drawback of DQN is that it over-estimate the value function because it uses max-operator for both selection and evaluation of an action. So, in order to overcome the problem of over-estimation, Van Hasselt et al. (2016) proposes a Double-DQN algorithm. It uses a max operator for the selection of actions only. Several survey papers on DeepRL applied in various applications have reported in (Nguyen et al. 2017; Liu et al. 2019; Nguyen et al. 2018). The evolution of DeepRL is shown in Fig. 3.

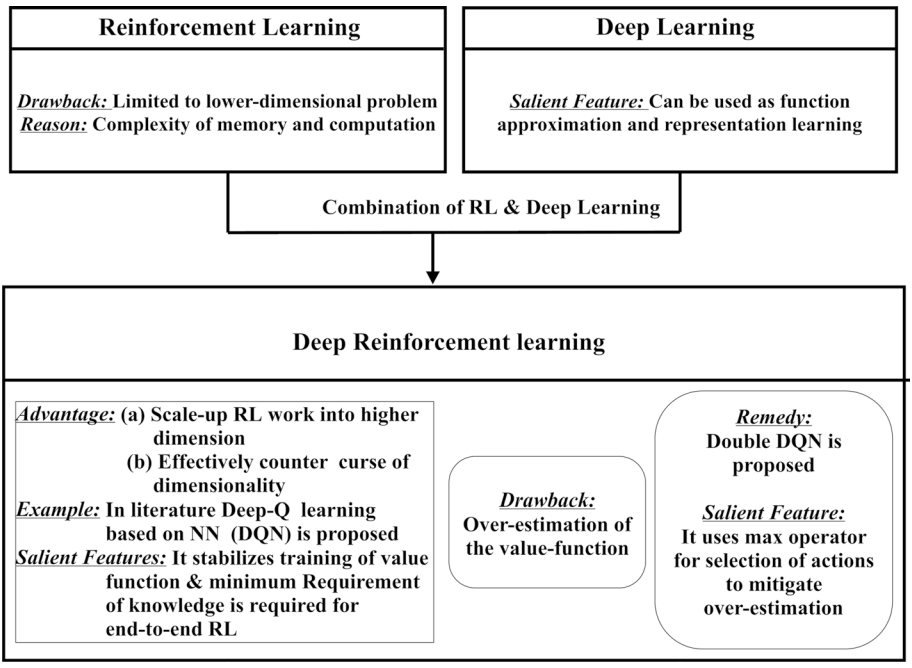

Fig. 3 Evolution of DeepRL

2.5 Multi-agent based RL algorithms

Agents in multi-agent systems are pre-programmed by using the knowledge of domain experts. Pre-programming of agents behaviour before encountering the different situation in complex environments is a very tedious task. Thus, multi-agent RL algorithms are developed. It is found in literature that multi-agent RL have been applied in robotics, economics, data mining, etc (Vlassis 2007; Parunak 1999; Stone and Veloso 2000; Yang and Gu 2004). Basically, multi-agent RL algorithms are derived from model-free algorithms like temporal-difference (specifically Q-learning). In multi-agent RL, MDPs is represented by tuple $< S, A_1, \ldots A_m, P, R_1, \ldots, R_m, \lambda >$. where, m is the number of agents; S: sets of the discrete environment states; A_i, $i = 1, \ldots, m$ are discrete sets of actions defined by joint action set $A = A_1 \times A_2 \cdots \times A_m$; State transition probability is defined as $P : S \times A \times S \rightarrow [0, 1]$ and $R_i : S \times A \times S \rightarrow \mathbb{R}, i = 1, 2, \ldots m$ are reward functions of all agents.

In Multi-agent RL, transition of states is a result of joint actions, i.e., $A_k = [A_{1,k}^T, \ldots A_{m,k}^T]^T$, $A_k \epsilon A$, $A_{i,k} \epsilon A_i$. Policies $\pi_i : S \times A_i \rightarrow [0, 1]$ forms the joint policy π. Q-function depends on joint action and policy defined as $Q_i^\pi : S \times A \rightarrow \mathbb{R}$. In multi-agent RL framework, the behavior of agents is given by the values of individual reward as: (1) If $R_1 = \cdots = R_m$ then the stochastic behavior of the agent is said to be fully-cooperative, it means the goal of all agents is to maximize the total reward (2) If $m = 2$ and $R_1 = -R_2$, then the stochastic behavior of the agent is said to be fully-competitive, it means the goal of agents is opposite to each other (3) In some cases, the stochastic behavior of the agent is neither fully-cooperative nor fully-competitive, said to be mixed behavior. Depending on agent awareness and type of task assigned, multi-agent RL algorithms are categorized as given in Table 2 (Bu et al. 2008). The goal of multi-agent RL is to make a balance in adaptiveness and stability of learning between agents. Here, stability and adaptiveness ensure the stationary policy and improvement in performance when agents change their policies respectively (Bowling and Veloso 2002; Greenwald et al. 2003; Hu and Wellman 2003; Bowling and Veloso 2001).

2.6 Human-centered RL algorithms

The major drawback of traditional RL is that the learning process is very slow. At the beginning of the learning process, RL needs a lot of trials and errors for finding an optimal policy. However, for real-time applications like real robotics, failure at the beginning may cause large costs. So, reward shaping techniques are developed for the learning of agent in complex robotics task (Ng et al. 1999; Thomaz and Breazeal 2008; Tenorio-Gonzalez et al. 2010). In real-world applications, robots need to learn how to perform a specific task and learn according to human likings. Thus, the human-centered algorithm has been developed

Table 2 Multi-agent RL algorithm based on awareness degree of agent and task type (Bu et al. 2008)

Awareness degree of Agent → Task type ↓	Independent	Tracking	Aware
Cooperative	Co-ordination free	Co-ordination based	Indirect co-ordination
Competitive	Opponent free	–	Opponent aware
Mixed	Agent-independent	Agent-tracking	Agent-aware

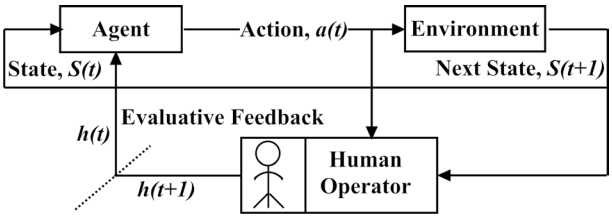

Fig. 4 Framework in human-centered RL algorithm (Li et al. 2019)

Table 3 Comparison of various human-centered RL algorithm

Methods → features ↓	Categorical feedback	Interactive shaping	Policy feedback
Model of feedback distribution	Required	Not required	Required
Tested	Human feedback	Human feedback	Run with traditional RL
Effectiveness with Human Feedback	Good	Good	Further exploration needed

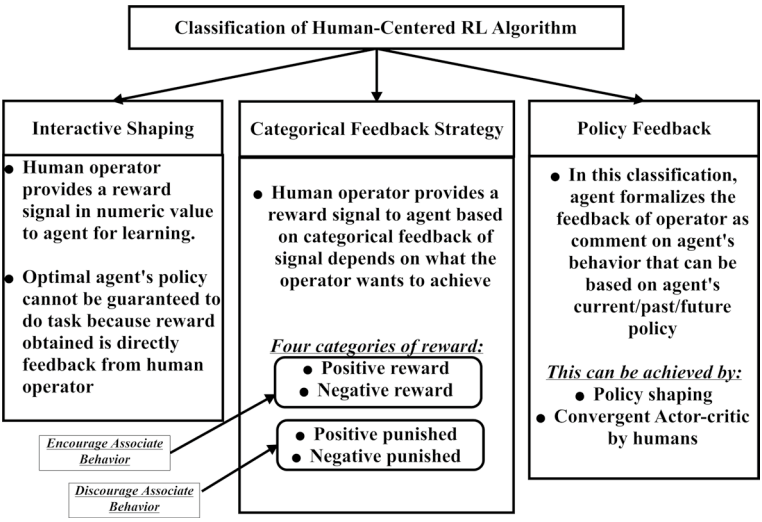

Fig. 5 Human-centered RL algorithm

(Ho et al. 2015; Li et al. 2019). Human-centered RL algorithms use reward shaping techniques. The agent learns the optimal policy using evaluative feedback from a human operator. This framework helps agents to speed up the learning process, especially feedback from ordinary people. In these algorithms, agents get evaluated feedback from the human observers which quantifies the quality of the selected action. Agent updates the policy online based on feedback. The complete framework of Human-centered RL is shown in Fig. 4. Depending on human-feedback, comparison and classification of human-centered RL are shown in Table 3 and Fig. 5 respectively.

2.7 Deep neuroevolution RL

Generally, DeepRL networks are trained using policy gradient learning algorithms. It has two drawbacks: (1) computation cost is high due to deep learning networks, and (2) tuning of network parameters is done by gradient methods. However, due to the structure of the deep network, the objective function is non-convex in most cases. Thus, the solution converges to a local optimum point. Evolutionary algorithm substitutes gradient methods for tuning of DeepRL networks. However, an evolutionary algorithm can be considered as policy gradient methods because it is similar to finite-difference-approximation of the gradient. Thus, neuroevolution comes into the picture. Neuroevolution is a technique that is used to design the architecture of the neural networks and learning rules which is based on evolutionary principles in the biological brain. It ultimately includes the learning of hyperparameters, activation function, forming of new architecture with time. Different algorithms are developed with time such as, neuroevolution of augmenting topologies (NEAT) that evolves both weights and architecture of ANN. However, it has encoding challenges in large ANN's. So, HyperNEAT is developed to address encoding issues (Such et al. 2017; Stanley et al. 2019). Basically, these methods tune a large number of parameters simultaneously rather than one parameter at a time. Thus, when neuroevolution integrated with DeepRL, it leads to generalized artificial intelligence, which is advantageous to produce more advanced intelligent robots.

3 Applications

In this section, the authors have discussed the application of RL in robotics. Here, the applications are categorized based on air, under-water, and land-based usage. Figure 6 shows the application of various RL based learning in robotic science based on usage as per literature.

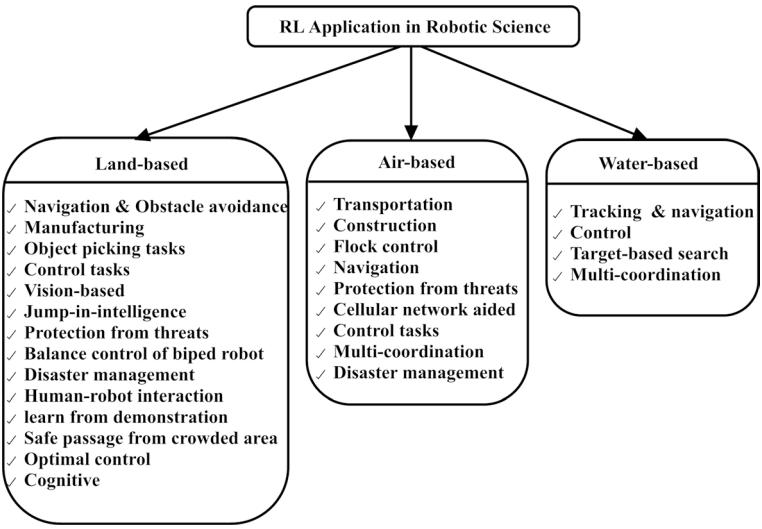

Fig. 6 Application of RL in Robotic Science

1. *Under-water based robots* This section focused on the application of RL in underwater robotics. The autonomous underwater robot is very advantageous in the study of deep-sea, i.e., minerals under the crust of the earth, aquatic life, etc. However, it is very difficult to control the autonomous underwater robot because the environment is dynamic. Thus, the authors have developed various architectures for underwater robots as reported in literature (Palomeras et al. 2012; El-Fakdi and Carreras 2013; Yu et al. 2015b; Hu et al. 2019; Cao et al. 2019; Carlucho et al. 2018; Cheng and Zhang 2018; Frost et al. 2015).

   (a) *Tracking & Navigation* Cables are laid widespread in the deep sea for communications or the internet. There is a necessity to track the cables for fault detection or required maintenance. Thus, in an article (Palomeras et al. 2012), authors have developed a hybrid approach using a natural actor-critic (NAC) algorithm with LSTD-Q(λ) for cable tracking mission. Likewise, the two-step learning approach based on an actor-critic algorithm for visual-based cable tracking is developed in El-Fakdi and Carreras (2013). Here the hydrodynamic model of the vehicle is used. Firstly, a policy is learned in simulation. Then, a learned policy is transferred to Ictineu AUVs in a real environment. In deep-sea, there is a requirement for effective control mechanisms for an autonomous system for navigation and obstacle avoidance. Episodic natural actor-critic (ENAC) algorithm is implemented for path planning and navigation in the deep sea for marine archaeology (Frost et al. 2015). In article (Cheng and Zhang 2018), a concise DeepRL obstacle avoidance (CDRLOA) based on avoidance reward algorithm is presented for navigation in an unknown environment.
   (b) *Control* The effective control mechanism consists of adaptiveness with parameter variation/noise is required for AUVs. A combined framework of DeepRL and Actor-critic algorithm is applied on Nessie VII AUVs for adaptive control (Carlucho et al. 2018).
   (c) *Target-search* Sometimes it is necessary to make target based search in deep-sea. Thus, the authors have developed an integrated approach of DeepRL and DQL algorithm based on dual Q-network for target-based search and implemented on Neptune-I AUVs (Cao et al. 2019). In article Hu et al. (2019), authors have developed an integrated methodology of temporal-difference with deterministic policy gradient (DPG) algorithm for plume-tracing task. Here, a model of the chemical plume is based on probabilistic descriptions of spatial and temporal evolution is used.
   (d) *Multi-coordination* In complex environments like deep-sea, it is necessary to make proper coordination between various AUVs. Thus, authors in Yu et al. (2015b) have developed a behavioral-based hierarchical architecture composed of fuzzy logic and Q-learning. The developed approach is tested on a 2vs2 water polo game.

Table 4 comprises the application of RL in underwater robotic.

2. *Air-based robots* Air-based robots have found numerous applications like surveillance, disaster management, real-time communication, multi-coordination operations, etc.

Table 4 Application of RL in Underwater Robotics

References	Vehicle	Objective	Technique	Domain
Palomeras et al. (2012)	Ictineu AUV	Cable tracking mission	Hybrid RL: NAC + LSTD-Q(λ) + Multi-layer: reactive, execution, mission layer	Tracking
El-Fakdi and Carreras (2013)	Ictineu AUV	Visual based cable tracking mission	Actor-critic	Tracking
Frost et al. (2015)	–	Path planning for marine aechaeology	Episodic NAC	Navigation and search
Cheng and Zhang (2018)	–	Navigation in unknown environment	CDRLOA	Navigation
Carlucho et al. (2018)	Nessie-VII AUV	Adaptive low-level control	DeepRL + actor-critic	Control
Cao et al. (2019)	Neptune-I AUV	Target based search	DeepRL + DQL based Dual stream Q-network	Target-search
Hu et al. (2019)	–	Plume-tracing task	Temporal difference + Deterministic policy gradient	Target-search
Yu et al. (2015b)	2vs2 water polo game	Coordinate between multiple AUVs	Multi-agent: Behaviour-based hierarchical architecture + fuzzy + Q-learning	Multi-coordination

Future aerial robots need to operate more intelligently with uncertainty present in an environment like turbulence present in the air, loss of GPS connection, etc. Therefore, the vehicle needs to operate efficiently and autonomously based on its own-board sensors. As reported in literature (dos Santos et al. 2015; Xiao et al. 2017; Zhu et al. 2018; Hu et al. 2018; Wang et al. 2019; Zeng et al. 2016; Yin et al. 2019; Wu et al. 2019; La et al. 2014; Faust et al. 2014; Hung and Givigi 2016; Hwangbo et al. 2017; Lambert et al. 2019; Shi et al. 2016), authors have developed various topologies to address the difficulty faced by aerial vehicles. In this section, we have discussed the application of RL in Air-based robots as given below:

(a) *Transportation* Delivery of time-sensitive goods like body organs, etc. in time, is a very challenging task. In article (Faust et al. 2014), the authors have proposed a model-free based RL with continuous inputs for rendezvous cargo delivery task by quadcopter. In which, quadcopter need to carry a suspended cargo and handover to a land-based robot in a swing-free fashion.

(b) *Construction* UAVs can be advantageous on the construction site. It improves communication, safety, or does survey by taking real-time imaging of the field. Thus, an adaptive scheme for the planning of construction using quad-rotor, an aerial vehicle based on the RL and heuristic is presented in (dos Santos et al. 2015).

(c) *Flock Control* In a flock, flocking-followers need to follow the leader of the flock. In article (Hung and Givigi 2016), authors have purposed a flock control mechanism based on Q-learning, where the learning rate of followers is variable based on Peng's Q(λ).

(d) *Navigation* Navigation of UAVs in a dynamic environment is a very big challenging task. In article Zhu et al. (2018), the authors have proposed DeepRL based DQN algorithms for shepherd game. Where the aerial robot needs to establish contact with ground vehicles and sequentially drive them to safe regions with obstacle avoidance in the path. Likewise, in article Wang et al. (2019), the authors have proposed a DeepRL algorithm based on policy gradient within an actor-critic framework for navigation of UAVs in complex environments. In article (Shi et al. 2016), the author proposed an integrated approach of image-based visual servoing (IBVS) with fuzzy logic based RL for quad-copter.

(e) *Protection from threats* Cyber-attack on UAVs have posed a great threat to its operation. To mitigate this challenge, authors have proposed a mechanism in Xiao et al. (2017), Deep Q-learning is used for speeding the learning process of agents in UAVs to counter when it is under-attack, without knowing the dynamics of the attacker model.

(f) *Cellular network aided* In the article, Hu et al. (2018) and Yin et al. (2019), authors have proposed real-time sensing with UVAs to serve as aerial bases for the ground station for effective communication.

(g) *Control task* Control of quadcopter in a stochastic environment is very challenging. In the article, Hwangbo et al. (2017) and Lambert et al. (2019), authors have presented a policy optimization and model-based DeepRL with Model predictive control, respectively for the control of quadcopter.

(h) *Multi-coordination* In article (Zeng et al. 2016), authors have presented an energy-efficient and continuous movement control (E^2CMC) algorithm for control of multiple drones. Its salient feature is that the multiple drones can coordinate each other in an efficient manner. Likewise, in article (Viseras and Garcia 2019), the authors have

Table 5 Application of RL in air-based Robotic

References	Vehicle	Objective	Technique	Domain
Faust et al. (2014)	Quadrotor	Rendezvous cargo delivery task	Model-free RL + Continuous inputs	Transportation
dos Santos et al. (2015)	Qball-X4 quadrotor	Planning of Construction task	Adaptive Scheme + RL + heuristic search	Construction
Hung and Givigi (2016)	Flocks of UAVs	Learn control policy to follow leader	Q-learning + Peng's Q(λ)	Flock control
Zhu et al. (2018)	Shepherd game	Navigation and Obstacle avoidance	DQN + Lookup table	Navigation
Wang et al. (2019)	UAVs	Navigation	DQN	Navigation
Shi et al. (2016)	Quadrotor	Visual servoing	IBVS + fuzzy + Q-learning	Navigation
Xiao et al. (2017)	–	Protection from cyber-attack on UAVs	DQN	Threat protection
Hu et al. (2018)	Multi-UAVs	Trajectory planning for real-time sensing over cellular networks	Q-learning	Cellular network aided
Yin et al. (2019)	Multiple drones	Energy efficient cellular aided	DPG	Cellular network aided
Hwangbo et al. (2017)	quadrotor	Control	Policy optimization algorithm	Control task
Lambert et al. (2019)	Quadrotor	Control	Deep learning + Model-based RL	Control task
Zeng et al. (2016)	Multiple drones	Intelligent trajectory with energy efficient	E^2CMC	Multi-coordination
Viseras and Garcia (2019)	3 quadcopters	Learn how to gather new information	DeepIG	Multi-coordination
Wu et al. (2019)	UAVs	Target-based disaster management and search	Snake algorithm	Disaster management

developed a DeepIG algorithm for a quadcopter. It helps in learning the quadcopters to learn the mechanism of gathering new knowledge in a coordinated manner. Here, the multiple robots use the principle of information-gain mechanism for either mapping or gathering information about unknown terrain.

(i) *Disaster Management* In a complex disaster environment, UAVs can be employed to target based search. In article (Wu et al. 2019), the author proposed a DeeRL based snake algorithm for searching the targets like injured humans.

Table 5 comprises the application of RL in air-based Robotic.

3. *Land-based robots* The autonomous land-based vehicle is a robot that operates without a human operator. The vehicle uses high-end technology based on AI. Though, land-based robots find a wide range of applications in the real-world such as picking objects, navigate through a crowded area without collision, manufacturing sites, etc. However, it faces some challenges like uncertainty present in the real environment, disturbances presented in feedback signals obtained from vision-based sensors, etc. In literature, authors have presented many approaches for control of land-based autonomous vehicles as described below:

   (a) *Navigation and obstacle avoidance* Navigation of the robot through a crowded area and finding the optimal path to reach the end goal of avoiding the collisions are very difficult tasks. Many learning mechanisms are proposed in the literature (Xu et al. 2011; Whitbrook et al. 2007). Fuzzy controllers are most prominent for solving this task (Beom and Cho 1995; Yung and Ye 1999; Gu and Hu 2007; Li et al. 2010; Er and Deng 2005). Hereby fuzzy rules and parameters of the controller are tuned by RL. Specifically, the hybrid approach for obstacle avoidance is presented for the adaption of robots to the new environment without human intervention. The developed approach is based on the actor-critic structure (Er and Deng 2005; Hwang et al. 2009). As per literature, some authors have proposed a fuzzy-based neural network, and its parameters are tuned using model-based RL, DeepRL, etc., for obstacle avoidance and navigation (Zalama et al. 2002; Ye et al. 2003; Meeden 1996; Antonelo and Schrauwen 2014; Wang et al. 2018b; Ohnishi et al. 2019; Markova and Shopov 2019). In article (Bejar and Moran 2019), the authors have presented a fuzzy-based neuro controller using DeepRL for reverse parking of truck in parking plot. It is important to navigate the mobile robot in complex environments. Thus, in an article (Juang and Hsu 2009) authors presented a Q-learning enabled based on ant-optimization. A policy network based on a deep Q-network is presented for path planning in the dynamic place (Lv et al. 2019). In order to increase the robustness for initial states and learning ability of mobile robots, a quantum-inspired RL is proposed in literature (Dong et al. 2010). A multi-objective neuro-evolutionary approach is proposed for autonomous driving, where perception-based planning deep neural network is used for estimation of desired state trajectories over a finite prediction horizon (Grigorescu et al. 2019). Response time to obstacle avoidance/navigation task should be less, thus in article (Plaza et al. 2009) authors have presented an integration of cell mapping technology and Q-learning for WMV car-robot. When the obstacles are moving like in crowded places, navigation becomes a very tough task. In (Lin et al. 2013),

weighted based Q-learning is presented for multi-robots. Autonomous RL based learning for multiple interrelated tasks for the iCub robot for obstacle avoidance is presented (Santucci et al. 2019), where learning is automated.

(b) *Manufacturing* In hazardous places like manufacturing sites (high temperature), etc., robots can be used effectively in place of human operators. Thus, in literature, some authors have proposed a mechanism for smoothing of metal surfaces(Tzafestas and Rigatos 2002). It is based on sliding mode control and a fuzzy controller. Controller parameters are tuned by RL. Controlling mechanism that involves multiple learning levels for the execution of partially changeable tasks like industrial tasks, etc., is also presented in literature (Roveda et al. 2017). Here, controller parameters are learned by iterative learning and RL.

(c) *Object picking tasks* In the industry or household work, robots need to pick things from one place and put that in another place. As per literature, An integrated approach of genetic programming and RL is developed (Kamio and Iba 2005). The proposed approach is applied to the four-legged robots and humanoid robots for box-moving applications. Likewise, a hybridized algorithm based on bio-inspired (Farahmand et al. 2009) i.e., RL, cooperative evolution, and culturally based memetic algorithm, is developed. It helps in the automatic development of behavior-agent. The effectiveness of the algorithm is evaluated on object-picking tasks. Likewise, the interactive RL approach is proposed for a simulated robot for cleaning of table (Cruz et al. 2016). Some authors in literature developed a learning mechanism for picking objects correctly without slipping. In the proposed mechanism, the author uses RL, plus visual perception based on reactive control (Falco et al. 2018). Hereby, the objective of RL is to fulfill the in-hand manipulation goals and minimization the intervention of reactive control. In (Rombokas et al. 2012), RL enabled synergistic control of ACT hand is presented. Here, learning of hand is based on path integral policy-improvement for basic tasks such as sliding the knob/switch. In order to distinguish between things and pick the right one in moving objects. A DeepRL based learning for the 3AT robot is presented in (Yang et al. 2018). Here, the control is in continuous action space. Likewise, a 7-DoF arm of ABB Yumi is presented to learn the policies for grasping, reaching, and lifting tasks based on DeepRL (Breyer et al. 2019).

(d) *Control tasks* Controlling the higher degrees of freedom (DOF) system is a very complex task. In literature, the authors have accomplished the control of different DOF robots (Foglino et al. 2019; Wang et al. 2013; Hazara and Kyrki 2019; Xi et al. 2019; Caarls and Schuitema 2015; Carlucho et al. 2017; Heidrich-Meisner and Igel 2008; Da Silva et al. 2012; Muelling et al. 2010). Likewise, 7-DOF simulated robot has been controlled using model-free RL (Stulp et al. 2012) and learning of controller for a 32-DOF humanoid robotic system based on policy gradient RL for the stair climbing task is developed (Fu and Chen 2008). Motion control of two-link brachiation and 2-DOF SCARA robot is presented in Hasegawa et al. (1999); Sharma and Gopal (2008). Here, fuzzy-based self-scaling RL is used for faster convergence and robustness against disturbance. Likewise, balancing of the robot on a rotating frame is a very challenging task. Thus, integration of model-based RL with model-free RL is proposed for balance control of a biped robot on a rotating frame where the velocity of the rotating frame is not known in advance (Xi et al. 2019).

(e) *Vision based* A Controller for wheeled mobile robots based on robust vision-based with Q-learning developed (Wang et al. 2010). Here, the vision sensor is used to

provide a feedback signal back to the controller. Likewise, a robotic soccer goalkeeper for catching the ball is learned based on experience replay (Adam et al. 2011). In article (Gottipati et al. 2019), authors developed a deep active localization based on DeepRL and end-to-end differential method. Here the perception and planning for complex environments like a maze is developed. The end-to-end differentiable method is presented for the training of agents in simulations and then the trained model is transferred to the real robot without any refinement. The proposed system is composed of two modules: (1) Convolutional Neural Network (CNN) for perception and (2) DeepRL for planning.

(f) *Jump in the intelligence of robot* Tabular model-free online algorithm,i.e., SARSA (λ) learning is applied to games like First Person Shooting (FPS), in which a lot of different learning methodology is combined with a RL controller to obtain new bot artificial intelligence (McPartland and Gallagher 2010).

(g) *Protection from threats* In the real world, it is necessary to make an AI system safe from threats. In literature (La et al. 2014), a multi-robot system which integrates the reinforcement learning and flocking control that ensures the agent to learn about the avoidance of enemy/predator in between the path while maintaining the communication and network connectivity with each other.

(h) *Disaster management* In a natural disaster, many lives are lost due to inadequate amount of human resources. Therefore, integrated AI-enabled robots can aid the disaster management team for rescue beings. Thus, as per literature authors have developed a robust architecture based on RL, which provides a semi-autonomous control for rescuing robots in unknown environment (Doroodgar et al. 2014), aim of the controller is to enable the robot to self-learning in unknown environments by its own experiences with the cooperation of human operators.

(i) *Human–robot interaction (HRI)* HRI has found numerous application in the literature (Modares et al. 2015; Li et al. 2017a; Ansari et al. 2017; Huang et al. 2019; Khamassi et al. 2018; Koç and Peters 2019; Deng et al. 2017). As per literature (Modares et al. 2015), a human–robot interaction framework is proposed to assist human operators for specific-task with better efficiency and minimum workload. Basically, the framework has two controllers. The first one is an inner loop controller, which makes unknown dynamics of robot behaving like a robot impedance model, then this model is converted into a linear quadratic regulator (LQR) problem. The obtained LQR problem is iteratively solved by RL. The second one is an outer loop controller, which consists of human and robot to perform a specific task. Likewise. in Li et al. (2017a), an adaptive impedance controller is developed based on the Lyapunov function, and then the complete framework is reformulated as an LQR problem when the human arm model is unknown. Afterward, the LQR problem is optimized using integral RL. It helps in minimizing tracking error. To address the issue of adaption of the robotic assistant rollator to patients in different situations. A model-based RL is proposed for adapting the control policy of robotic assistant (Chalvatzaki et al. 2019).

(j) *Learn from demonstration* In article (Hwang et al. 2015), biped robot learns from walking patterns. Two different Q-learning mechanisms are proposed, the first mechanism learns a policy to adjust its walking by the gait-to-gait analysis to make the robot stable, and the second mechanism for refinement of walking. This allows walking faster subject to constraints. It also reduces the dimensionality of action space for faster learning.

(k) *Safe passage from crowded area* When the dynamics of system and environment is unknown, then learning approaches are most suitable in designing a control policy for robots. However, learning from scratch required a lot of trial runs. These trails can damage the robot. Therefore, in literature (Koryakovskiy et al. 2018), an approach is proposed to mitigate this problem by integrating the RL with the model-predictive controller. Likewise, an algorithm is proposed for the safe passage of robot in crowded dynamic environments (Truong and Ngo 2017).

(l) *Optimal control* Parametrized batch RL algorithm based on the actor-critic architecture is designed for optimal control of autonomous land vehicles (ALVs) wherein, least-square batch updating rule is implemented for better efficiency (Huang et al. 2017). Likewise, event-triggering optimal controller is proposed to non-linear robotic arm based on identifier-critic architecture (Yang et al. 2017), where identifier network is composed of the feed-forward network (FFN) which aim to retrieve knowledge about unknown dynamics and complete framework of architecture is developed using RL.

(m) *Cognitive* In order to provide cognitive capability like human-level decision-making ability. A multi-task learning policy is proposed for the learning of the non-linear feedback policy that makes robot self-reliant (Wang et al. 2018a).

(n) *Miscellaneous* In recent times, the usage of robots in medical science is increased significantly. In article (Turan et al. 2019) author presented a magnetically actuated soft capsule endoscope with minimizing the trajectory tracking error based on DeepRL.

Table 6 comprises the different approaches integrated with RL in land-based robotic applications.

4 Reinforcement learning algorithms/mechanism

In this section, a brief discussion of RL algorithms applied in various applications in the previous section is presented.

1. *Sarsa (λ) Algorithm* Integration of sarsa algorithm with eligibility traces results in sarsa (λ) (McPartland and Gallagher 2010). Eligibility traces are used to speed up the learning process by allowing the past actions for taking the advantage from the current reward, and it also learn the sequence of actions. Algorithm 1 shows the steps of the Sarsa algorithm. $e(s,a) \leftarrow \gamma \lambda e(s,a)$ reduces the eligibility trace of past visited state-action pair. Thus, this reduction impacts the state-action pairs that have been visited in past and allows them to get more of the current reward.

Table 6 Application of RL in Land-based Robotic

References	Vehicle	Objective	Technique	Domain
Beom and Cho (1995)	LCAR mobile robot	Navigation and obstacle avoidance in complex unknown environment	Fuzzy logic + RL	Navigation and obstacle avoidance
Yung and Ye (1999)	Mobile robot	Navigation and obstacle avoidance in complex unknown environment	Fuzzy logic + RL	Navigation and obstacle avoidance
Li et al. (2010)	aiRobot-3 biped robot	Walking, tracing and chasing a ball	Fuzzy + PGRL	Navigation and obstacle avoidance
Bejar and Moran (2019)	Truck and trailer	Reverse parking in parking slot	DRL + fuzzy + neur-controller	Navigation
Juang and Hsu (2009)	Wheel mobile robot	Walk near to wall	RAOFC + Q-learning	Navigation
Gu and Hu (2007)	Quadrupled-robot	Learn soccer-playing behaviours in robo-cup domain	Fuzzy + RL	Navigation
Zalama et al. (2002)	Mobile robot	Behavioural navigation of robot	RL	Navigation and obstacle avoidance
Antonelo and Schrauwen (2014)	Mobile robot	Learn navigation in unknown partial observable environment	RNN + RL + Reservoir computing	Navigation
Wang et al. (2018b)	Mobile robot	Avoid dynamic objects and navigate safely	Deep NN + Q-learning	Obstacle avoidance and navigation
Lv et al. (2019)	–	Path tracking control	PN-DQN	Navigation
Meeden (1996)	Car robot	Local and global exploration	NN + RL	Navigation and obstacle avoidance
Dong et al. (2010)	Mobile robot	Robustness to initial states is improved	Quantum inspired RL (QiRL)	Navigation
Whitbrook et al. (2007)	P3DX robot	Navigation in maze world	Idiotypic-AIS network + RL	Navigation
Ohnishi et al. (2019)	Brushbot	Navigation in unknown environment	RL	Navigation
Grigorescu et al. (2019)	Autonomous land vehicle	Navigate in real world	Neuro-evolution + RL	Navigation
Santucci et al. (2019)	Simulated iCub robot	Automatic learning for various task	Q-learning	Obstacle avoidance

Table 6 (continued)

References	Vehicle	Objective	Technique	Domain
Er and Deng (2005)	Khepra II mobile robot	Learning of robot for wall-following and obstacle avoidance	Fuzzy + Actor-critic	Obstacle avoidance
Hwang et al. (2009)	Mobile robot	Safely pass in unstructured environment	Adaptable receptive module (ARM) based critic-actor	Obstacle avoidance
Markova and Shopov (2019)	Multi-agent	Transfer knowledge	Q-learning	Navigation and obstacle avoidance
Plaza et al. (2009)	WMV car-robot	Navigation and obstacle avoidance with fast response	Cell-mapping + Q-learning	Navigation and obstacle avoidance
Lin et al. (2013)	Multi-robot	Manoeuvring of various robots with self-awareness capability for communication with each other	Weighted behaviour Q-learning (WBQL)	Obstacle avoidance
Ye et al. (2003)	Virtual emulator	Learning of navigation through complex environment	Neural fuzzy + RL	Obstacle avoidance
Xu et al. (2011)	P3-AT wheeled mobile robot	Path tracking control	Hierarchical API	Navigation and avoidance
Roveda et al. (2017)	Universal robot UR 10	Learn various repetitive changeable task	RL	Manufacturing
Tzafestas and Rigatos (2002)	Two-arm manipulator	De-burring compliance task	SMC + FuzzyRL	Manufacturing
Kamio and Iba (2005)	Humanoid + four-legged robot	Box moving	Genetic programming + RL	Object picking
Farahmand et al. (2009)		Automatic generation of behaviour-based agents	RL + Cooperative co-evolution + Culturally inspired memetic algorithm	Object picking
Breyer et al. (2019)	7-DoF arm of ABB Yumi	Learn policies for grasping, reaching and lifting objects	DeepRL	Object picking
Falco et al. (2018)	Openbionics ADA hand	Learn to pick object without slipping	Reactive control + RL	Object picking
Rombokas et al. (2012)	ACT hand	Learn to slide the switch and knob	Path integral and policy improvement	Object picking

Table 6 (continued)

References	Vehicle	Objective	Technique	Domain
Yang et al. (2018)	3AT robot	Learn to distinguish and moving objects	DeepRL	Object picking
Cruz et al. (2016)	V-REP Simulator	Interaction feedback provides advantage in learning, Robot cleaning of table	Iteractive RL	Object picking
Hasegawa et al. (1999)	Two-link brachiation robot	Faster convergence and robustness against disturbance	Fuzzy + SSRL	Control task
Sharma and Gopal (2008)	2-DoF SCARA robot	Robust tacking control	Fuzzy + Q-learning	Control task
Foglino et al. (2019)	Grid world and block-dude	Optimize framework for task sequencing	RL	Control task
Wang et al. (2013)	ALV from toyota-L and cruise 4500	Cruise controller developed	Kernel-based LSPI	Control task
Fu and Chen (2008)	32-DoF humanoid robot (THBIP-I)	Stair climbing	Policy gradient	Control task
Hazara and Kyrki (2019)	MuJoCo + KUKA LBR	Transfer of skill, ball-in-a-cup and basketball task to real system	RL	Control task
Xi et al. (2019)	NAO robot	Balance control of robot on rotating frame	PILCO; MBRL + MFRL	Control task
Caarls and Schuitema (2015)	Two-link manipulator	Real time complex task	Parallel LLFQI and Parallel DYNA	Control task
Stulp et al. (2012)	7-DoF simulated robot	Learn based on experience without knowing dynamics	PI^2	Control task
Carlucho et al. (2017)	Pioneer 3AT	Adaptive PID Control	Incremental Q-learning	Control task
Heidrich-Meisner and Igel (2008)	Double cart-pole	Balancing of pole	CMA-ES + RL	Control task

Table 6 (continued)

References	Vehicle	Objective	Technique	Domain
Da Silva et al. (2012)	Robotic arm	Underactuated arm learned to accurately throw darts at apara-metrized location	PoWER	Control task
Muelling et al. (2010)	Anthropomor- phic robot arm	Learning the arm for table tennis and also predict the hitting point in time and space where, we want to hit the ball	PoWER	Control task
Gottipati et al. (2019)	Maze	Perception and planning	DeepRL + End-to-end differen-tial method	Vision based
Wang et al. (2010)	Pioneer Dx3 wheel mobile robot	Search the components in unstructured environment	IBVS + Q-learning	Vision-based
Adam et al. (2011)	Robotic soccer goalkeeper	Keeper learn to catch balls shot toward goal	Experience replay RL (EX-SARSA)	Vision-based
Zhu et al. (2017)	SCITOS mobile robot	Target-driven visual navigation	Actor-critic policy	Vision-based
McPartland and Gallagher (2010)	First person shooter-game	Navigation and combat task with other opponents	Tabular-SARSA (λ)	Jump in intelligence
La et al. (2014)	Multi-robots	Learn to avoid enemy/predator and reach to destination	Q-learning + Flock-control	Protection from threats
Doroodgar et al. (2014)	Mobile robot	Rescue robots in unknown and cluttered environment	MAXQ-HRL	Disaster management
Khamassi et al. (2018)	Baby robot	Perform task in dynamic envi-ronment	Active exploration + meta learning	Human–robot interaction (HRI)
Modares et al. (2015)	PR2 robot	Assist human to perform task in least time	Off-policy integral RL	Human–robot interaction (HRI)
Ansari et al. (2017)	Soft robot arm	Automate the bathing task for older peoples	Model-free RL + Iterated pris-oner's dilemma (IPD)	Human–robot interaction (HRI)
Li et al. (2017a)	Robotic exoskeleton	Assist human to complete coop-erative task	Integral RL	Human–robot interaction (HRI)

Table 6 (continued)

References	Vehicle	Objective	Technique	Domain
Huang et al. (2019)	HUALEX		PI^2 + Policy iteration	Human–robot interaction (HRI)
Chalvatzaki et al. (2019)	Robotic assistant rollator	Adaption of robotic assistant rollator to patients in different situation	MBRL + MPC	Human–robotic Interaction (Human-in-loop control)
Koç and Peters (2019)	Barrett WAM arm	Learning from demonstration	LSDP and cLSDP	Human–robot interaction (HRI)
Deng et al. (2017)	UR5 robot	Human–robot handover experiment	Model-based Q-learning	Human–robot interaction (HRI)
Hwang et al. (2015)	18 DoF biped robot	Learning from walking patterns and adapt to terrain	Q-learning	Learn from demonstration
Koryakovskiy et al. (2018)	Robot Leo	Learning a control policy such as damage to robot is avoided while training	Non-linear MPC and RL	Safe passage from crowded region
Truong and Ngo (2017)	Eddie mobile robot	Safe passage of robot in crowd	RL	Safe passage from crowded region
Huang et al. (2017)	HQ7 platform	Improve performance in slopy, flat, slippery and bumpy road	Parametrized batch Actor-critic (PBAC)	Optimal control
Yang et al. (2017)	Single-link robot		Event-triggered + RL	Optimal control
Wang et al. (2018a)	SMSM, TLMDCP and PAC	Make robot decision making upto human	Multi-task policy adversarial learning (MTPAL)	Cognitive
Yu et al. (2015a)	Gridwrold	Two-robot coordinated for navigation	RL	Multi-coordinated
Turan et al. (2019)	Magnetically actuated soft capsule endoscope (MASCE)	Control of MASCE with trajectory tracking error	DeepRL	Medical

Algorithm 1: Steps for Sarsa (λ) algorithm

Initialization: Q(s,a) = 0 and e(s,a) = 0 for all s,a
Procedure:
(a) Repeat for all each update step
- Set s' to the current state
- Select an action a'
- Take an action a' and get reward r
- Calculate TD-error,

$$\delta \leftarrow r + \lambda Q(s', a') - Q(s, a)$$

- Set e(s,a) ← 1
- Repeat for all s,a;
  - Q(s,a) ← Q(s,a) + β δ e(s,a)
  - e(s,a) ← γ λ e(s,a)
  - s ← s' and a ← a'

2. *Natural Actor-Critic (NAC) Algorithm* Basic policy gradient methods have shown their applicability in various applications. However, their implementation in real tasks suggests that its result are not much satisfactory. Hence, the NAC algorithm is developed. It integrates the advantage of both policy gradient and value function methods. It contains two parts, (a) First one is critic structure, which is represented as a value function and it is approximated as a linear combination of parameter l and basis function ϕ(s) i.e., $V^{\pi}(s) = \phi(s)l$, and (b) Second one is actor structure, where policy is represented over probability distribution defined as, $\pi(a|s)$. The goal of the RL algorithm is to maximize the expected return represented by (12),

$$C_l = E[(1-\gamma)\sum_{t=0}^{\infty} \lambda^t r_t | l] = \int_S d^{\pi}(s) \int_A \pi(a|s) r(s|a) ds da \tag{12}$$

where $d^{\pi}(s)$ is discounted state distribution. Actor-critic and various other policy iteration contain two parts, the first one is policy evaluation and the second is policy improvement. Thus, policy evaluation exploits the experienced data, and policy improvement utilizes the evaluation step to improve the current policy until the convergence is obtained.

Actor improvements: Actor's policy derivative is defined as $\nabla_l log\pi(a|s)$. Gradient of the equation (12) is given by (13),

$$\nabla_l C_l = \int_S d^{\pi}(s) \int_A \nabla_l \pi(a|s)(Q^{\pi}(s,a) - b^{\pi}(s)) da ds \tag{13}$$

Where $b^{\pi}(s)$ is defined as a baseline. The difference between $(Q^{\pi}(s,a) - b^{\pi}(s))$ is replaced by compatable function approximation parametrized by vector w defined by (14),

$$f_w^{\pi}(s,a) = (\nabla_l log\pi(a|s))^T w = Q^{\pi}(s,a) - b^{\pi}(s) \tag{14}$$

Combining equation (13) & (14) yields (15),

$$\nabla_l C_l = \int_S d^{\pi}(s) \int_A \pi(a|s) \nabla_l log\pi(a|s) (\nabla_l log\pi(a|s))^T da ds w = F_l w \tag{15}$$

However, computation of F_l is expensive because discount state distribution is unknown. By using fisher metric, $\nabla_l C_l$ is approximately equal to w. Therefore, the estimation of parameter w is necessary. Hence, policy parameters are updated as: $l_{i+1} = l_i + \gamma w$.

Critic estimation: Critic evaluates whether policy π is good or bad. Accordingly, it provides the basis of improvement, i.e., change in parameter l as $\Delta l = \gamma w$. Here, compatable function approximation is used and that is represented by advantage function as: $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$. The major disadvantage of this function is that it cannot be learned from TD-methods. Therefore, compatable function is approximated using rollouts or least square minimization methods. It gives the estimated state-action value function. However, these methods required training data sets and also have high variance. Bellman equation is utilized as an alternative and it is defined by (16),

$$Q^\pi(s,a) = A^\pi(s,a) + V^\pi(s) \tag{16}$$

Substituting, $A^\pi(s,a) = f_w^\pi(s,a)$ and $V^\pi(s) = (\phi(s))^T l$ in equation (16). State-action value function at time t is defined by (17),

$$\begin{aligned} Q^\pi(s_t,a_t) &= (\nabla_l log\pi(a_t|s_t))^T w + (\phi(s_t))^T l \\ &= r(s_t,a_t) + \lambda(\phi(s_{t+1}))^T l + \epsilon(s_t,a_t,s_{t+1}) \end{aligned} \tag{17}$$

Equation (17), leads to two algorithms:

(a) *NAC with LSTD-Q(λ)* The solution of equation (17) is obtained using LSTD-Q(λ) policy evaluation algorithm. Thus, define a new basis for the value function and it is given by (18),

$$\begin{aligned} \hat{\phi}_t &= [\phi(s_t)^T, \nabla_l log\pi(a_t|s_t)^T]^T \\ \tilde{\phi}_t &= [\phi(s_{t+1})^T, 0]^T. \end{aligned} \tag{18}$$

A new basis helps in the reduction of bias and variance in the learning process. It estimates the state action function. In critic-part, it helps in an approximation of equation (17), yields two results: the first one is parameter l and the second one is w. Thus, the policy parameters are updated as $\Delta l_t = l w_t$.

(b) *Episodic NAC* Guarantee of unbiased estimate for natural gradient is necessary. So, the equation (17) is summed along a sample path as represented in (19),

$$\sum_{t=0}^{N-1} \lambda^t A^\pi(s_t,a_t) = V^\pi(s_0) + \sum_{t=0}^{N-1} \lambda^t r(s_t,a_t) - \lambda^N V^\pi(s_N) \tag{19}$$

As $N \rightarrowtail \infty$ or episodic task, the last term is eliminated. Therefore, each rollout provides one equation. It becomes a regression problem. For non-stochastic tasks the gradient is obtained after $dim(l) + 1$ rollouts.

For more information, refer to article (Peters et al. 2005), where authors derive the natural actor-critic algorithm based on LSTD-Q(λ) and episodic.

3. *Off-policy integral RL* Algebraic Riccati Equation (ARE) is given by,

$$0 = A^T P + PA + PBR^{-1}B^T P + Q$$

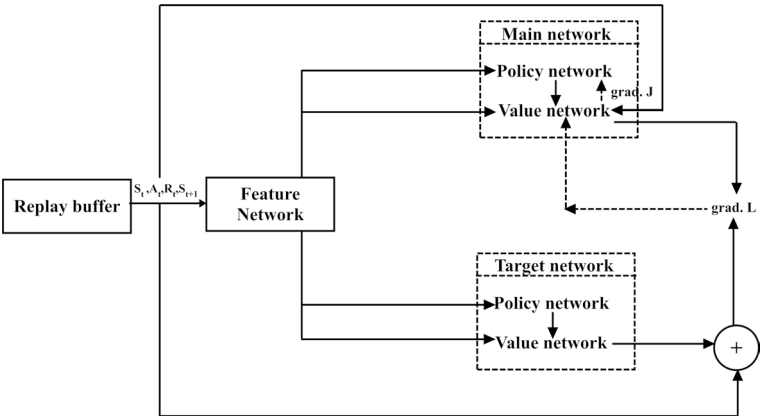

Fig. 7 Framework of DPG algorithm (Yin et al. 2019)

Basically, it is an iterative policy method, that involves two steps: (a) policy evaluation evaluates the value function related to fixed policy using IRL Bellman equation and system dynamics is not involved (b) policy is improved using the value obtained in policy evaluation step. A small exploratory probing noise is added to control input for sufficient exploration of the state space, which is important for proper convergence of optimal value function and this satisfies the persistently exciting (PE) qualitatively (Modares et al. 2015).

4. *Deterministic Policy Gradient (DPG) Algorithm* DPG algorithm comes under the framework of actor-critic methods. In article (Silver et al. 2014; Yin et al. 2019), authors have discussed DPG algorithms. When the action space in MDP is continuous then there is a required deterministic policy, i.e., $a = \pi_l(s)$ instead of stochastic policy. In other words, the learning model should map the continuous state to continuous action. DPG algorithms have two components: (a) Experience replay and (b) Model learning. The DPG algorithm framework is shown in Fig. 7. In the first component, training data are generated by taking actions based on a policy network whose parameters are randomly initialized. Collected data in every time-step is placed in the buffer and have formed of tuples: $< S_t, A_t, R_t, S_{t+1} >$. For diversity in experience, additional noise is mixed in a policy network. Thus, actions generated by the policy network are much more diverse. In the second portion, the model is trained using the collected data. Therefore, the main policy and value network is learned alternatively using stored data in replay buffer. Parameters of the main value network are updated using the gradient method, i.e., minimization of the loss function. The loss function is defined as (20),

$$L = \frac{1}{2M} \sum_{i=1}^{M} [T_i - Q(F(S_i), A_i)]^2 \tag{20}$$

Whereas, the main policy network is updated using a policy gradient given by (21),

$$\nabla_{l_\pi} J = \frac{1}{M} \sum_{i=1}^{M} \nabla_{l_\pi} \pi(F(S_i)) \nabla_{A_i} Q(F(S_i), A_i) \tag{21}$$

where $T_i = Q'(F(S_{i+1}), \pi'(F(S_{i+1})))$, F is feature network, Q and Q' are main and target value network respectively, π and π' represents main and target policy network respectively and l denotes network parameters. Here, parameters of the feature network are updated once either with the main or target network. Simultaneously, the target network is updated as a copy of the main network for some iterations.

5. *Parametrized Batch Actor-Critic (PBAC) Algorithm* PBAC algorithm in which the parametrized feature vectors are trained from sample data for approximating value function and policies. This algorithm has a same linear feature for both actor and critic structures compared to previous actor-critic methods. It is theoretically proved that the same feature can provide efficient learning and fast convergence (Huang et al. 2017). Least-square batch methods are used for better learning efficiency. Pseudo code for PBAC learning is given in algorithm 2.

Algorithm 2: Steps for PBAC learning algorithm

Initialization: Collect sample data using random or fixed policy
Procedure:
(a) Select sub-sample set from collected sample data, using approximately linear dependency
(b) Selected sub-sample is used for construction of kernel-based features that are used as basis for value function approximation
(c) Initialize the parameter x and y for critic and actor framework
while $t > t_{max}$ || $||y_{t+1} - y_t|| < \epsilon$ **do**
  Repeat for all states ;
  – Take action a based on policy (which is represented as gaussian disturbance whose center is based on actor's output
  $$\pi_t(s, y) = \frac{1}{\sqrt{2\pi\sigma}} e^{-(a - A_t(s))^2 / 2\sigma^2}$$
  Observe reward r_t and next state s_{t+1}
  – Calculate TD-error as,
  $$\delta_t(s_i) = r_i + \lambda V_t(s_{t+1}) - V_t(s_i)$$
  where, $V_t^{\pi}(s_i) = K^T(s_i) x_t$
  – Actor's target signal is,
  if $\delta_t(s_i) > 0$ **then**
    $a_t = \pi(s_i, y_t)$;
  else
    $A_t(s_i) = K^T(S_i) y_t$ i.e., actor's output is target;
  end
  – After end of states ;
  update the parameters of policy ;
end
(d) Return parameters
Result: Reurn the trained policy network

6. *Probabilistic Inference for Learning Control (PILCO)* Commonly, model-based learning methods are more promising compared to model-free methods like Q-learning and TD-learning. Because model-based methods can efficiently extract more information from available data. However, model-based methods suffer from model bias, i.e., here it is assumed that the learned model more appropriately resembles the real world. Especially, more problem occurs when few data are available and have no prior knowledge of tasks to be learned. In article (Deisenroth and Rasmussen 2011), the authors

proposed a probabilistic inference for learning control (PILCO) algorithm, which is a model-based policy search method. The framework has three parts: (a) dynamic learning model: in order to address the model uncertainty, probabilistic dynamic model (non-parametric Gaussian process) is used, (b) analytic approximate policy evaluation: deterministic approximate inference is used and (c) gradient-based policy improvement. The salient feature of this algorithm is that it achieves data efficiency even in the continuous domain and hence, it can directly be applied to robots.

7. *Policy Improvement with Path Integrals (PI^2) PI^2* is a probabilistic learning method and inherited the principles of stochastic optimal control from the path integral. Hamilton-Jacobi-Bellman equation is used to transform the policy improvements into the approximated path integral. This algorithm doesn't have an open tuning parameter, except exploration noise. In article (Theodorou et al. 2010), authors derives the generalized path integral optimal control problem as:

Optimal control

$$a_{t_i} = \int p(\tau_i) a_L(\tau_i) d\tau_i^{(c)} \tag{22}$$

Probability

$$P(\tau_i) = \frac{e^{-\frac{1}{\lambda}\hat{S}(\tau_i)d\tau_i^{(c)}}}{\int e^{-\frac{1}{\lambda}\hat{S}(\tau_i)d\tau_i^{(c)}}} \tag{23}$$

Local control

$$a_L(\tau_i) = R^{-1} G_{t_i}^{(c)^T} (G_{t_i}^{(c)} R^{-1} G_{t_i}^{(c)^T})(G_{t_i}^{(c)} \epsilon_{t_i} - b_{t_i}) \tag{24}$$

where superscript c is used to represent the actuated/controllable state. $G_{t_i}^{(c)}$ is control transition matrix, λ is regularization parameter. The path integral problem is connected with reinforcement learning as parametrized policies. Here, dynamic movement primitives (DMP) representation is used as generalized policies. It is beneficial to use this representation because by varying only one parameter the trajectory changes its shape. Initial and final (goal) states are fixed. The authors derived some important expressions are (For more detailing refer to this article (Theodorou et al. 2010)):

Probability

$$P(\tau_i) = \frac{e^{-\frac{1}{\lambda}S(\tau_i)d\tau_i^{(c)}}}{\int e^{-\frac{1}{\lambda}S(\tau_i)d\tau_i^{(c)}}} \tag{25}$$

Cost

$$S(\tau_i) = \phi_{t_K} + \sum_{j=1}^{K-1} q_{t_j} dt + \frac{1}{2} \sum_{j=1}^{K-1} (l + N_{t_j} \epsilon_{t_j})^T R (l + N_{t_j} \epsilon_{t_j}) \tag{26}$$

where $N_{t_j} = \frac{R^{-1} g_{t_j} g_{t_j}^T}{g_{t_j}^T R^{-1} g_{t_j}}$

Parameter update for every time step based on probability average

$$\delta l_{t_j} = \int P(\tau_i) N_{t_i} \epsilon_{t_i} d\tau_i \tag{27}$$

Final updation of parameter

$$[\delta l]_j = \frac{\sum_{i=0}^{K-1} (K-i) x_{j,t_i} [\delta l_{t_i}]_j}{\sum_{i=0}^{K-1} (K-i) x_{j,t_i}} \tag{28}$$

New value of parameter

$$l^{(new)} = l^{(old)} + \delta l \tag{29}$$

Given: immediate cost: $r_t = q_t + l_t{}^T R l_t$, terminal cost: ϕ_{t_K}, stochastic policy: $a_t = (g_t)^T (l + \epsilon_t)$, basis function: g_{t_i}, initial parameter l and variance in mean-zero noise: $\sum_\epsilon$. Initially after rollout, equation (25) is used to compute the probability at time t_i for each trajectory and cost is computed by (26). Then in every step, parameter update is calculated based on (27). Equation (28) is used as a final average of parameter updating. Then, change in parameter is added to the old value of the parameter given by (29).

Salient features of this algorithm are: (a) updated equation have no numerical instabilities (b) scalable to a higher dimension.

8. *Deep-Q network (DQN)* As the state and action become very large in number or continuous, memory required by Q-learning becomes inevitable. Thus, mnih et. al. in 2013 presented an integrated framework of deep learning and Q-learning that solves the sample correlation and memory limitation. In 2015 (Mnih et al. 2015b), authors proposed a framework of the double-network structure. Its salient features are: (a) deep convolutional neural network (CNN), state-action value Q(s,a) is represented by Q(s,a;l) and parametrized by l. It solves the memory limitation up-to some extent (b) experience replay is used to solve the correlation of samples (c) a separate target network is used to handle the TD targets. It estimates the TD target and state-action values. Thus, the weights l are updated by (30),

$$\begin{aligned} l_{t+1} = l_t + \alpha [r_t + \lambda * max_{a_{t+1} \epsilon A} Q(s_{t+1}, a_{t+1}; l^{TD}) \\ - Q(s_t, a_t; l_t)] * \nabla Q(s_t, a_t; l_t) \end{aligned} \tag{30}$$

9. *Weighted Behaviour Q-Learning (WBQL) Algorithm* In multi-robot formations, each robot must interact with each other and the environment based on information gain. In article (Lin et al. 2013), these capabilities are regarded as self-awareness behavior. Thus, the authors proposed an algorithm that is based on a self-awareness mechanism. Here, weights are used to mark individual behavior as relative importance. Q-learning is used to update the weights, i.e.,

$$Q(s,a) \leftarrow Q(s,a) + \gamma * [w(s,a) * r + \lambda * \sum_{k=1}^{n} (w(s',k) * Q(s',k)) - Q(s,a)] \tag{31}$$

10. *Policy learning by weighting exploration with the returns (PoWER) Algorithm* A deterministic mean policy is defined as: $\hat{a} = l^T \phi_{s,t}$. Parameter l and basis function ϕ for policy is derived from the motor primitive formulation. A stochastic policy is obtained by adding exploration $\epsilon(s,t)$ to deterministic policy. Thus, the obtained policy $\pi(a_t|s_t,t)$ is transformed as (32),

$$a = l^T \phi(s,t) + \epsilon(\phi(s,t)) \tag{32}$$

Here, gaussian exploration is used, i.e., $\epsilon(\phi(s,t)) \approx N(\epsilon|0, \sum)$. Without structure exploration, actions become random. Thus, it can create a strain or pressure on the joints of a robot, which can be dangerous to the robot. Thus, in article (Kober and Peters 2011), authors used structured form state-dependent exploration as (33),

$$\epsilon(\phi(s,t)) = \epsilon_t^T \phi_{(s,t)} \tag{33}$$

Thus, the resulted policy is defined by (34),

$$a \approx \pi(a_t|s_t,t) = N(a|l^T\phi(s,t), \phi(s,t)^T \hat{\sum} \phi(s,t)) \tag{34}$$

Hence, update rule for a parameter is derived using resulted policy and it is given by (35),

$$l' = l + E[\sum_{t=1}^{T} Z(s,t)Q^{\pi}(s,a,t)]^{-1} E[\sum_{t=1}^{T} Z(s,t)\epsilon_t Q^{\pi}(s,a,t)] \tag{35}$$

where $Z(s,t) = \phi(s,t)\phi(s,t)^T(\phi(s,t)^T \hat{\sum} \phi(s,t))^{-1}$. Here, importance sampling is used to reduce the overall-number of roll-outs.

11. *Covariance Matrix Adaptation Evolution Strategy (CMA-ES) Algorithm* In article (Hansen and Ostermeier 2001), the authors presented a self-adaption of mutation distribution based on cumulation and de-randomization. Here, the mutation strategy is adopted such that it favors the previous mutation in future selections. If this approach performed more rigorously, resulted in a de-randomized self-adaption scheme. This scheme is known as covariance matrix adaption (CMA). Its performance is improved by using cumulation. Complete framework results in the CMA-ES algorithm. It is applied for solving problems which are described as MDPs (Heidrich-Meisner and Igel 2008). It is showing its applicability on balancing tasks such as double cart-pole and it outperforms the policy gradient and stochastic search methods.
12. *Quantum Inspired RL (QiRL) Algorithm* Forming a policy for the selection of action is a complex task in robotic systems. In literature (Dong et al. 2010), ϵ-greedy, Boltzmann exploration, etc., is presented for action selection, but these methods have inherent exploration and exploitation dilemma. In article (Dong et al. 2010), quantum-inspired RL is proposed to address this issue. For a selection of probabilistic action and probability of "good" and "bad" action based on reward are adopted by quantum measurement and amplitude amplification (a phenomenon in quantum computing) respectively.

    Here, policy is defined as a mapping of states to actions as, $f(s) = \pi : S \rightarrowtail A$. The corresponding action is probabilistic in nature (based on the quantum phenomenon) and it is given by (36),

$$f(s) = \frac{|a_1>}{|\alpha_1|^2} + \frac{|a_2>}{|\alpha_1 2^2} + \cdots = \sum_n \frac{|a_n>}{|\alpha_n|^2} \tag{36}$$

    Equation (38) means, state s have the action set as $\{ |a_1>, \ldots |a_n>\}$, the agent will select the eigen-action $|a_n>$ with probability $|\alpha_n|^2$.
13. *Energy-Efficiency and Continuous-Movement Control (E^2CMC) Algorithm* In article Zeng et al. (2016), the authors proposed an E^2CMC algorithm for continuous

movement control of drones with energy efficiency. Here, information like drone-cell location, energy consumption, and coverage is transmitted to the cloud for central computing. The problem is set up as MDP. Thus, the observations by the agent are set of coverage fairness and energy consumption. Actions are set off moving distance, yaw angle, and a pitch angle of drone-cells. Likewise, the reward is designed based on the consumption of energy. The penalty for boundary limitation is also included. If the connection of the drone is lost, then the corresponding movement is canceled. Procedure for E^2CMC algorithm is:

- *Initialize* Parameters of critic evaluation network $Q(s, a|l^Q)$ and actor evaluation network $\mu(s, a|l^\mu)$, Parameters of critic target networks $Q(s, a|l^{Q'})$ and actor target network $\mu(s, a|l^{\mu'})$, buffer replay F and drone-cell graph R.
- *Perform roll-outs* select probabilistic action as $a_t = \mu(s_t|l^\mu) + N_t$ and get next state and reward.
- Store the transition of states (s_t, a_t, r_t, s_{t+1}) in replay buffer F.
- Update the parameters l^Q by minimizing the loss function and parameters l^μ using a gradient.

14. *Fuzzy Controller Design* A Control architecture is proposed by Hasegawa et al. (1999), mechanism maps a table that interrelates robot state variable to control inputs. This helps in the transfer of knowledge to form fuzzy rules. Block diagram of fuzzy-based RL is shown in Fig. 8. Salient Features of given topology are:

    (a) A RL algorithm generates a wide range of continuous real-value actions.
    (b) Reinforcement signal is self-scaled so it prevents parameters from overshooting when the system receives a large reinforcement value.

As per literature, some authors have proposed different topologies for tuning of fuzzy controller parameters (Dai et al. 2005). Given architecture has comprised of a Q estimator network (QEN) and Takagi–Sugeno-type fuzzy inference system (TSK-FIS). TD methods and gradient-descent algorithm are used for tuning of controller parameters.

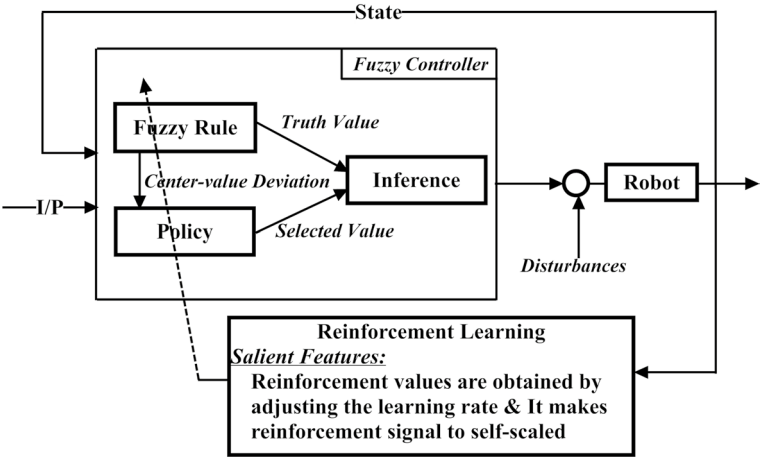

Fig. 8 Fuzzy controller (Hasegawa et al. 1999)

15. *Coordination Mechanism between Multi-agents* In article (Abul et al. 2000), independent coordination-mechanism based on RL is proposed for coordinating between multiple agents. Two coordinate mechanism is presented:

    (a) *Perceptual Co-ordinate Mechanism* Co-ordination information learned from environment.
    (b) *Observing Co-ordinate Mechanism* Rewards obtained by nearby agents are noted. Thus, noted reward from the environment and its own reward are used for restructuring of optimal policy.

    The salient feature of the mechanism is that the other agents present in the environment are also included in the state description. It helps in the increment of joint rewards in a multiagent system. An adversarial food-collecting world (AFCW) environment is used for showing the effectiveness of the coordination mechanism between agents for survival in hostile environments. AFCW environment is shown in Fig. 9, where, agents, enemy, food and obstacles are represented by letter **A**, **E**, **F** and **black blocks** respectively. Legal movement of agents and enemies are pre-defined in (Abul et al. 2000). Agents are said to be learned coordination if every agent tried to maximize total accumulated reward.
16. *Designing of Reward Function* The interaction of the agent with the environment generates a trajectory and data. A subset of data (I^C) is chosen and applied to the input of the controller, it generates the next action which is applied to the actuators. Likewise, another subset of data (I^R) is taken (this can be overlap with previous input I^C), and that is used for evaluation of the performance of the learning agent. RL algorithm is used for the updating of controller parameters. It leads to an overall improvement in the effectiveness of agent performance, herein RL is used to compute a reinforcement function. The primary target of function is to provide correct knowledge of learning tasks and behavior of agent to the learning algorithm. The basic aim of the learning system is to improve the performance of an overall system by maximizing the reward function. As per literature, the function is designed through a trial and error approach. However, some authors (Bonarini et al. 2001) provides a complete engineering per-

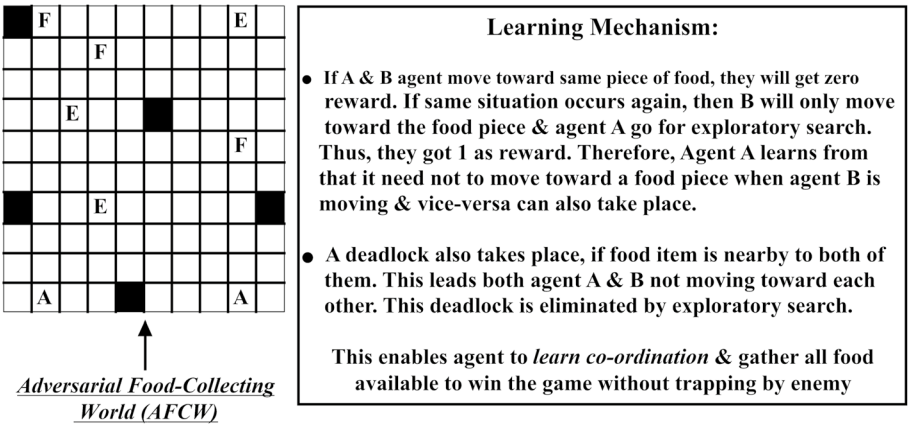

Fig. 9 Multi-agent coordination

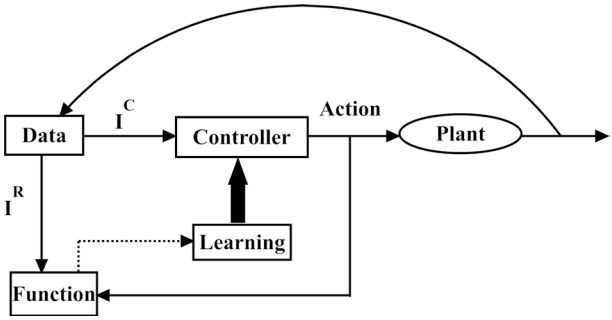

Fig. 10 Learning from reward function

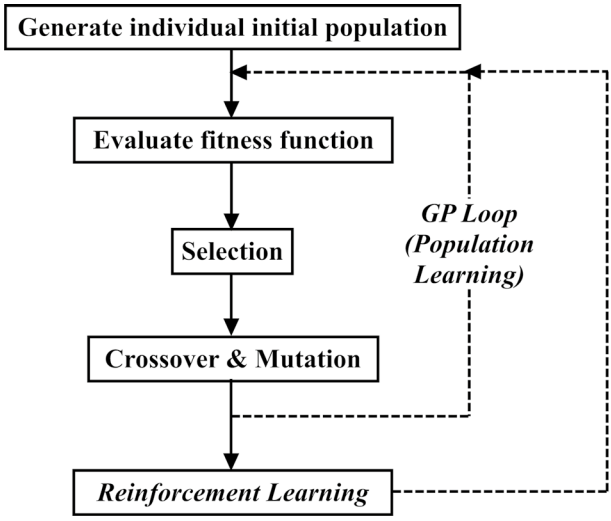

Fig. 11 Framechart of integration of GP with RL (Kamio and Iba 2005)

spective for designing the reward function. Which is used to generate the control input for complex tasks, such as robotic, autonomous systems, etc. A complete framework of learning with reward function is shown in Fig. 10.

17. *Integration of Meta-heuristic with RL* Integration of genetic programming with RL is proposed for learning of effective actions for adaption of a robot to real environment (Kamio and Iba 2005). The complete frame-chart is shown in Fig. 11. The frame-chart shows that RL and genetic programming are feedback from outside and inner loop respectively. It speedup the convergence speed and adaption of a real robot to a stochastic environment simultaneously. In literature, some authors also proposed an ant optimizer based fuzzy controller, and rules are learned from RL.
18. *Gait Synthesis of walking patterns & Sensory Control*Stair-climbing gait and sensor control strategy are considered for feed-forward control and feedback control (parameter of the controller is adjusted by RL) respectively for control of humanoid robots (Fu and Chen 2008).
19. *Impedance Learning by Actor-Critic Algorithm* Modified learning strategy based on control theory and reinforcement learning for skill the motor is presented in (Kim et al. 2009). In proposing an impedance controller, parameters are modified using actor-critic based RL. Basically, in a proposed mechanism, two structures are developed,

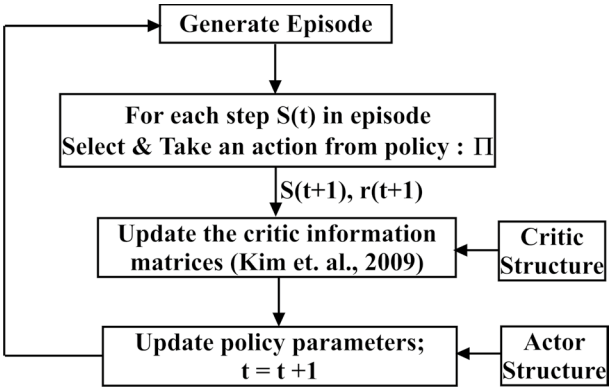

Fig. 12 Framework of Impedance Learning (Kim et al. 2009)

first one is the actor structure that determines the policy for the selection of action and the second one is the critic structure that evaluates whether the selected action is good or bad. A framework of the proposed mechanism is presented in Fig. 12.

20. *Deep neural-network trained using RL* Zero-bias and zero-variance in samples can lead to learning high-performance policy. In (Hwangbo et al. 2017), a framework is proposed for control of quad-rotor based on a deep neural network which is trained using RL. The Deep neural network has two hidden layers. Each layer has 64 nodes with tanh function. Parameters of the neural network are tuned using the natural gradient descent algorithm. Monte-Carlo samples are used for the estimation of the value function.
21. *Information-gain mechanism based on DeepRL* DeepRL is used to coordinate between aerial robots to either mapping or gathering information about unknown terrain. The key challenge is that robots need to perceive their current states through sensors that are imperfect in nature. In order to address this issue, the authors have proposed a framework as shown in Fig. 13 (Viseras and Garcia 2019).

5 Outlook

In this paper, a lot of different mechanisms proposed by different authors and their application in robotic applications are surveyed. Sometimes, it is very difficult to find the appropriate method that results in a better control policy for problem-solving. However, we can use intuition for the selection of methods in advance from an actor, critic, and actor-critic techniques. If a type of control policy is learned for continuous space, then the critic method is not appropriate. Whereas, if policy learned for finite and countable space, choose the critic method. Although the choice between actor-critic and actor methods is simple. If the problem is presented as stationary MDPs then actor-critic methods should be used. Because it provides a lower-variance. However, actor methods are more robust in a non-stationary environment. In these techniques, the choice of better parameterization for a policy is an important question because it highly influences the performance of learning. Dimensionality is also a significant concern. As the number of the state increases, learning of the agent becomes computationally difficult. Thus, in literature, many authors have tried to reduce the complexity of dimension (O'Flaherty and Egerstedt 2014; Polat et al. 2002; Colomé

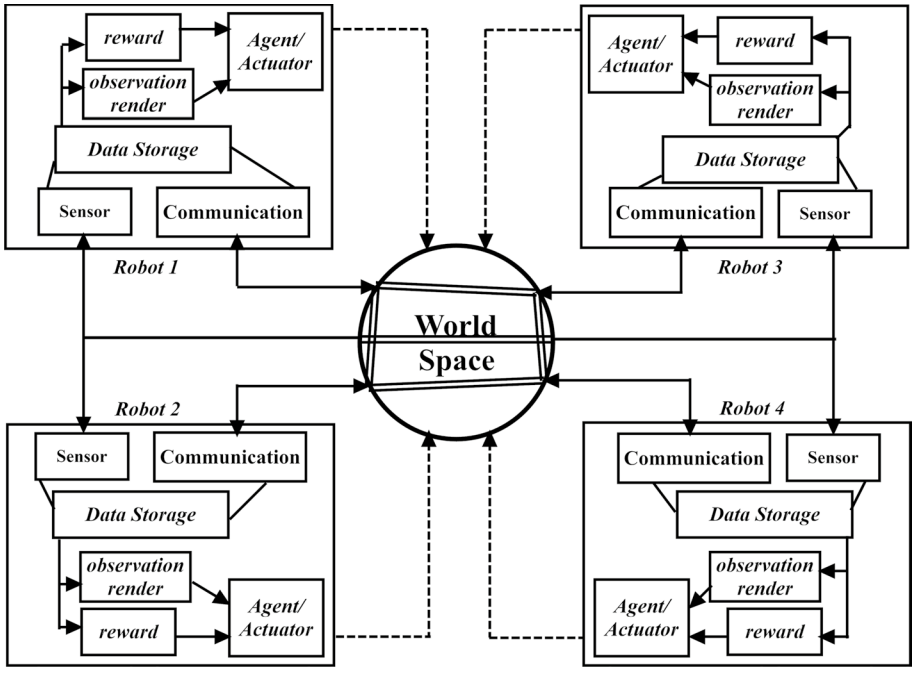

Fig. 13 Shematic View of DeepRL in multi-aerial Robot (Viseras and Garcia 2019)

and Torras 2018; Lasheng et al. 2012). In some cases, the authors have presented an algorithm that uses a hybridization of a continuous and discrete form of the complex physical robotic system. These are based on an abstraction of learning boundary-action pairs that mitigate the curse of dimensionality.

In traditional RL, the estimate of the Q-value function is used for designing policy and action taken in a particular state by the agent. Due to the uncertainty present in the estimated Q-value, the agent can trap inside a local optimum. It may occur due to insufficient exploration. Hence, the dilemma between exploration and exploitation is present. Therefore, in article (Iwata et al. 2004), authors have proposed a new criterion based on a ratio of return loss to information gain. This forms a strategy of action taken in a particular state. Purely exploitation and exploration leads to an optimal local solution quickly and degrades the overall performance of learning respectively. Therefore, the goal of the algorithm is to find an optimal policy without trapping in local optima. It can be achieved by transforming the goal into a search problem. In article (Guo et al. 2004), metropolis criterion is used in simulated annealing (SA) algorithm. It is applied to a search procedure in order to find a balance between exploration and exploitation. Hence, integration of SA with Q-learning resulted in a balance between exploration and exploitation. Exploration will decay gradually when the solution is approaching optimum value. It can be applied in robotics for balancing exploration and exploitation as a future scope.

Automatic learning for robots is done by the data-driven controller using RL, but the significant drawback is that agent requires many interactions with the environment for learning of the controller parameters. In order to reduce the required number of interactions before reaching an optimal solution, a domain expert is needed to reshape the policies, it may yield a reduction in the computational cost requirement. However, it may be difficult

to find the appropriate domain expert. And it may also be possible that the dynamics of the model are not known completely. Therefore, the control of the robot becomes very difficult. So, it is advisable to learn the model through probabilistic modeling, non-parametric Gaussian process transition method and go for multi-fidelity simulators (Deisenroth et al. 2013a; Cutler et al. 2015). In most cases, the robot uses a visual feedback signal but the presence of disturbances (like signal interference or dust) makes it difficult for controlling the robot. Thus, in literature authors have proposed a mechanism for effective control of the robot (Li et al. 2017b; Sharma et al. 2018; Zhang et al. 2019).

Another interesting area is to develop new approaches based on DeepRL for unstructured data. It can help in the control of humanoid robots. DeepRL for feedback control requires new approaches such that the stability can be assured in the sense of Lyapunov. As Bayesian optimization is a global search method, though till now it does not well scale up to large parameter policy space and applicable only to low dimensional. Thus, there is further need for an investigation that how to use Bayesian optimization to apply for high dimensional parameter space like deep neural networks.

The robot can autonomously learn from demonstrations, instruction, and advice. Here, human feedback can reduce the number of trials because it can help in the reduction of total incorrect actions. However, as a future scope, a methodology that includes the demonstrations and human feedback, or an integrated approach of advice and feedbacks needs to investigate. And also it is needed for further investigation of agent learning from facial expression as well. The agent can learn either implicitly or explicitly from human feedback. In the future, humans and robots will interact with each other frequently. However, learning from human feedback is a very complex task as compared to traditional RL because it is based on environmental feedback. Therefore, there is a need of understanding human behavior before developing a new topology so that agents can effectively learn from feedback. Thus, a proper multi-coordination and human–robot ecosystem need to develop.

RL could be effective in real-robot like human-level abilities. If the robot's cognitive structure is suitably pre-programmed with a decomposition of a task into easy learned behaviors or skills. There can be a possibility of a new learning policy. Further, the learning process can be speedup by the reuse of data and experience replay.

Mostly, RL is applicable to sequential decision-making problems. Likewise for multi-task problems integrated Q-learning and knowledge transfer yields a better control policy. But, it has significant drawbacks of long-time exploration. Therefore, integration of Deep learning with RL for the multi-task problem can provide better control policy in less time. And also there is a need for a new policy framework to develop more advanced autonomous robots. Like in recent times, Neuro-evolution is emerging as a powerful tool since computation power increased. It improves the capabilities of a neural network that includes learning of hyper-parameters, architecture, rules of learning and activation functions, etc. Therefore, researchers found that neuro-evolution is more competitive than gradient-based methods to train a neural network for RL problems. When the neuro-evolution technique combined with DeepRL could help in progress towards the development of artificial general intelligence (AGI) framework. This framework when applied to the robotic application, it leads to more advanced intelligence and that can be the future research direction.

Biological and artificial systems need to achieve goals for their survival. Neuromorphic is an engineering field that makes a bridge between device physics and behavior inspired by the brain's building blocks, such as spiking neural networks. A recent development in neuromorphic hardware yields a platform for brain-inspired learning algorithms. Likewise, biological retinas-based neuromorphic vision sensors already changing the landscape of the computer vision field. It has a greater impact on robotics as well. Thus, the focus

should be more on the deeper insights of the brain. In the near future, it can lead to the development of RL based artificial agents that can solve complex real-world problems.

6 Conclusion

We have discussed the motivation of reinforcement learning, where most of the learning problems are complex due to the stochastic environment. In this paper, we have provided a comprehensive survey on the application of RL algorithms in robotics, which includes land, water, and air-based applications. Firstly, we have introduced a detailed explanation of various RL mechanisms such as Actor-critic, Human-centered, DeepRL, multi-agent, etc. Then briefly explain the various application of RL. Then, discussed various methodology/algorithms developed by authors in literature. In the latter part, we have expressed our view on the challenges faced by RL algorithms with different applications and also discuss future scope. Differential topology like neuro-evolution, spiking neural networks, modification in cognitive architecture, etc., can be used to improve the learning of robots. Neuro-evolution is a methodology in which the architecture of the artificial neural networks has evolved with time. Thus, as future scope, integration of neuro-evolution with RL algorithm can provide a better optimal policy for robots. Moving forward from robotic application based on RL, surveyed learning mechanisms in this paper is also well suited for application in larger prospective fields such as power system, communication, aerospace, etc., where RL can make system autonomous.

References

Abul O, Polat F, Alhajj R (2000) Multiagent reinforcement learning using function approximation. IEEE Trans Syst Man Cybern Part C (Appl Rev) 30(4):485–497

Adam S, Busoniu L, Babuska R (2011) Experience replay for real-time reinforcement learning control. IEEE Trans Syst Man Cybern Part C (Appl Rev) 42(2):201–212

Ansari Y, Manti M, Falotico E, Cianchetti M, Laschi C (2017) Multiobjective optimization for stiffness and position control in a soft robot arm module. IEEE Robot Autom Lett 3(1):108–115

Antonelo EA, Schrauwen B (2014) On learning navigation behaviors for small mobile robots with reservoir computing architectures. IEEE Trans Neural Netw Learn Syst 26(4):763–780

Arulkumaran K, Deisenroth MP, Brundage M, Bharath AA (2017) Deep reinforcement learning: a brief survey. IEEE Signal Process Mag 34(6):26–38

Averbeck BB, Costa VD (2017) Motivational neural circuits underlying reinforcement learning. Nat Neurosci 20(4):505–512

Baird L (1995) Residual algorithms: reinforcement learning with function approximation. In: Machine learning proceedings 1995, Elsevier, pp 30–37

Baird III LC, Moore AW (1999) Gradient descent for general reinforcement learning. In: Advances in neural information processing systems, pp 968–974

Barto AG, Sutton RS, Anderson CW (1983) Neuron like adaptive elements that can solve difficult learning control problems. IEEE Trans Syst Man Cybern 5:834–846

Bejar E, Moran A (2019) A preview neuro-fuzzy controller based on deep reinforcement learning for backing up a truck-trailer vehicle. In: 2019 IEEE canadian conference of electrical and computer engineering (CCECE), IEEE, pp 1–4

Beom HR, Cho HS (1995) A sensor-based navigation for a mobile robot using fuzzy logic and reinforcement learning. IEEE Trans Syst Man Cybern 25(3):464–477

Bertsekas DP (1995) Dynamic programming and optimal control. Athena scientific, Belmont

Bertsekas DP (2018) Feature-based aggregation and deep reinforcement learning: a survey and some new implementations. IEEE/CAA J Autom Sin 6(1):1–31

Böhmer W, Springenberg JT, Boedecker J, Riedmiller M, Obermayer K (2015) Autonomous learning of state representations for control: an emerging field aims to autonomously learn state representations for reinforcement learning agents from their real-world sensor observations. KI-Künstliche Intelligenz 29(4):353–362
Bonarini A, Bonacina C, Matteucci M (2001) An approach to the design of reinforcement functions in real world, agent-based applications. IEEE Trans Syst Man Cybern Part B (Cybern) 31(3):288–301
Bowling M, Veloso M (2001) Rational and convergent learning in stochastic games. Int Joint Conf Artif Intell 17:1021–1026
Bowling M, Veloso M (2002) Multiagent learning using a variable learning rate. Artif Intell 136(2):215–250
Boyan JA (2002) Technical update: least-squares temporal difference learning. Mach Learn 49(2–3):233–246
Bradtke SJ, Ydstie BE, Barto AG (1994) Adaptive linear quadratic control using policy iteration. In: Proceedings of 1994 American control conference-ACC'94, IEEE, vol 3, pp 3475–3479
Breyer M, Furrer F, Novkovic T, Siegwart R, Nieto J (2019) Comparing task simplifications to learn closed-loop object picking using deep reinforcement learning. IEEE Robot Autom Lett 4(2):1549–1556
Bu L, Babu R, De Schutter B et al (2008) A comprehensive survey of multiagent reinforcement learning. IEEE Trans Syst Man Cybern Part C (Appl Rev) 38(2):156–172
Caarls W, Schuitema E (2015) Parallel online temporal difference learning for motor control. IEEE Trans Neural Netw Learn Syst 27(7):1457–1468
Cao X, Sun C, Yan M (2019) Target search control of auv in underwater environment with deep reinforcement learning. IEEE Access 7:96549–96559
Carlucho I, De Paula M, Villar SA, Acosta GG (2017) Incremental q-learning strategy for adaptive pid control of mobile robots. Expert Syst Appl 80:183–199
Carlucho I, De Paula M, Wang S, Petillot Y, Acosta GG (2018) Adaptive low-level control of autonomous underwater vehicles using deep reinforcement learning. Robot Auton Syst 107:71–86
Chalvatzaki G, Papageorgiou XS, Maragos P, Tzafestas CS (2019) Learn to adapt to human walking: a model-based reinforcement learning approach for a robotic assistant rollator. IEEE Robot Autom Lett 4(4):3774–3781
Cheng Y, Zhang W (2018) Concise deep reinforcement learning obstacle avoidance for underactuated unmanned marine vessels. Neurocomputing 272:63–73
Colomé A, Torras C (2018) Dimensionality reduction for dynamic movement primitives and application to bimanual manipulation of clothes. IEEE Trans Robot 34(3):602–615
Cruz F, Magg S, Weber C, Wermter S (2016) Training agents with interactive reinforcement learning and contextual affordances. IEEE Trans Cogn Dev Syst 8(4):271–284
Cutler M, Walsh TJ, How JP (2015) Real-world reinforcement learning via multifidelity simulators. IEEE Trans Robot 31(3):655–671
Da Silva B, Konidaris G, Barto A (2012) Learning parameterized skills. Preprint arXiv:12066398
Dai X, Li CK, Rad AB (2005) An approach to tune fuzzy controllers based on reinforcement learning for autonomous vehicle control. IEEE Trans Intell Transp Syst 6(3):285–293
Dayan P, Niv Y (2008) Reinforcement learning: the good, the bad and the ugly. Curr Opin Neurobiol 18(2):185–196
de Bruin T, Kober J, Tuyls K, Babuška R (2018) Integrating state representation learning into deep reinforcement learning. IEEE Robot Autom Lett 3(3):1394–1401
Deisenroth M, Rasmussen CE (2011) Pilco: a model-based and data-efficient approach to policy search. In: Proceedings of the 28th international conference on machine learning (ICML-11), pp 465–472
Deisenroth MP, Fox D, Rasmussen CE (2013a) Gaussian processes for data-efficient learning in robotics and control. IEEE Trans Pattern Anal Mach Intell 37(2):408–423
Deisenroth MP, Neumann G, Peters J et al (2013b) A survey on policy search for robotics. Found Trends Robot 2(1–2):1–142
Deng Z, Guan H, Huang R, Liang H, Zhang L, Zhang J (2017) Combining model-based q-learning with structural knowledge transfer for robot skill learning. IEEE Trans Cogn Dev Syst 11(1):26–35
Dong D, Chen C, Chu J, Tarn TJ (2010) Robust quantum-inspired reinforcement learning for robot navigation. IEEE/ASME Trans Mech 17(1):86–97
Doroodgar B, Liu Y, Nejat G (2014) A learning-based semi-autonomous controller for robotic exploration of unknown disaster scenes while searching for victims. IEEE Trans Cybern 44(12):2719–2732
Doshi-Velez F, Pfau D, Wood F, Roy N (2013) Bayesian nonparametric methods for partially-observable reinforcement learning. IEEE Trans Pattern Anal Mach Intell 37(2):394–407
Duan Y, Cui BX, Xu XH (2012) A multi-agent reinforcement learning approach to robot soccer. Artif Intell Rev 38(3):193–211

El-Fakdi A, Carreras M (2013) Two-step gradient-based reinforcement learning for underwater robotics behavior learning. Robot Auton Syst 61(3):271–282
Er MJ, Deng C (2005) Obstacle avoidance of a mobile robot using hybrid learning approach. IEEE Trans Ind Electron 52(3):898–905
Falco P, Attawia A, Saveriano M, Lee D (2018) On policy learning robust to irreversible events: an application to robotic in-hand manipulation. IEEE Robot Autom Lett 3(3):1482–1489
Farahmand AM, Ahmadabadi MN, Lucas C, Araabi BN (2009) Interaction of culture-based learning and cooperative co-evolution and its application to automatic behavior-based system design. IEEE Trans Evol Comput 14(1):23–57
Faust A, Ruymgaart P, Salman M, Fierro R, Tapia L (2014) Continuous action reinforcement learning for control-affine systems with unknown dynamics. IEEE/CAA J Autom Sin 1(3):323–336
Foglino F, Christakou CC, Leonetti M (2019) An optimization framework for task sequencing in curriculum learning. In: 2019 Joint IEEE 9th international conference on development and learning and epigenetic robotics (ICDL-EpiRob), IEEE, pp 207–214
Frost G, Maurelli F, Lane DM (2015) Reinforcement learning in a behaviour-based control architecture for marine archaeology. In: OCEANS 2015-Genova, IEEE, pp 1–5
Fu C, Chen K (2008) Gait synthesis and sensory control of stair climbing for a humanoid robot. IEEE Trans Ind Electron 55(5):2111–2120
Gordon GJ (1995) Stable function approximation in dynamic programming. In: Machine learning proceedings 1995, Elsevier, pp 261–268
Gosavi A (2009) Reinforcement learning: a tutorial survey and recent advances. INFORMS J Comput 21(2):178–192
Gottipati SK, Seo K, Bhatt D, Mai V, Murthy K, Paull L (2019) Deep active localization. IEEE Robot Autom Lett 4(4):4394–4401
Greenwald A, Hall K, Serrano R (2003) Correlated q-learning. ICML 3:242–249
Grigorescu S, Trasnea B, Marina L, Vasilcoi A, Cocias T (2019) Neurotrajectory: a neuroevolutionary approach to local state trajectory learning for autonomous vehicles. Preprint arXiv:190610971
Grondman I, Busoniu L, Lopes GA, Babuska R (2012) A survey of actor-critic reinforcement learning: standard and natural policy gradients. IEEE Trans Syst Man Cybern Part C (Appl Rev) 42(6):1291–1307
Gu D, Hu H (2007) Integration of coordination architecture and behavior fuzzy learning in quadruped walking robots. IEEE Trans Syst Man Cybern Part C (Appl Rev) 37(4):670–681
Gullapalli V (1990) A stochastic reinforcement learning algorithm for learning real-valued functions. Neural Netw 3(6):671–692
Guo M, Liu Y, Malec J (2004) A new q-learning algorithm based on the metropolis criterion. IEEE Trans Syst Man Cybern Part B (Cybern) 34(5):2140–2143
Hansen N, Ostermeier A (2001) Completely derandomized self-adaptation in evolution strategies. Evol Comput 9(2):159–195
Hasegawa Y, Fukuda T, Shimojima K (1999) Self-scaling reinforcement learning for fuzzy logic controller-applications to motion control of two-link brachiation robot. IEEE Trans Ind Electron 46(6):1123–1131
Hazara M, Kyrki V (2019) Transferring generalizable motor primitives from simulation to real world. IEEE Robot Autom Lett 4(2):2172–2179
He W, Li Z, Chen CP (2017) A survey of human-centered intelligent robots: issues and challenges. IEEE/CAA J Autom Sin 4(4):602–609
Heidrich-Meisner V, Igel C (2008) Evolution strategies for direct policy search. In: International conference on parallel problem solving from nature, Springer, pp 428–437
Ho MK, Littman ML, Cushman F, Austerweil JL (2015) Teaching with rewards and punishments: Reinforcement or communication? In: CogSci
Hu H, Song S, Chen CP (2019) Plume tracing via model-free reinforcement learning method. IEEE Trans Neural Netw Learn Syst
Hu J, Wellman MP (2003) Nash q-learning for general-sum stochastic games. J Mach Learn Res 4:1039–1069
Hu J, Zhang H, Song L (2018) Reinforcement learning for decentralized trajectory design in cellular uav networks with sense-and-send protocol. IEEE Internet of Things Journal
Huang R, Cheng H, Qiu J, Zhang J (2019) Learning physical human–robot interaction with coupled cooperative primitives for a lower exoskeleton. IEEE Trans Autom Scie Eng
Huang Z, Xu X, He H, Tan J, Sun Z (2017) Parameterized batch reinforcement learning for longitudinal control of autonomous land vehicles. IEEE Trans Syst Man Cybern Syst 49(4):730–741

Hung SM, Givigi SN (2016) A q-learning approach to flocking with uavs in a stochastic environment. IEEE Trans Cybern 47(1):186–197
Hwang KS, Lo CY, Liu WL (2009) A modular agent architecture for an autonomous robot. IEEE Trans Instrum Meas 58(8):2797–2806
Hwang KS, Lin JL, Yeh KH (2015) Learning to adjust and refine gait patterns for a biped robot. IEEE Trans Syst Man Cybern Syst 45(12):1481–1490
Hwangbo J, Sa I, Siegwart R, Hutter M (2017) Control of a quadrotor with reinforcement learning. IEEE Robot Autom Lett 2(4):2096–2103
Iwata K, Ikeda K, Sakai H (2004) A new criterion using information gain for action selection strategy in reinforcement learning. IEEE Trans Neural Netw 15(4):792–799
Juang CF, Hsu CH (2009) Reinforcement ant optimized fuzzy controller for mobile-robot wall-following control. IEEE Trans Ind Electron 56(10):3931–3940
Kaelbling LP, Littman ML, Moore AW (1996) Reinforcement learning: a survey. J Artif Intell Res 4:237–285
Kamio S, Iba H (2005) Adaptation technique for integrating genetic programming and reinforcement learning for real robots. IEEE Trans Evol Comput 9(3):318–333
Khamassi M, Velentzas G, Tsitsimis T, Tzafestas C (2018) Robot fast adaptation to changes in human engagement during simulated dynamic social interaction with active exploration in parameterized reinforcement learning. IEEE Trans Cogn Dev Syst 10(4):881–893
Kim B, Park J, Park S, Kang S (2009) Impedance learning for robotic contact tasks using natural actor-critic algorithm. IEEE Trans Syst Man Cybern Part B (Cybern) 40(2):433–443
Kiumarsi B, Vamvoudakis KG, Modares H, Lewis FL (2017) Optimal and autonomous control using reinforcement learning: a survey. IEEE Trans Neural Netw Learn Syst 29(6):2042–2062
Kober J, Peters J (2011) Policy search for motor primitives in robotics. Mach Learn 84:171–203
Kober J, Bagnell JA, Peters J (2013) Reinforcement learning in robotics: a survey. Int J Robot Res 32(11):1238–1274
Koç O, Peters J (2019) Learning to serve: an experimental study for a new learning from demonstrations framework. IEEE Robot Autom Lett 4(2):1784–1791
Konda VR, Tsitsiklis JN (2000) Actor-critic algorithms. In: Advances in neural information processing systems, pp 1008–1014
Koryakovskiy I, Kudruss M, Vallery H, Babuška R, Caarls W (2018) Model-plant mismatch compensation using reinforcement learning. IEEE Robot Autom Lett 3(3):2471–2477
La HM, Lim R, Sheng W (2014) Multirobot cooperative learning for predator avoidance. IEEE Trans Control Syst Technol 23(1):52–63
Lambert NO, Drew DS, Yaconelli J, Levine S, Calandra R, Pister KS (2019) Low-level control of a quadrotor with deep model-based reinforcement learning. IEEE Robot Autom Lett 4(4):4224–4230
Lasheng Y, Zhongbin J, Kang L (2012) Research on task decomposition and state abstraction in reinforcement learning. Artif Intell Rev 38(2):119–127
Le TP, Ngo VA, Jaramillo PM, Chung T (2019) Importance sampling policy gradient algorithms in reproducing kernel hilbert space. Artif Intell Rev 52(3):2039–2059
Li G, Gomez R, Nakamura K, He B (2019) Human-centered reinforcement learning: a survey. IEEE Trans Hum Mach Syst
Li THS, Su YT, Lai SW, Hu JJ (2010) Walking motion generation, synthesis, and control for biped robot by using pgrl, lpi, and fuzzy logic. IEEE Trans Syst Man Cybern Part B (Cybern) 41(3):736–748
Li Z, Liu J, Huang Z, Peng Y, Pu H, Ding L (2017a) Adaptive impedance control of human-robot cooperation using reinforcement learning. IEEE Trans Ind Electron 64(10):8013–8022
Li Z, Zhao T, Chen F, Hu Y, Su CY, Fukuda T (2017b) Reinforcement learning of manipulation and grasping using dynamical movement primitives for a humanoidlike mobile manipulator. IEEE/ASME Trans Mech 23(1):121–131
Lin JL, Hwang KS, Wang YL (2013) A simple scheme for formation control based on weighted behavior learning. IEEE Trans Neural Netw Learn Syst 25(6):1033–1044
Lin Y, Makedon F, Xu Y (2011) Episodic task learning in markov decision processes. Artif Intell Rev 36(2):87–98
Littman ML (2015) Reinforcement learning improves behaviour from evaluative feedback. Nature 521(7553):445–451
Liu S, Ngiam KY, Feng M (2019) Deep reinforcement learning for clinical decision support: a brief survey. Preprint arXiv:190709475
Luo B, Liu D, Huang T, Liu J (2017) Output tracking control based on adaptive dynamic programming with multistep policy evaluation. IEEE Trans Syst Man Cybern Syst

Luo B, Yang Y, Liu D (2018) Adaptive q-learning for data-based optimal output regulation with experience replay. IEEE Trans Cybern 48(12):3337–3348
Luo B, Yang Y, Liu D, Wu HN (2019) Event-triggered optimal control with performance guarantees using adaptive dynamic programming. IEEE Trans Neural Netw Learn Syst 31(1):76–88
Lv L, Zhang S, Ding D, Wang Y (2019) Path planning via an improved dqn-based learning policy. IEEE Access
Madden MG, Howley T (2004) Transfer of experience between reinforcement learning environments with progressive difficulty. Artif Intell Rev 21(3–4):375–398
Markova VD, Shopov VK (2019) Knowledge transfer in reinforcement learning agent. In: 2019 international conference on information technologies (InfoTech), IEEE, pp 1–4
McPartland M, Gallagher M (2010) Reinforcement learning in first person shooter games. IEEE Trans Comput Intell AI Games 3(1):43–56
Meeden LA (1996) An incremental approach to developing intelligent neural network controllers for robots. IEEE Trans Syst Man Cybern Part B (Cybern) 26(3):474–485
Melo FS, Meyn SP, Ribeiro MI (2008) An analysis of reinforcement learning with function approximation. In: Proceedings of the 25th international conference on Machine learning, ACM, pp 664–671
Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G et al (2015a) Human-level control through deep reinforcement learning. Nature 518(7540):529
Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G et al (2015b) Human-level control through deep reinforcement learning. Nature 518(7540):529–533
Modares H, Ranatunga I, Lewis FL, Popa DO (2015) Optimized assistive human–robot interaction using reinforcement learning. IEEE Trans Cybern 46(3):655–667
Modares H, Lewis FL, Kang W, Davoudi A (2017) Optimal synchronization of heterogeneous nonlinear systems with unknown dynamics. IEEE Trans Autom Control 63(1):117–131
Muelling K, Kober J, Peters J (2010) Learning table tennis with a mixture of motor primitives. In: 2010 10th IEEE-RAS international conference on humanoid robots, IEEE, pp 411–416
Neftci E, Averbeck B (2019) Reinforcement learning in artificial and biological systems. Nat Mach Intell. https://doi.org/10.1038/s42256-019-0025-4
Neftci EO, Averbeck BB (2002) Reinforcement learning in artificial and biological systems. Environment p 3
Ng AY, Harada D, Russell S (1999) Policy invariance under reward transformations: theory and application to reward shaping. ICML 99:278–287
Nguyen ND, Nguyen T, Nahavandi S (2017) System design perspective for human-level agents using deep reinforcement le arning: a survey. IEEE Access 5:27091–27102
Nguyen TT, Nguyen ND, Nahavandi S (2018) Deep reinforcement learning for multi-agent systems: a review of challenges, solutions and applications. Preprint arXiv:181211794
O'Flaherty R, Egerstedt M (2014) Low-dimensional learning for complex robots. IEEE Trans Autom Sci Eng 12(1):19–27
Ohnishi M, Wang L, Notomista G, Egerstedt M (2019) Barrier-certified adaptive reinforcement learning with applications to brushbot navigation. IEEE Trans Robot 35(5):1186–1205
Palomeras N, El-Fakdi A, Carreras M, Ridao P (2012) Cola2: a control architecture for auvs. IEEE J Ocean Eng 37(4):695–716
Parunak HVD (1999) Industrial and practical applications of dai. Multiagent systems: a modern approach to distributed artificial intelligence pp 337–421
Peters J, Schaal S (2008) Reinforcement learning of motor skills with policy gradients. Neural netw 21(4):682–697
Peters J, Vijayakumar S, Schaal S (2005) Natural actor-critic. In: European conference on machine learning, Springer, pp 280–291
Peters J, Mulling K, Altun Y (2010) Relative entropy policy search. In: Twenty-Fourth AAAI conference on artificial intelligence
Plaza MG, Martínez-Marín T, Prieto SS, Luna DM (2009) Integration of cell-mapping and reinforcement-learning techniques for motion planning of car-like robots. IEEE Trans Instrum Meas 58(9):3094–3103
Polat F et al (2002) Learning intelligent behavior in a non-stationary and partially observable environment. Artif Intell Rev 18(2):97–115
Puterman ML (2014) Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons

Rescorla R, Wagner A, Black AH, Prokasy WF (1972) Classical conditioning ii: current research and theory pp 64–99
Ribeiro C (2002) Reinforcement learning agents. Artif Intell Rev 17(3):223–250
Riedmiller M, Peters J, Schaal S (2007) Evaluation of policy gradient methods and variants on the cart-pole benchmark. In: 2007 IEEE international symposium on approximate dynamic programming and reinforcement learning, IEEE, pp 254–261
Rombokas E, Malhotra M, Theodorou EA, Todorov E, Matsuoka Y (2012) Reinforcement learning and synergistic control of the act hand. IEEE/ASME Trans Mech 18(2):569–577
Roveda L, Pallucca G, Pedrocchi N, Braghin F, Tosatti LM (2017) Iterative learning procedure with reinforcement for high-accuracy force tracking in robotized tasks. IEEE Trans Ind Inform 14(4):1753–1763
Rummery GA, Niranjan M (1994) On-line Q-learning using connectionist systems, vol 37. University of Cambridge, Department of Engineering Cambridge, England
Rylatt M, Czarnecki C, Routen T (1998) Connectionist learning in behaviour-based mobile robots: a survey. Artif Intell Rev 12(6):445–468
Sallab AE, Abdou M, Perot E, Yogamani S (2017) Deep reinforcement learning framework for autonomous driving. Electron Imaging 19:70–76
dos Santos SRB, Givigi SN, Nascimento CL (2015) Autonomous construction of multiple structures using learning automata: description and experimental validation. IEEE Syst J 9(4):1376–1387
Santucci VG, Baldassarre G, Cartoni E (2019) Autonomous reinforcement learning of multiple interrelated tasks. Preprint arXiv:190601374
Schaul T, Horgan D, Gregor K, Silver D (2015) Universal value function approximators. In: International conference on machine learning, pp 1312–1320
Sharma R, Gopal M (2008) A markov game-adaptive fuzzy controller for robot manipulators. IEEE Trans Fuzzy Syst 16(1):171–186
Sharma RS, Nair RR, Agrawal P, Behera L, Subramanian VK (2018) Robust hybrid visual servoing using reinforcement learning and finite-time adaptive fosmc. IEEE Syst J
Shi H, Li X, Hwang KS, Pan W, Xu G (2016) Decoupled visual servoing with fuzzyq-learning. IEEE Trans Ind Inform 14(1):241–252
Silver D, Lever G, Heess N, Degris T, Wierstra D, Riedmiller M (2014) Deterministic policy gradient algorithms
Stanley KO, Clune J, Lehman J, Miikkulainen R (2019) Designing neural networks through neuroevolution. Nat Mach Intell 1(1):24–35
Stone P, Veloso M (2000) Multiagent systems: a survey from a machine learning perspective. Auton Robots 8(3):345–383
Stulp F, Buchli J, Ellmer A, Mistry M, Theodorou EA, Schaal S (2012) Model-free reinforcement learning of impedance control in stochastic environments. IEEE Trans Auton Mental Dev 4(4):330–341
Such FP, Madhavan V, Conti E, Lehman J, Stanley KO, Clune J (2017) Deep neuroevolution: genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. Preprint arXiv:171206567
Sutton RS (1988) Learning to predict by the methods of temporal differences. Mach Learn 3(1):9–44
Sutton RS (1992) A special issue of machine learning on reinforcement learning. Mach Learn 8
Sutton RS, Barto AG (2018) Reinforcement learning: an introduction. MIT press
Sutton RS, McAllester DA, Singh SP, Mansour Y (2000) Policy gradient methods for reinforcement learning with function approximation. In: Advances in neural information processing systems, pp 1057–1063
Tenorio-Gonzalez AC, Morales EF, Villaseñor-Pineda L (2010) Dynamic reward shaping: training a robot by voice. In: Ibero-American conference on artificial intelligence, Springer, pp 483–492
Theodorou E, Buchli J, Schaal S (2010) A generalized path integral control approach to reinforcement learning. J Mach Learn Res 11:3137–3181
Thomaz AL, Breazeal C (2008) Teachable robots: understanding human teaching behavior to build more effective robot learners. Artif Intell 172(6–7):716–737
Truong XT, Ngo TD (2017) Toward socially aware robot navigation in dynamic and crowded environments: a proactive social motion model. IEEE Trans Autom Sci Eng 14(4):1743–1760
Tsitsiklis JN, Van Roy B (1996) Feature-based methods for large scale dynamic programming. Mach Learn 22(1–3):59–94
Tsitsiklis JN, Van Roy B (1997) Analysis of temporal-diffference learning with function approximation. In: Advances in neural information processing systems, pp 1075–1081
Turan M, Almalioglu Y, Gilbert HB, Mahmood F, Durr NJ, Araujo H, Sarı AE, Ajay A, Sitti M (2019) Learning to navigate endoscopic capsule robots. IEEE Robot Autom Lett 4(3):3075–3082

Tzafestas SG, Rigatos GG (2002) Fuzzy reinforcement learning control for compliance tasks of robotic manipulators. IEEE Trans Syst Man Cybern Part B (Cybern) 32(1):107–113
Van Hasselt H, Guez A, Silver D (2016) Deep reinforcement learning with double q-learning. In: Thirtieth AAAI conference on artificial intelligence, IEEE, pp 2094–2100
Viseras A, Garcia R (2019) Deepig: multi-robot information gathering with deep reinforcement learning. IEEE Robot Autom Lett 4(3):3059–3066
Vlassis N (2007) A concise introduction to multiagent systems and distributed artificial intelligence. Synth Lect Artif Intell Mach Learn 1(1):1–71
Wang C, Wang J, Shen Y, Zhang X (2019) Autonomous navigation of uavs in large-scale complex environments: a deep reinforcement learning approach. IEEE Trans Veh Technol 68(3):2124–2136
Wang J, Xu X, Liu D, Sun Z, Chen Q (2013) Self-learning cruise control using kernel-based least squares policy iteration. IEEE Trans Control Syst Technol 22(3):1078–1087
Wang JP, Shi YK, Zhang WS, Thomas I, Duan SH (2018a) Multitask policy adversarial learning for human-level control with large state spaces. IEEE Trans Ind Inform 15(4):2395–2404
Wang S, Chaovalitwongse W, Babuska R (2012) Machine learning algorithms in bipedal robot control. IEEE Trans Syst Man Cybern Part C (Appl Rev) 42(5):728–743
Wang Y, Lang H, De Silva CW (2010) A hybrid visual servo controller for robust grasping by wheeled mobile robots. IEEE/ASME Trans Mech 15(5):757–769
Wang Y, He H, Sun C (2018b) Learning to navigate through complex dynamic environment with modular deep reinforcement learning. IEEE Trans Games 10(4):400–412
Watkins CJ, Dayan P (1992) Q-learning. Mach Learn 8(3–4):279–292
Watkins CJCH (1989) Learning from delayed rewards
Whitbrook AM, Aickelin U, Garibaldi JM (2007) Idiotypic immune networks in mobile-robot control. IEEE Trans Syst Man Cybern Part B (Cybern) 37(6):1581–1598
Williams RJ (1992) Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach Learn 8(3–4):229–256
Witten IH (1977) An adaptive optimal controller for discrete-time markov environments. Inf Control 34(4):286–295
Wu C, Ju B, Wu Y, Lin X, Xiong N, Xu G, Li H, Liang X (2019) Uav autonomous target search based on deep reinforcement learning in complex disaster scene. IEEE Access 7:117227–117245
Xi A, Mudiyanselage TW, Tao D, Chen C (2019) Balance control of a biped robot on a rotating platform based on efficient reinforcement learning. IEEE/CAA J Autom Sin 6(4):938–951
Xiao L, Xie C, Min M, Zhuang W (2017) User-centric view of unmanned aerial vehicle transmission against smart attacks. IEEE Trans Veh Technol 67(4):3420–3430
Xu X, Liu C, Yang SX, Hu D (2011) Hierarchical approximate policy iteration with binary-tree state space decomposition. IEEE Trans Neural Netw 22(12):1863–1877
Yang E, Gu D (2004) Multiagent reinforcement learning for multi-robot systems: a survey. Tech. rep., tech. rep
Yang X, He H, Liu D (2017) Event-triggered optimal neuro-controller design with reinforcement learning for unknown nonlinear systems. IEEE Trans Syst Man Cybern Syst
Yang Z, Merrick K, Jin L, Abbass HA (2018) Hierarchical deep reinforcement learning for continuous action control. IEEE Trans Neural Netw Learn Syst 29(11):5174–5184
Ye C, Yung NH, Wang D (2003) A fuzzy controller with supervised learning assisted reinforcement learning algorithm for obstacle avoidance. IEEE Trans Syst Man Cybern Part B (Cybern) 33(1):17–27
Yin S, Zhao S, Zhao Y, Yu FR (2019) Intelligent trajectory design in uav-aided communications with reinforcement learning. IEEE Trans Veh Technol 68(8):8227–8231
Yu C, Zhang M, Ren F, Tan G (2015a) Multiagent learning of coordination in loosely coupled multiagent systems. IEEE Trans Cybern 45(12):2853–2867
Yu J, Wang C, Xie G (2015b) Coordination of multiple robotic fish with applications to underwater robot competition. IEEE Trans Ind Electron 63(2):1280–1288
Yung NH, Ye C (1999) An intelligent mobile vehicle navigator based on fuzzy logic and reinforcement learning. IEEE Trans Syst Man Cybern Part B (Cybern) 29(2):314–321
Zalama E, Gomez J, Paul M, Peran JR (2002) Adaptive behavior navigation of a mobile robot. IEEE Trans Syst Man Cybern Part A Syst Hum 32(1):160–169
Zeng Y, Zhang R, Lim TJ (2016) Wireless communications with unmanned aerial vehicles: opportunities and challenges. IEEE Commun Mag 54(5):36–42
Zhang H, Jiang H, Luo Y, Xiao G (2016) Data-driven optimal consensus control for discrete-time multi-agent systems with unknown dynamics using reinforcement learning method. IEEE Trans Ind Electron 64(5):4091–4100

Zhang J, Tai L, Yun P, Xiong Y, Liu M, Boedecker J, Burgard W (2019) Vr-goggles for robots: real-to-sim domain adaptation for visual control. IEEE Robot Autom Lett 4(2):1148–1155

Zhou L, Yang P, Chen C, Gao Y (2016) Multiagent reinforcement learning with sparse interactions by negotiation and knowledge transfer. IEEE Trans Cybern 47(5):1238–1250

Zhu J, Zhu J, Wang Z, Guo S, Xu C (2018) Hierarchical decision and control for continuous multitarget problem: policy evaluation with action delay. IEEE Trans Neural Netw Learn Syst 30(2):464–473

Zhu Y, Mottaghi R, Kolve E, Lim JJ, Gupta A, Fei-Fei L, Farhadi A (2017) Target-driven visual navigation in indoor scenes using deep reinforcement learning. In: 2017 IEEE international conference on robotics and automation (ICRA), IEEE, pp 3357–3364