



# Introduction to Reinforcement Learning: A Robotics Perspective

**Taken from:** <https://lamarr-institute.org/blog/reinforcement-learning-and-robotics/>

# Introduction to Reinforcement Learning: A Robotics Perspective



Reinforcement Learning (RL) is known as a branch of Machine Learning where an intelligent agent takes decisions in an environment to maximize a cumulative reward. In contrast to other machine learning paradigms, the training data thus is generated in an interactive process of exploration and exploitation. This blog post explores the general idea of applying RL to the fast-growing domain of robotics, as well as potential applications, and outstanding challenges that arise.

## Robotic Workflow – How Robots Learn to Act

The overall RL workflow can be seen as an iterative learning process due to sensorimotor activity of an agent with an environment (compare Fig. 1) – a natural and very similar perspective of how we humans learn too!

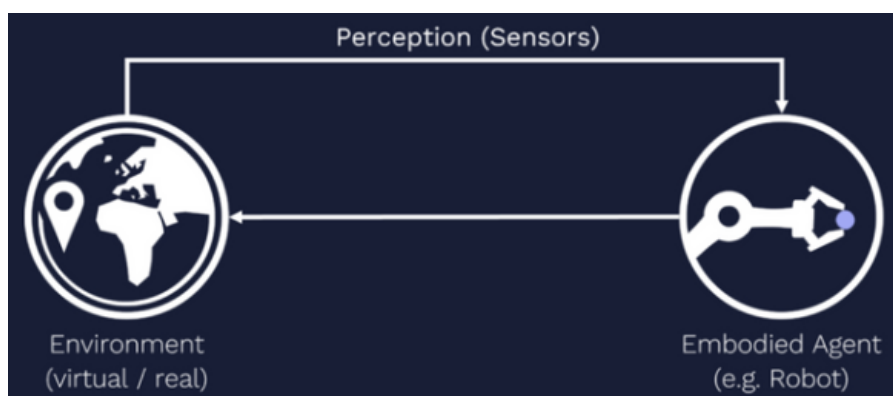
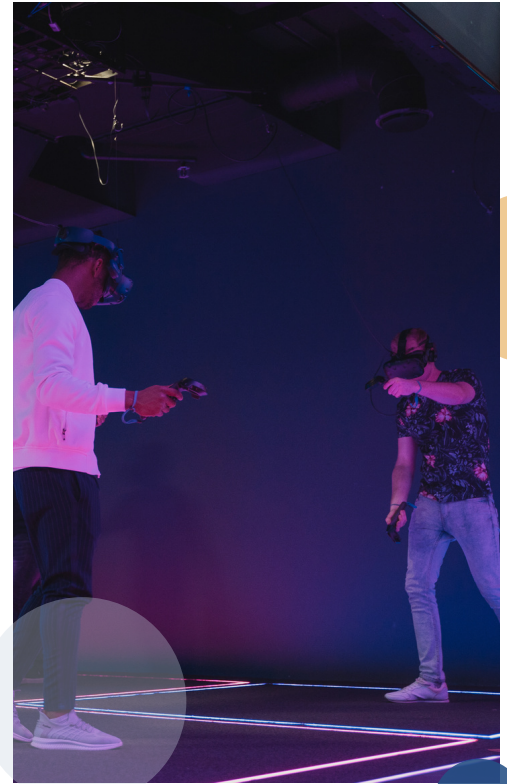


Fig. 1: The workflow of Reinforcement Learning in the context of robotics. An iterative learning process of an agent with the environment as result of sensorimotor activity and reward signals. © Fraunhofer IML/ Julian Eßer

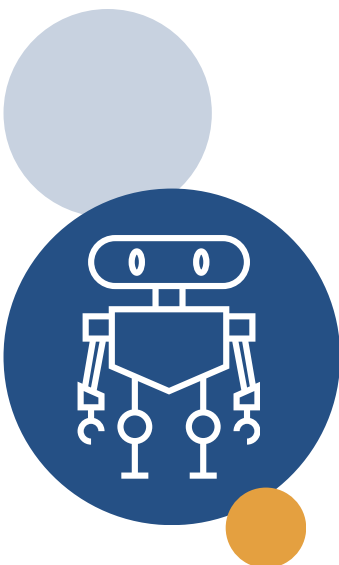


## Environment

The first step lies in setting up the training environment, often a simulation environment, with specifications for actions, observations, and rewards. In the context of robotics, the observation space typically refers to the available sensor sources on the real robotic system and desired control inputs. While discrete action spaces are often found in other RL applications, in robotics typically continuous actions are preferred, covering e.g., position or velocity joint targets. Since robotic tasks often involve constraints either of the physical system (e.g., joint limits) or some desired behavior style, usually dense reward functions are used to explicitly encode some goal specifications.



## Training



The second step of RL in robotics then includes specifying the actual training scheme of the agent. Although different ways exist for representing the final policy, nowadays typically deep neural networks are adopted for the state-action mapping because of their capabilities to handle nonlinearities. Moreover, a wide range of potential algorithms have been proposed over the last years. For robotic control, typically model-free RL algorithms are adopted since they do not require a ground-truth model of the environment that often is not available for the robot.

## Deployment

Lastly, after successfully evaluating the trained policies in the virtual training environments, it's time to deploy it to the real robotic system! In the end, the success of the transfer depends on many factors, such as the gap between the virtual and real world, the difficulty of the learning task ahead, or the complexity of the robot platform itself.



## Robotic Applications – Bringing RL to the Real World

Reinforcement Learning is being applied to a wide variety of robotic platforms and tasks right now. In particular, learning-based control approaches are especially promising when solving hard-to-engineer behaviors with a high degree of uncertainty – as we typically find in the real world. So, let's take a look at three very different applications of applying RL to different real-world robotic tasks:

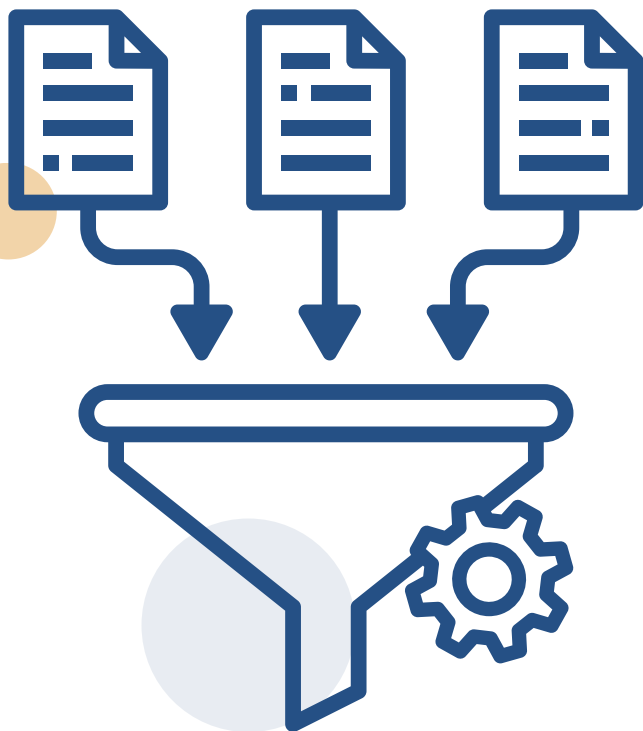
- Learning Quadrupedal Locomotion over Challenging Terrain
- 2a: Locomotion
- Autonomous Drone Racing with Deep Reinforcement Learning (IROS 2021)
- 2b: Navigation
- Robots learn Grasping by sharing their hand-eye coordination experience with each other | QPT
- 2c: Manipulation



## Locomotion

One of the ongoing research areas in RL for robotics lies in learning dynamic locomotion. In this case, the complexity of this task rises not only with the type of robotic platform, but also with terrain to be traversed. For instance, locomotion of a four-wheeled robot on flat terrain generally can be considered a rather simple task, while locomotion of a quadrupedal robot over rough terrain already involves higher degrees of freedom and uncertainties (see Fig. 2a). However, even for simple robotic platforms (such as a four-wheeled robot) in simple environments (e.g. flat floor) as seen in the first example, the complexity potentially further increases when considering highly-dynamic motions with drifts, making it again hard to model due to the larger Sim-to-Real gap.

## Manipulation



Likewise, also the field of robotic manipulation can benefit from learning-based control approaches such as RL. Although the task of grasping a specific object with a certain robot platform certainly is not easy, it can be considered solvable if it is deployed in a controlled environment. However, if the objective is to grasp and manipulate objects that are not known beforehand and located in environments that may involve many other (dynamic) objects (see Fig. 2c), learning-based approaches can offer unique capabilities. For instance, vision data could be used to train RL agents that can understand the scene in a humanlike way to make appropriate decisions.

## Navigation

Another application of RL in robotics is the field of navigation, which involves not only sensory perception of the environment but also higher-level reasoning about the decisions to make. Again, RL here can play to its strengths, as the environments turn out to be difficult to model. For instance, the task of navigating a static environment that is known a priori can be considered as relatively straightforward. However, if the goal would be quickly navigating a drone in an unknown environment (see Fig. 2b) that may even involve dynamic obstacles (such as other robots), the level of uncertainty rises to a large extent – making it a perfect field application for RL again.



## Robotic Challenges – The Road Ahead

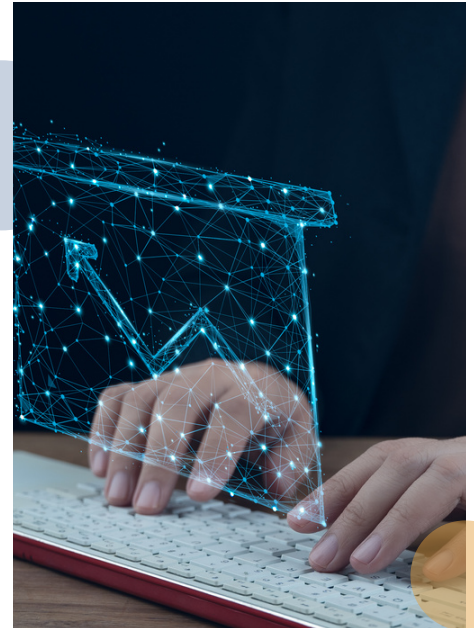
Although already promising applications have shown the potential of Reinforcement Learning for robot control, particular challenges are related to the training of robots for real-world tasks.

### 1. Performance (Effectiveness):

One of the current challenges of applying RL to robotics lies in achieving the desired level of performance of the trained agents. One of the reasons this may be challenging is that RL as a data-driven approach is sensitive to a specific selection of hyperparameters. But even for the exact same selection of hyperparameters, the overall return in the training can be different due to changing random seed in the environment. Another part of the challenge lies in the fact that RL tends to tap into local minima, depending on the reward formulation. For instance, local minima could be achieved by exploiting wrongly specified dense rewards, while in the case of sparse rewards the convergence to performant policies can be even more difficult.

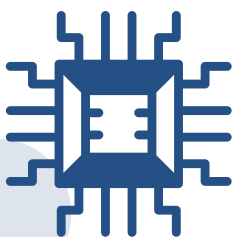
## 2. Amount of Data (Efficiency):

Another key challenge when applying RL to robotics comes from the fact that RL typically needs millions of interactions with the environment in order to converge, which is impractical for the real world. Although different kinds of RL algorithms have been proposed over the last years, such as various off- and on-policy algorithms or even model-based algorithms, the problem of high sample cost and computational exploitation is far from being solved. To mitigate this gap, simulation environments are adopted for providing virtual training environments for the RL agents.



## 3. Sim-to-Real Gap:

Simulation environments often are adopted in RL for robotics to generate training samples for robotic tasks in a fast, safe, and scalable way. However, since simulations are abstractions from the real robot and the real world, there will always remain a difference between these two worlds – also known as the sim-to-real gap. For instance, the physical behavior of the robot in simulation strongly relies on the quality of the modeling of the kinematics, dynamics, actuators, and so on. Moreover, also perceiving the environment in simulation is part of the sim-to-real gap since the realism strongly depends on the underlying sensor models that are used. Lastly, delays in communication as well as hardware failures are typically found on real robotic systems and contribute to the sim-to-real gap when applying RL to the domain of robotics.





# Summary

In this blog post we've explored the potential of RL for the purpose of robotic control. In particular, we've seen how a typical workflow for applying RL to a robotic task looks like and that it can be applied to a wide range of potential robotic applications. However, we've also seen that robotics brings its own challenges and complexities to this field. How can we train policies for robotic control in an efficient and effective way so that they successfully transfer to the real robotic system? We will explore one potential path towards this goal in the next blog post – stay tuned ...

