Reinforcement Learning in Robotics

from

Tim Reipschläger

31.05.2018

Contents

- Introduction to Reinforcement Learning
- Model-based vs. Model-free approaches
- Q-Learning
- Actor-Critic-Learning
- Collective Learning
- Trends

Introduction to Reinforcement Learning

What is Reinforcement Learning?

- An agent explores an environment and receives feedback in form of rewards
- The agent tries to learn a optimal policy

Introduction to Reinforcement Learning

Terms of Reinforcement Learning:

- A state(s) determines a possible state of the environment
- An action(a) determines a possible action that changes the state of the environment
- A reward(R) determines what reward is given for certain states or state action combinations
- A policy(Π) determines which action is taken in the actual state

Introduction to Reinforcement Learning

Why is Reinforcement Learning used in robotics?

- a robot can autonomously learn an optimal behavior
- Instead of describing the solution in detail, only rewards have to be given for reaching goals
- Policies are learned not concrete action sequences

Introduction to Reinforcement Learning

Challenges of Reinforcement Learning used in robotics:

- States and actions of the robots are continuous
- Complex and dynamic physical systems
- Behaviors learned in a simulator can’t be transferred directly to the real robot
- Good reward functions are needed for the learning process

Model-based vs Model-free approaches

Model-based:

- The agent creates a model of the environment
- A transition function(T) is generated which takes a state and an action and then predicts the following state
  - $T(s,a) = s'$
- Once the environment is modelled the policy can be found using a planning algorithm

Model-based vs Model-free approaches

Model-free:

- A model is not needed to find a good policy
- Q-Learning and Actor-Critic methods evaluate actions to determine the best action in a given state
- No model of the environment is created

Q-Learning

- Neural Network is used to estimate Q-Values
- Q-Values map a state of the environment to a numerical value for each possible action in this state
- Q-Values indicate which action is expected to result in the highest future reward
- Q-Values are used to decide which action should be performed in the actual state

Q-Learning

Learning procedure:

- At the start the Q-Values are 0
- The agent starts randomly exploring the environment and gets rewards
- After some exploration the Q-Values get updated with an update function

Q-Learning

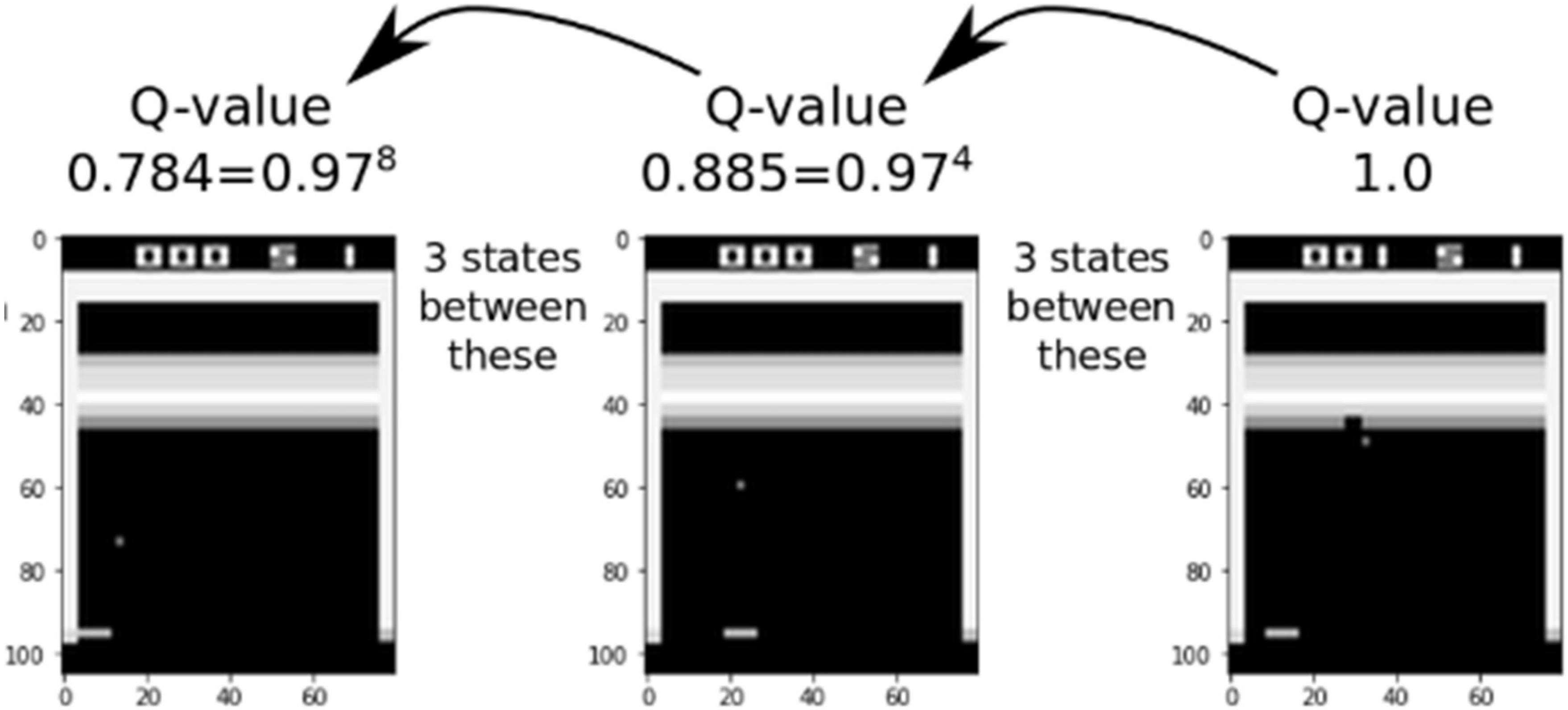
Q-Value update function:

$$Q(s_t, a_t) = \ r_t + \gamma \ \cdot \text{max}Q(s_{t+1}, a)$$

- $\boldsymbol{Q}(\boldsymbol{s_t}, \boldsymbol{a_t})$: Q-Value for action a in state s a at time t
- $\boldsymbol{r_t}$: Reward at time t
- $\boldsymbol{\gamma}$: discount factor
- **max**$\boldsymbol{Q}(\boldsymbol{s_{t+1}}, \boldsymbol{a})$: Maximal Q-Value in the state s at time t+1 for any possible action a

Q-Learning

Q-Value update function:



Q-Learning

Learning procedure:

- At the start the Q-Values are 0
- The agent starts randomly exploring the environment and gets rewards
- After some exploration the Q-Values get updated with an update function
- The Network gets trained to output the updated Q-Values

Actor-Critic-Learning

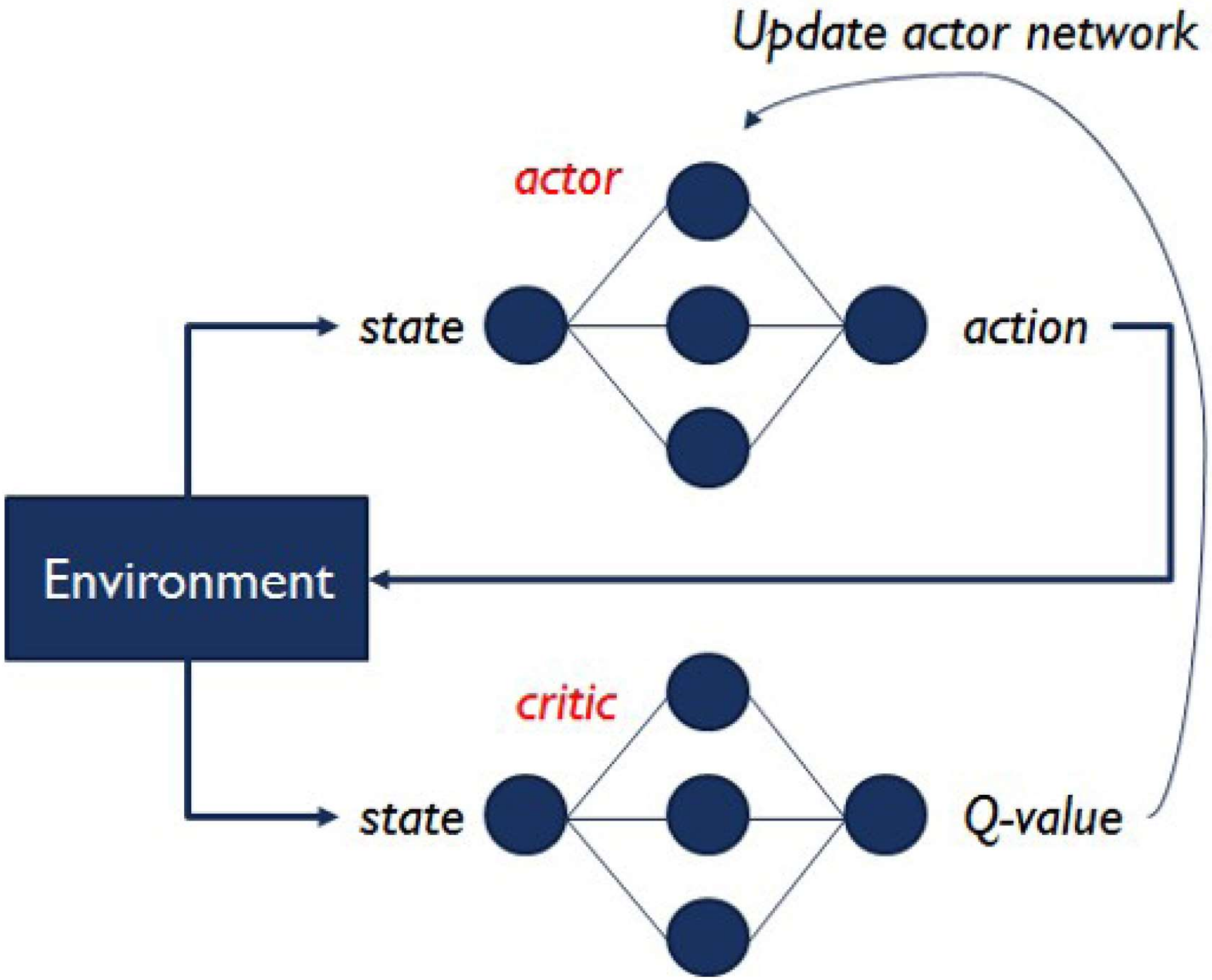
- Uses two Neural Networks
- The Actor Neural Network selects the action for the actual state
- The Critic Neural Network evaluates the action taken in a state

Actor-Critic-Learning

Training of Actor and Critic:

- The Critic is updated so the predicted values correspond to the experienced values
- The Actor is updated using the evaluation of the Critic
- To determine good and bad actions a baseline can be used

Actor-Critic-Learning



Collective Learning

- Experiences of robots can be shared with each other
- Using multiple robots decreases the time needed for learning
- Small changes between tasks of the different robots increase adaptability
- Consists of local worker and global worker

Collective Learning

Learning procedure:

- Pretraining of convolution layers
- Teacher demonstrates the task
- Local Worker generates sample trajectories

Collective Learning

Learning procedure:

- Samples are used to optimize the local policy
- Optimized trajectories are appended to a global memory
- Global Worker uses the optimized trajectories to train its Neural Network

Trends

- Using Deep Reinforcement learning to solve more complex tasks
- Train the desired behavior on the actual hardware and not in a simulator
- Use imitation learning on actual hardware to learn tasks

The End

Thank you for your attention.

Any questions?

References

Pictures:

- Denisov Sergey, Jee-Hyong Lee „Actor-Critic Algorithm with Transition Cost Estimation" Figure 3 https://www.e-sciencecentral.org/articles/SC000021256

- Magnus Erik Hvass Pedersen "Reinforcement Learning (Q-Learning)" Figure 2 https://github.com/Hvass-Labs/TensorFlow-Tutorials/blob/master/16_Reinforcement_Learning.ipynb

References

- Jens Kober, J. Andrew Bagnell and Jan Peters
  “Reinforcement learning in robotics: A survey“
- Christopher J.C.H. Watkins and Peter Dayan “Q-Learning“
- Magnus Erik Hvass Pedersen “Reinforcement Learning (Q-Learning)“ https://github.com/Hvass-Labs/TensorFlow-Tutorials/blob/master/16_Reinforcement_Learning.ipynb
- Vijay R. Konda and John N. Tsitsiklis „Actor-Critic Algorithms“
- Ali Yahya, Adrian Li, Mrinal Kalakrishnan, Yevgen Chebotar and Sergey Levine
- Collective Robot Reinforcement Learning with Distributed Asynchronous Guided Policy Search