



ITMO UNIVERSITY

Машинное обучение и элементы искусственного интеллекта в робототехнике

Ведяков А. А Антипов В. А Труфанова А. А Овчаров А. О Чергинец Д. А Беспалов В. В



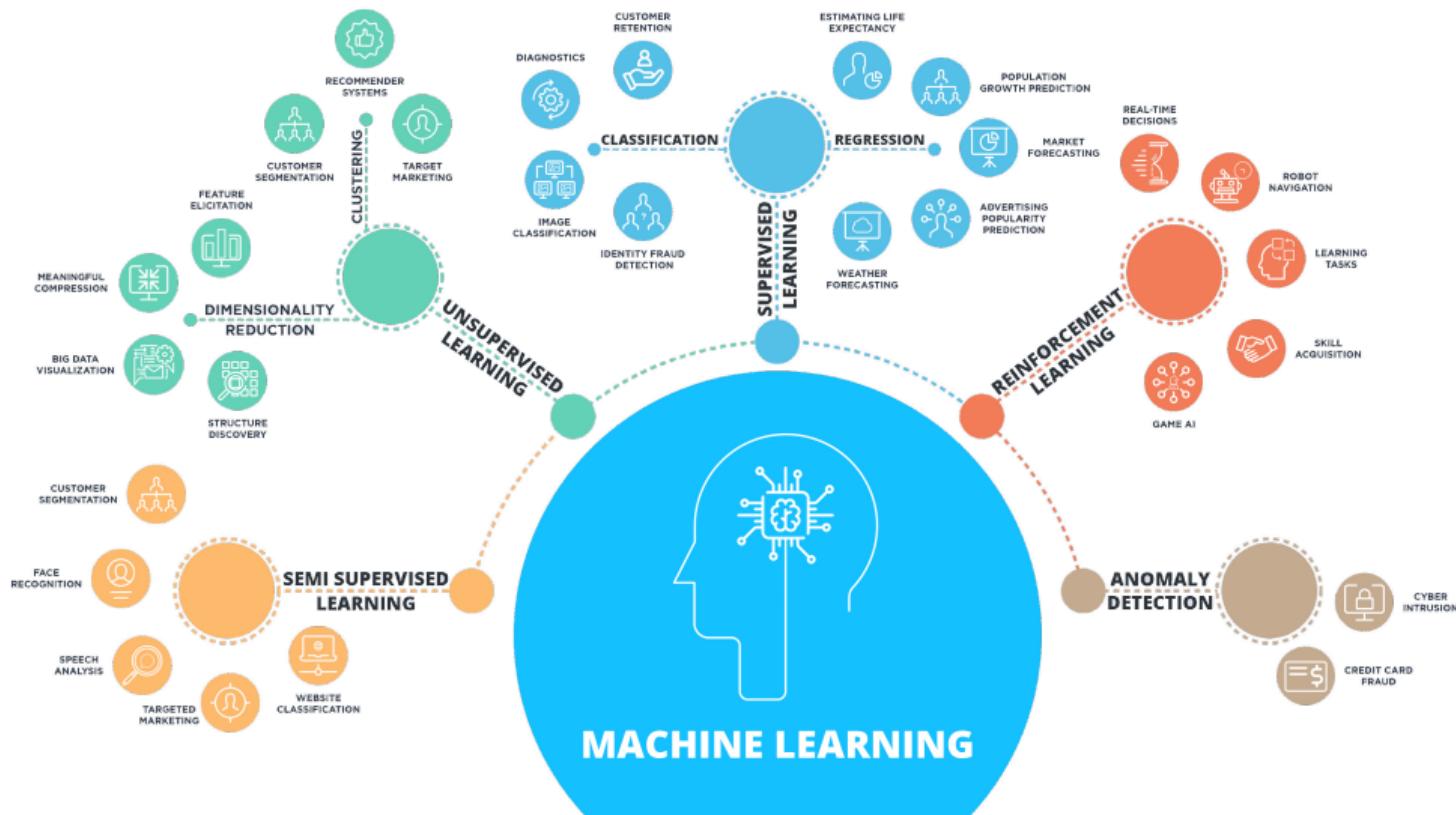
Figure: Hugo – DARPA Robotics

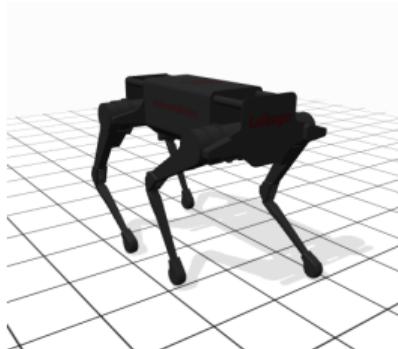
- ▶ Интеллектуальный агент — это система, действующая разумно. То, что он делает, соответствует его цели и обстоятельствам, он гибок к изменяющейся среде и меняющимся целям, он учится на опыте и делает соответствующий выбор с учетом ограничений восприятия и конечных вычислений.
- ▶ Машинное обучение (ML, Machine Learning) — класс методов, характеризующийся нацеленностью не на прямое решение задачи, а обучение для её решения.

- ▶ In many cases, AI can surpass the best human performance levels at specific tasks.
- ▶ In order to understand someone else, it is necessary to know oneself.
- ▶ How could a software box have a subjective viewpoint of, and in, the physical world, the world that humans inhabit?

Source

- Mark Lee:** Why AI can't ever reach its full potential without a physical body: <https://theconversation.com/why-ai-cant-ever-reach-its-full-potential-without-a-physical-body-146870>.
- Mark H. Lee:** How to Grow a Robot. Developing Human-Friendly, Social AI, 2020





(a) Управление роботом



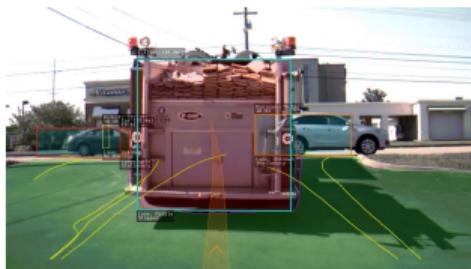
(c) Взаимодействие с окружающими объектами



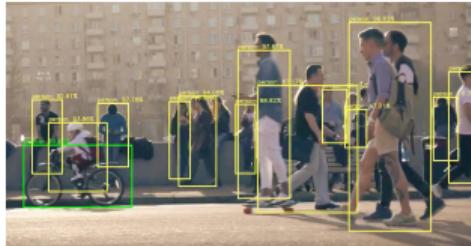
(b) Распознавание и синтез речи



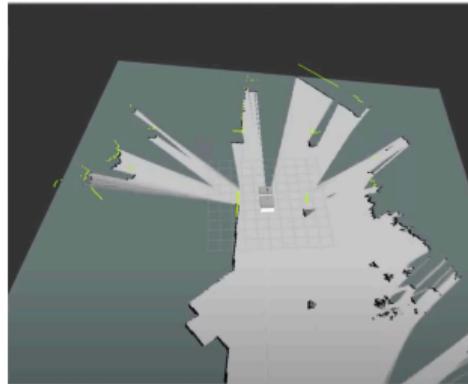
(d) Синтез новых систем и конструкций



(a) Идентификация окружения



(c) Детектирование людей в движении



(b) SLAM



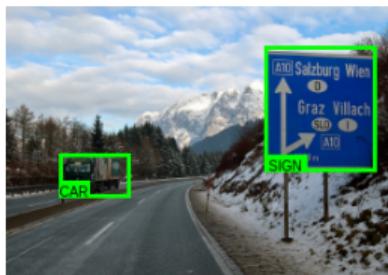
(d) Отслеживание брака на производстве



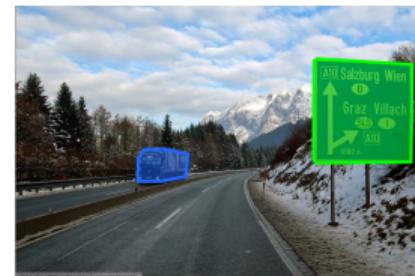
(a) Классификация изображения



(b) Локализация объекта



(c) Детектирование объектов



(d) Сегментация

Figure: Задачи компьютерного зрения

- ▶ Интерпретируемость, надежность и гарантии
- ▶ Нехватка данных
- ▶ Катастрофическое забывание

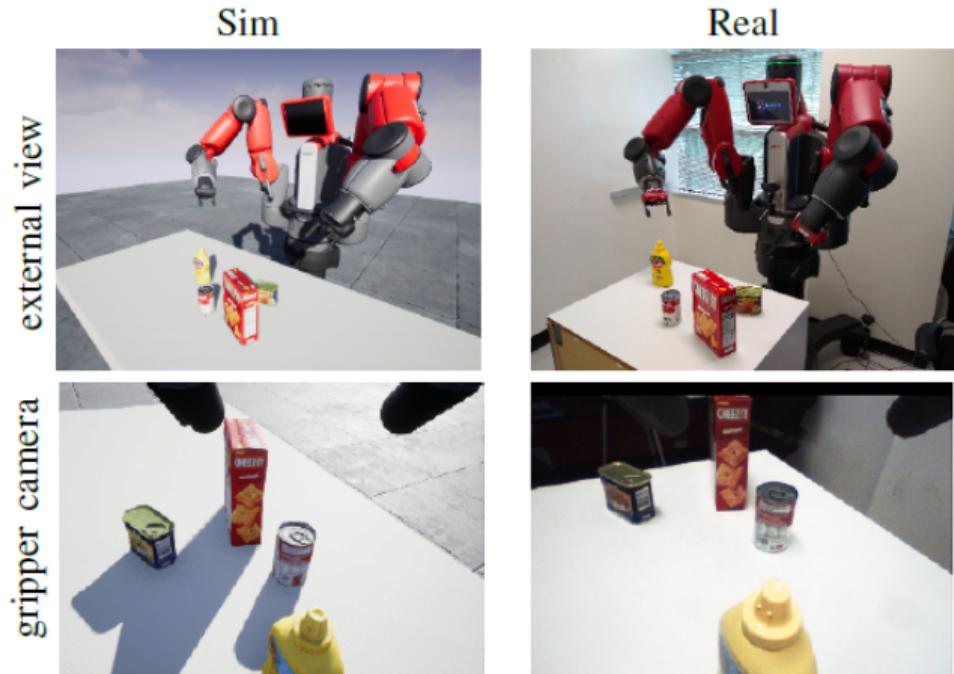


Figure: Toward Sim-to-Real Directional Semantic Grasping
(NVIDIA)

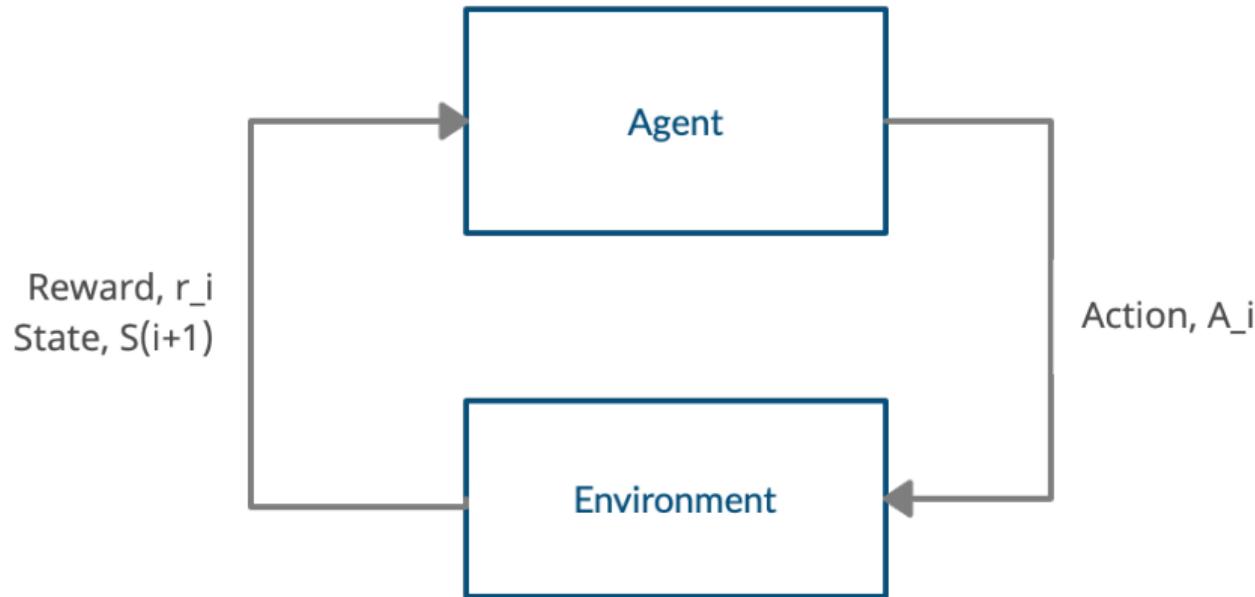


Figure: Reinforcement Learning

Рассмотрим Стандартную задачу обучения с подкреплением, состоящую из взаимодействия агента с окружающей средой. Для упрощения: Окружающая среда полностью наблюдаема:

Enviroment

Среда описывается набором состояний S , набором действий A и распределением начальных состояний $P(s_0)$, а также функцией вознаграждения: $r : S \times A \rightarrow R$, функцией вероятности перехода: $P(s_{t+1}|s_t, a_t)$ и коэффициентом дисконтирования $\gamma \in [0, 1]$.

Policy

Детерминированная политика (стратегия) — это отображение состояний в действия: $\pi : S \rightarrow A$. Каждый эпизод начинается с выбора начального состояния s_0 . На каждом временном шаге t агент производит действие в зависимости от текущего состояния: $a_t = \pi(s_t)$. Затем он получает награду $r_t = r(s_t, a_t)$, и новое состояние среды берется из распределение $p(\cdot|s_t, a_t)$.

Return, Q-function

Сумма всех будущих наград называется вознаграждением (доходностью):

$$R_t = \sum_{i=t}^{\infty} \gamma^{i-t} r_i$$

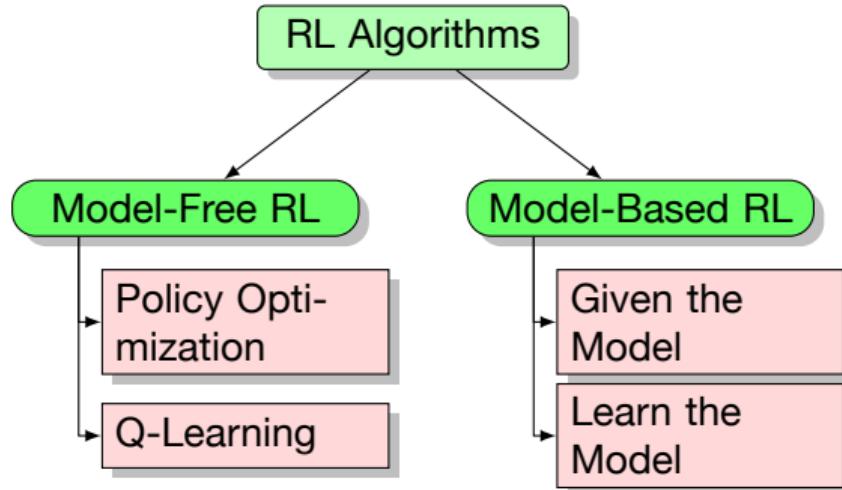
Цель агента — максимизировать ожидаемое вознаграждение (прибыль) $E_{s_0}[R_0|s_0]$. Q-функция или функция действий-значений определяется как:

$$Q^\pi(s_t, a_t) = E[R_t|s_t, a_t]$$

Optimal Q-function

Пусть π^* — обозначает оптимальную политику. Все оптимальные политики имеют одну и ту же Q-функцию, которая называется оптимальной Q-функцией и удовлетворяет уравнению Беллмана:

$$Q^*(s, a) = E_{s' \sim p(\cdot|s, a)} \left[r(s, a) + \gamma \max_{a' \in A} Q^*(s', a') \right] \quad (1)$$



- <https://ai.google>
Artificial intelligence
- <https://ai.googleblog.com>
Blog Google AI
- <https://sites.google.com/site/brainrobotdata>
Deep learning
-
- <https://ai.googleblog.com/2018/06/scalable-deep-reinforcement-learning.html>
Robotic Manipulation

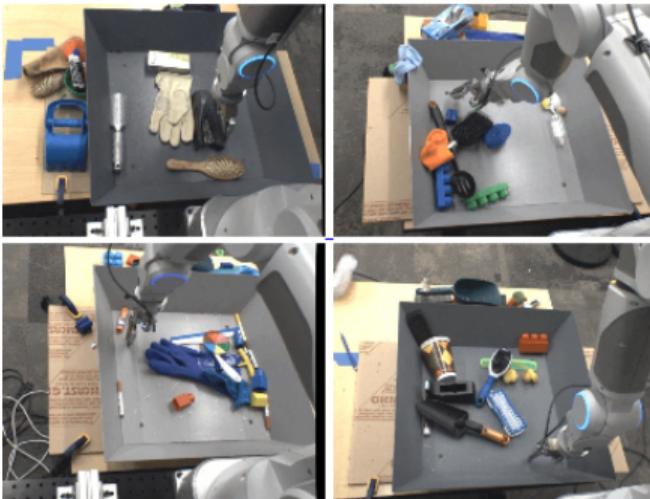


Figure: Link

“We want to use AI to augment the abilities of people, to enable us to accomplish more and to allow us to spend more time on our creative endeavors.”
Jeff Dean, Google Senior Fellow

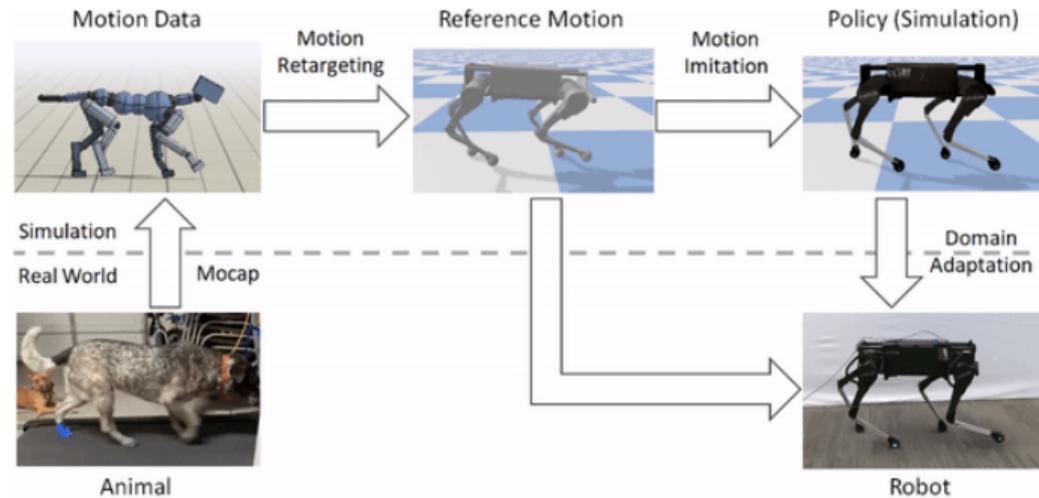


Figure: Link



Figure: Link



Figure: Link

Gym Open AI: CartPole-v1

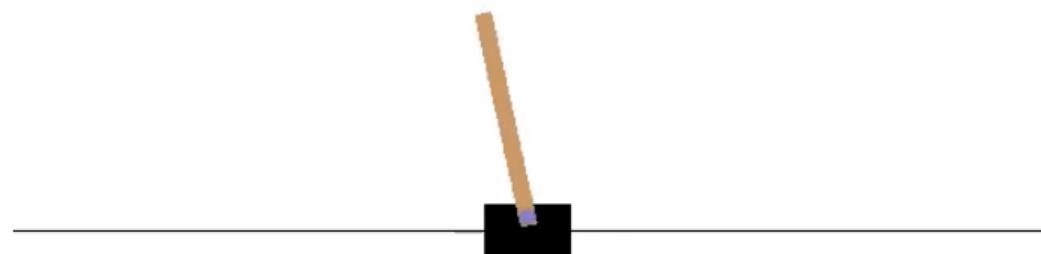


Figure: Link

Наблюдаемые состояния	Min	Max
Положение тележки	-4.8	4.8
Скорость тележки	$-\infty$	∞
Угол обратного маятника	-0.418 рад (-24°)	0.418 рад (24°)
Угловая скорость обратного маятника	$-\infty$	∞

Table: Правила функционирования модели

Завершение эпизода:

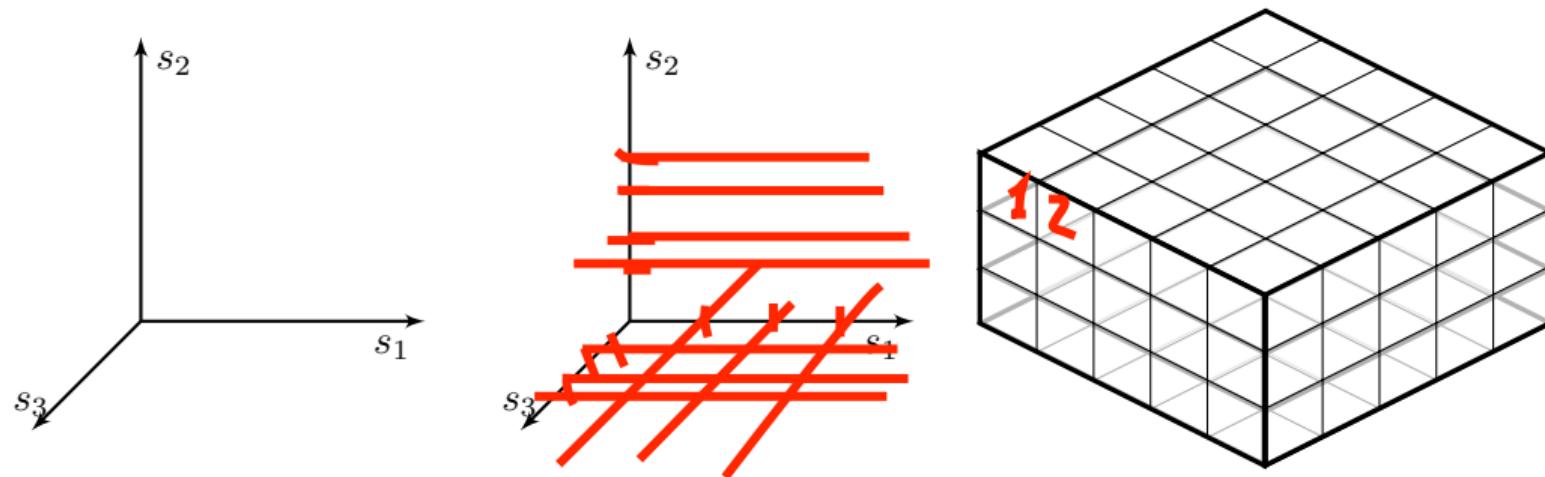
- ▶ Угол маятника по модулю больше 12° .
- ▶ Положение тележки больше 2.4 (центр тележки достигает края дисплея).
- ▶ Длительность эпизода больше 200 шагов.
- ▶ Задача решена, если средняя длительность эпизода больше 195.0 за более чем 100 испытаний подряд.

Действия:

- ▶ 0 – Толкнуть тележку влево
- ▶ 1 – Толкнуть тележку вправо

Вознаграждение: 1 за каждый сделанный шаг, включая шаг завершения.

Начальное состояние: всем начальным состояниям присваивается случайное значение с равномерным распределением в диапазоне: $[-0.05..0.05]$



Действия →
Состояния →

Q-Table		Действия	
Состояния	0	0	0
	.	.	.
Состояния	100	0	0
	.	.	.
Состояния	200	0	0
	.	.	.

Обучение →

Q-Table		Действия	
Состояния	0	0	0
	.	.	.
Состояния	100	1.23	-3.23
	.	.	.
Состояния	200	43.03	23.69
	.	.	.

Знач-е Q →

Ключевое место в реализации алгоритма занимает функция, которая вычисляет качество комбинации состояния (S) - действие (A):

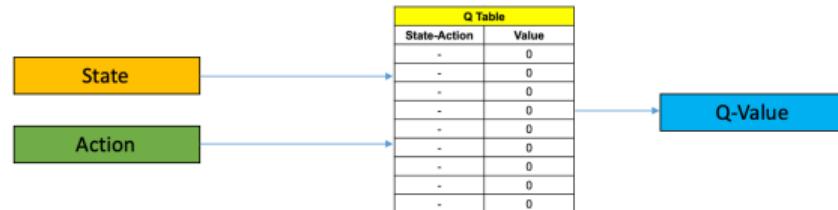
$$Q : S \times A \rightarrow \mathbb{R} \quad (2)$$

С помощью уравнения Беллмана происходит рекуррентное обновление функции Q на основе средневзвешенного значения старого значения и новой информации:

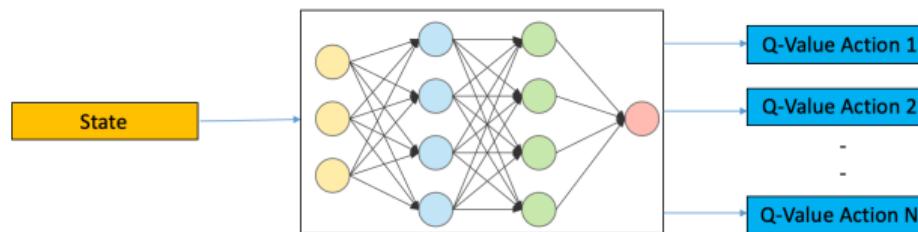
$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{старое значение}} + \alpha \overbrace{\left(r_t + \gamma \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{предсказание оптимальной величины}} - \underbrace{Q(s_t, a_t)}_{\text{старое значение}} \right)}^{\text{временная разница}}, \quad (3)$$

где $\alpha \in (0, 1]$ — коэффициент, соответствующий скорости обучения, r_t — награда, $\gamma \in [0, 1]$ — коэффициент дисконтирования. $Q(s_t, a_t)$ — старое значение целевой функции

$$Q^\pi(s_t, a_t) = \mathbb{E} [R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots | s_t, a_t] \quad (4)$$



Q Learning



Deep Q Learning

Figure: Q-table Learning vs Q-network¹

¹Choudhary, A Hands-On Introduction to Deep Q-Learning using OpenAI Gym in Python.

Q-network, behaviour distribution

Назовём аппроксиматор функции нейронной сети с весами θ Q-сетью.

Обучение достигается путём минимизации последовательности функций потерь $L_i(\theta_i)$, меняющихся на каждой итерации i :

$$L_i(\theta_i) = \mathbb{E}_{s,a \sim \rho(\cdot)} \left[(y_i - Q(s,a;\theta_i))^2 \right], \quad (5)$$

где $y_i = \mathbb{E}_{s' \sim \mathcal{E}} \left[r + \gamma \max_{a'} Q(s', a'; \theta_{i-1}) | s, a \right]$ – целевая функция для i -ой итерации, а $\rho(s, a)$ распределение вероятностей по последовательностям s и действиям a , которое мы будем называть **распределением поведения**.

Дифференцируя (5) по весам получим следующий градиент:

$$\nabla_{\theta_i} L_i(\theta_i) = \mathbb{E}_{s,a \sim \rho(\cdot); s' \sim \mathcal{E}} \left[\left(r + \gamma \max_{a'} Q(s', a'; \theta_{i-1}) - Q(s, a; \theta_i) \right) \nabla_{\theta_i} Q(s, a; \theta_i) \right]. \quad (6)$$

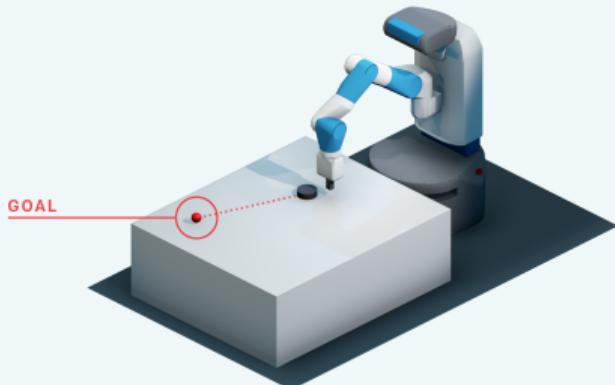


Figure: Желаемое положение шайбы

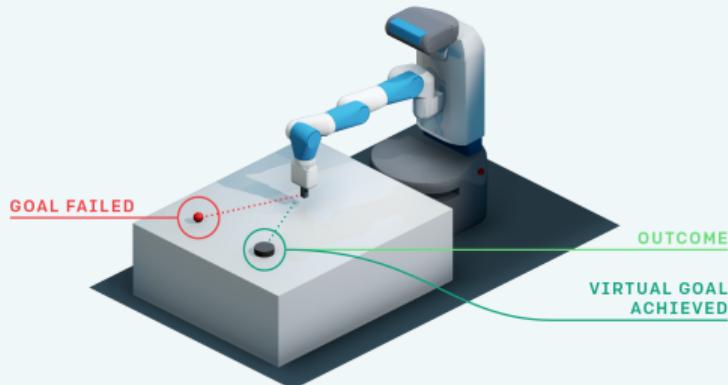


Figure: Виртуальная замена желаемого положения

<https://openai.com/blog/ingredients-for-robotics-research/>

Algorithm HER pseudocode

Given:

- ▶ an off-policy RL algorithm \mathbb{A}
e.g. DQN, DDPG, NAF, SDQN
- ▶ a strategy $\$$ for sampling goals for replay
e.g. $\$(s_0, \dots, s_T) = m(s_T)$
- ▶ a reward function $r : \mathcal{S} \times \mathcal{A} \times \mathcal{G} \rightarrow \mathbb{R}$
e.g. $r(s, a, g) = -[f_g(s) = 0]$

Initialize \mathbb{A}

e.g. initialize neural networks

Initialize replay buffer R

```
1: for episode=1, M do
2:   Sample a goal  $g$  and an initial state  $s_0$ 
3:   for  $t = 0, T - 1$  do
4:     Sample an action  $a_t$  using the behavioral policy from  $\mathbb{A}$ :  $a_t \leftarrow \pi_b(s_t || g)$ 
5:     Execute the action  $a_t$  and observe a new state  $s_{t+1}$ 
6:   end for
7:   for  $t = 0, T - 1$  do
8:      $r_t := r(s_t, a_t, g')$ 
9:     Store the transition  $(s_t || g, a_t, r', s_{t+1} || g')$  in  $R$ 
10:    Sample a set of additional goals for replay  $G := \$$ (current episode)
11:    for  $g' \in G$  do
12:       $r' := r(s_t, a_t, g')$ 
13:      Store the transition  $(s_t || g', a_t, r', s_{t+1} || g')$  in  $R$ 
14:    end for
15:  end for
16:  for  $t = 1, N$  do
17:    Sample a minibatch  $B$  from the replay buffer  $R$ 
18:    Perform one step of optimization using  $\mathbb{A}$  and minibatch  $B$ 
19:  end for
20: end for
```
