Robust Deep Reinforcement Learning for Quadcopter Control

26.01.2023

Friedrich Maximilian Sokol,
Nicolas Marco Hahn &
Marc Ubbelohde

Structure

- General setting
- Formulate the problem as a reinforcement problem
- AR-DDPG Algorithm
- Training and Testing
- Evaluation

Source: Dall-E

Unmanned Aerial Vehicles (UAV’s)
Quadrocopters

- are becoming increasingly cheaper
- made possible by developments in microelectronics
- wide range of applications
  - Aerial Photography
  - Search and Rescue
  - Agriculture
- motor configuration -> flexible flight behavior

[1]

[1] https://enterprise-insights.dji.com/blog/drones-in-agriculture

Hovering & Waypoint Navigation

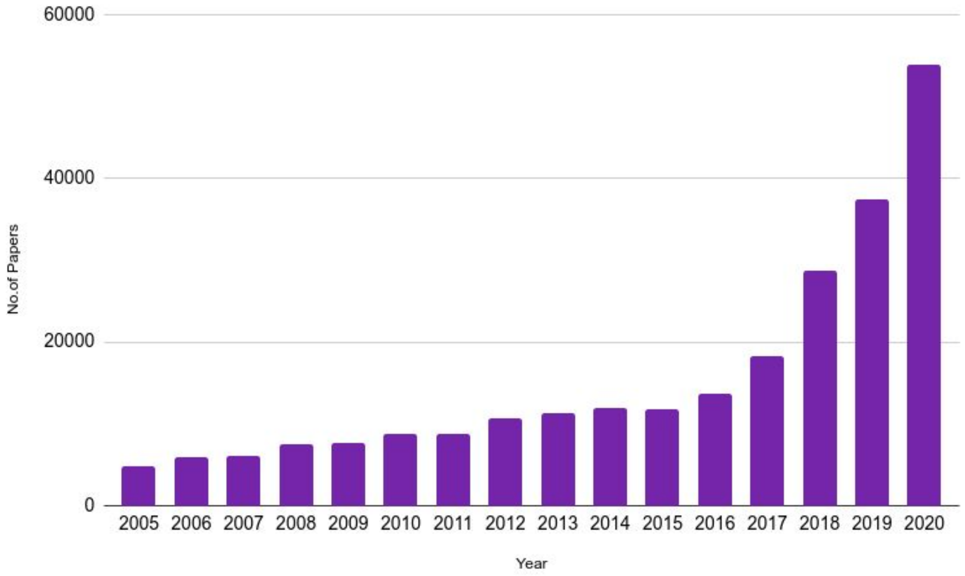
- crucial ability for stabilization
- necessary for many applications
- makes simple and automated operation possible
- precise data is necessary -> sensor fusion
- early controllers included PID-Controllers -> sensitive to misconfiguration
- GPS enabled waypoint navigation

[1]

[1] https://www.sphengineering.com/news/ugcs-photogrammetry-tool-for-uav-land-survey-missions

Recent Developments of Reinforcement Learning

- deep learning developments extend the scope of application for reinforcement learning
- high adaptability to dynamic environments
- simplifies implementation and makes tuning unnecessary
- more complex controls are made possible



[1] Number of released RL-papers by year

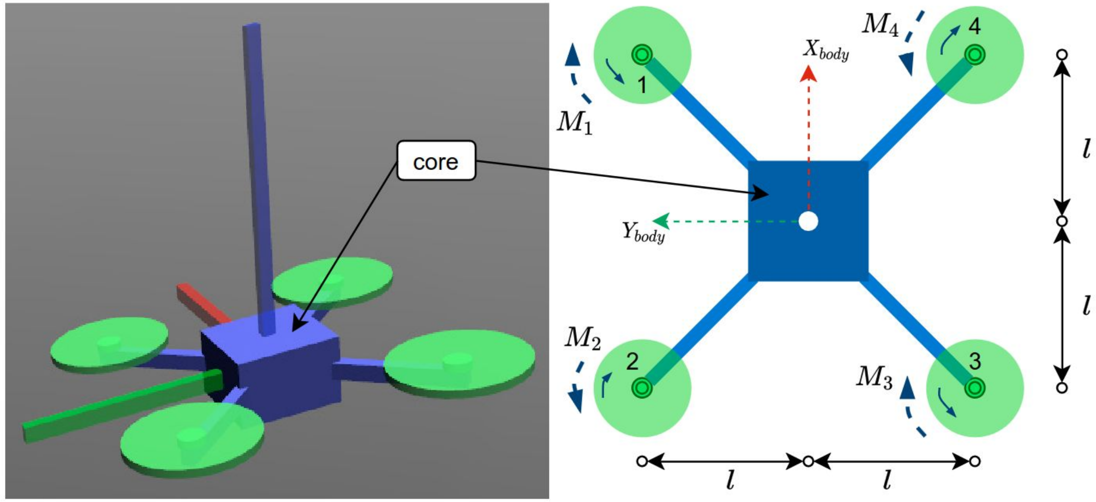
[1] https://www.researchgate.net/figure/Shown-are-the-number-of-Reinforcement-learning-papers-published-on-a-yearly-basis-The_fig1_351224274

Formulate the problem as a reinforcement problem

Quadcopter

- average quadcopter in X - configuration
- mass=1.5 kg, length=0.26 m, thrust range=[0, 15.0] N
- rotational matrix:

$$R = \begin{bmatrix} c_\psi c_\rho & s_\xi s_\rho c_\psi - s_\psi c_\xi & s_\xi s_\psi + s_\rho c_\xi c_\psi \\ s_\psi c_\rho & s_\xi s_\psi s_\rho + c_\xi c_\psi & -s_\xi c_\psi + s_\psi s_\rho c_\xi \\ -s_\rho & s_\xi c_\rho & c_\xi c_\rho \end{bmatrix}$$



[1]

- motion from actions:

$$m \begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -mg \end{pmatrix} + R \begin{pmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} F_i \end{pmatrix}$$

Translational motion [2]

$$\begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} = \begin{pmatrix} l(F_1 + F_2 - F_3 - F_4) \\ l(-F_1 + F_2 + F_3 - F_4) \\ -M_1 + M_2 - M_3 + M_4 \end{pmatrix} - \begin{pmatrix} p \\ q \\ r \end{pmatrix} \times I \begin{pmatrix} p \\ q \\ r \end{pmatrix}$$

Rotational motion [3]

[1-3] Deshpande, A. M., Minai, A. A., Kumar, M. (2021). Robust Deep Reinforcement Learning for Quadcopter Control

Formulate the problem as a reinforcement problem

Environment & State

- continuous 3-dimensional space
- discrete time step: 0.01 s
- state space: $s_t = (e_{p_t}, e_{v_t}, R_t, e_{w_t})$

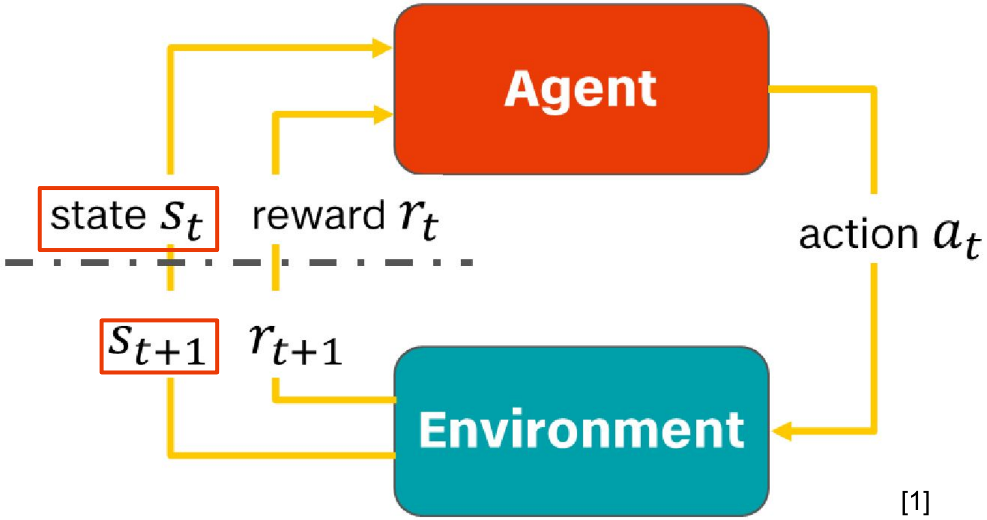
position error: $e_p \in \mathbb{R}^3$

velocity error: $e_v \in \mathbb{R}^3$

body rates error: $e_w \in \mathbb{R}^3$

flattened vector rotational matrix: $r \in \mathbb{R}^9$

$$s_t \in \mathbb{R}^{18}$$



[1]

[1] Prof. Dr.-Ing. Francisco Morales Serrano (2023) Reinforcement Learning

Formulate the problem as a reinforcement problem

Action

- Action space [-1, 1] :

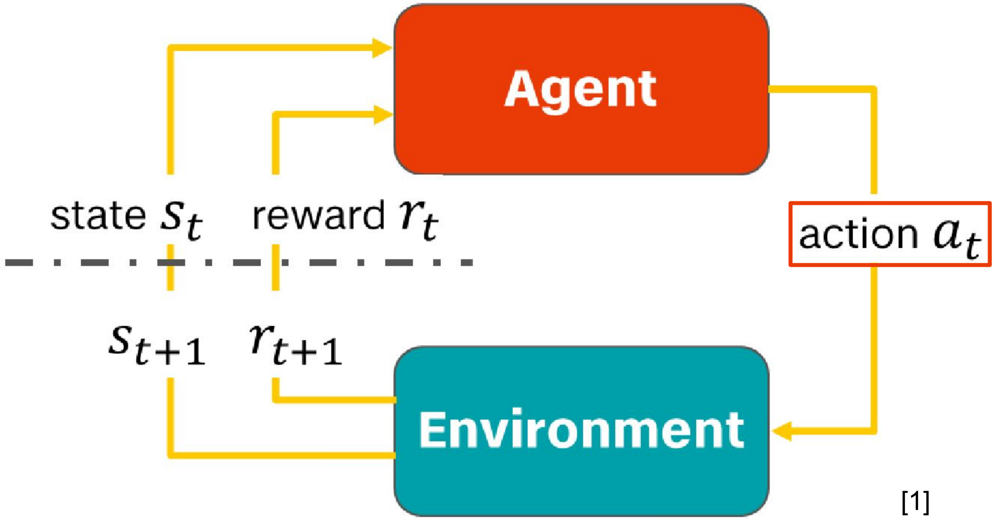
$$a_t = (F_{1_t}, F_{2_t}, F_{3_t}, F_{4_t})$$ [2]

- normalized by the hovering force F_h :

$$F_i = F_h + \frac{a_i(F_{max} - F_{min})}{2}$$ [3]

- hovering thrust:

$$F_h = \frac{mg}{4}$$ [4]



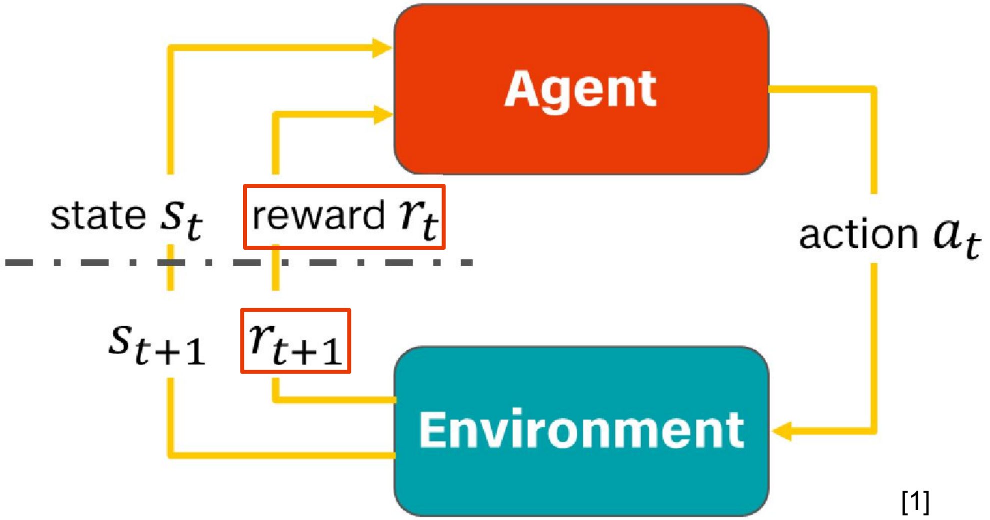
[1]

[1] Prof. Dr.-Ing. Francisco Morales Serrano (2023) Reinforcement Learning
[2-4] Deshpande, A. M., Minai, A. A., Kumar, M. (2021). Robust Deep Reinforcement Learning for Quadcopter Control

Formulate the problem as a reinforcement problem

Reward

- goal: reach a fixed waypoint (0m, 0m, 5m)
- reward for being alive
- errors are subtracted from alive reward
- yaw rotation is not relevant



[1]

$$r_t = \beta - \alpha_a \|a\|_2 - \sum_{k \in \{p,v,\omega\}} \alpha_k \|e_k\|_2 - \sum_{j \in \{\xi,\rho\}} \alpha_j \|e_j\|_2$$

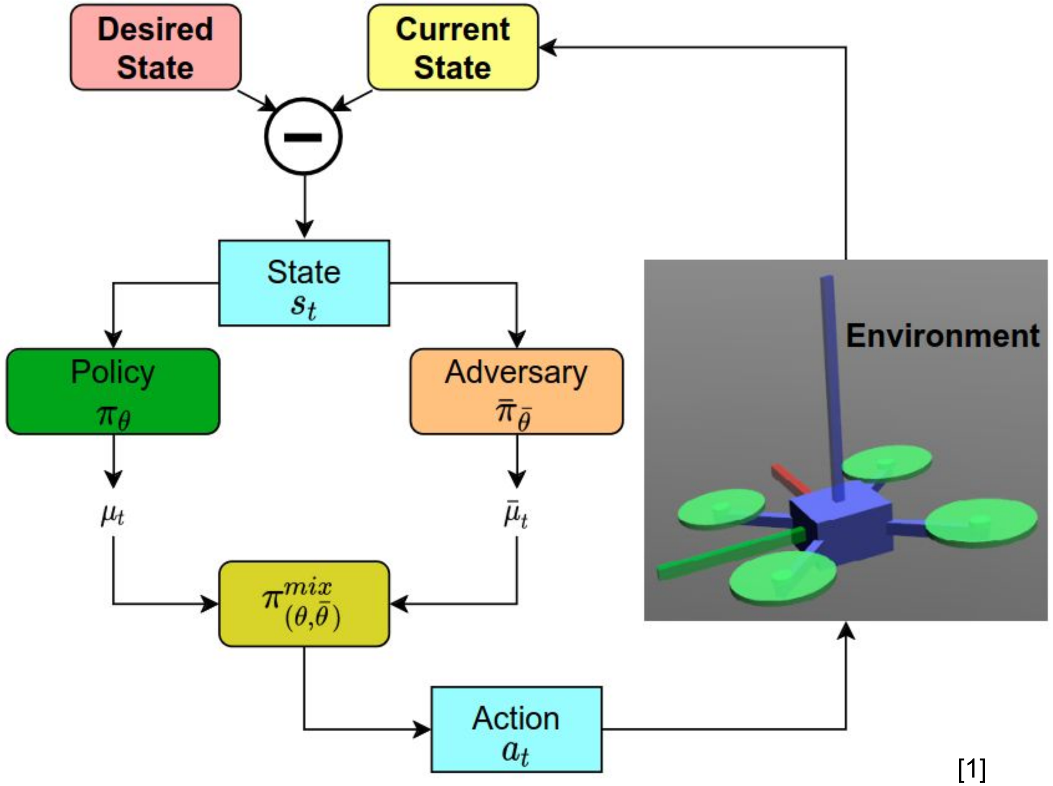
Reward Function

[2]

[1] Prof. Dr.-Ing. Francisco Morales Serrano (2023) Reinforcement Learning
[2] Deshpande, A. M., Minai, A. A., Kumar, M. (2021). Robust Deep Reinforcement Learning for Quadcopter Control

Action Robust Deep Deterministic Policy Gradient (AR-DDPG)

- combines deep learning and policy gradients (neural network is updated based on)
- can handle continuous action space
- actor-critic architecture
- replay buffer for randomly sampling experiences
- adversarial agent
- uses target networks → Off-Policy

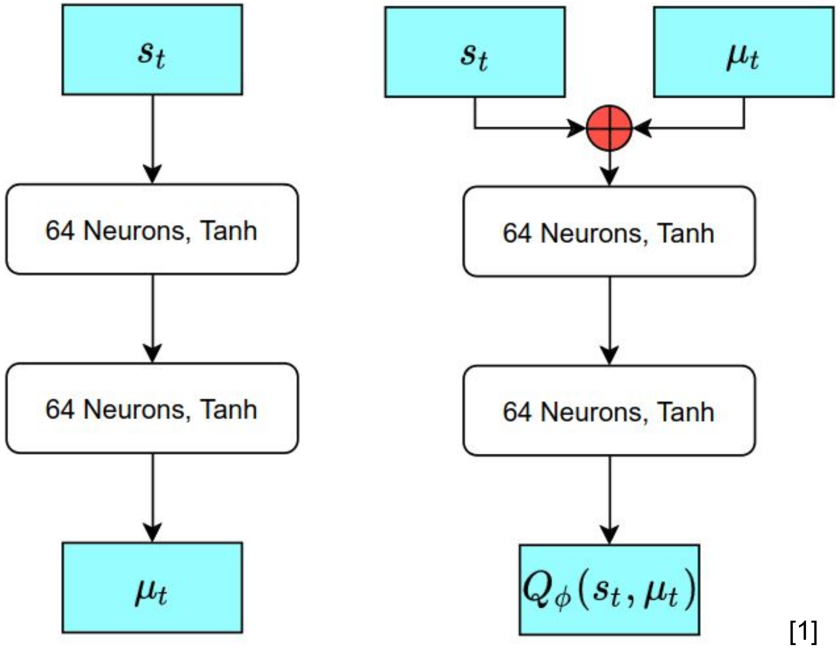


Simplified Training Model

[1] Deshpande, A. M., Minai, A. A., Kumar, M. (2021). Robust Deep Reinforcement Learning for Quadcopter Control

Training Setup

- random starting state inside 2m cube around target position
- adversary policy chosen 10% of the time + exploration noise
- time-correlated Ornstein–Uhlenbeck noise
- 2 million training iterations with max. 1500 steps each



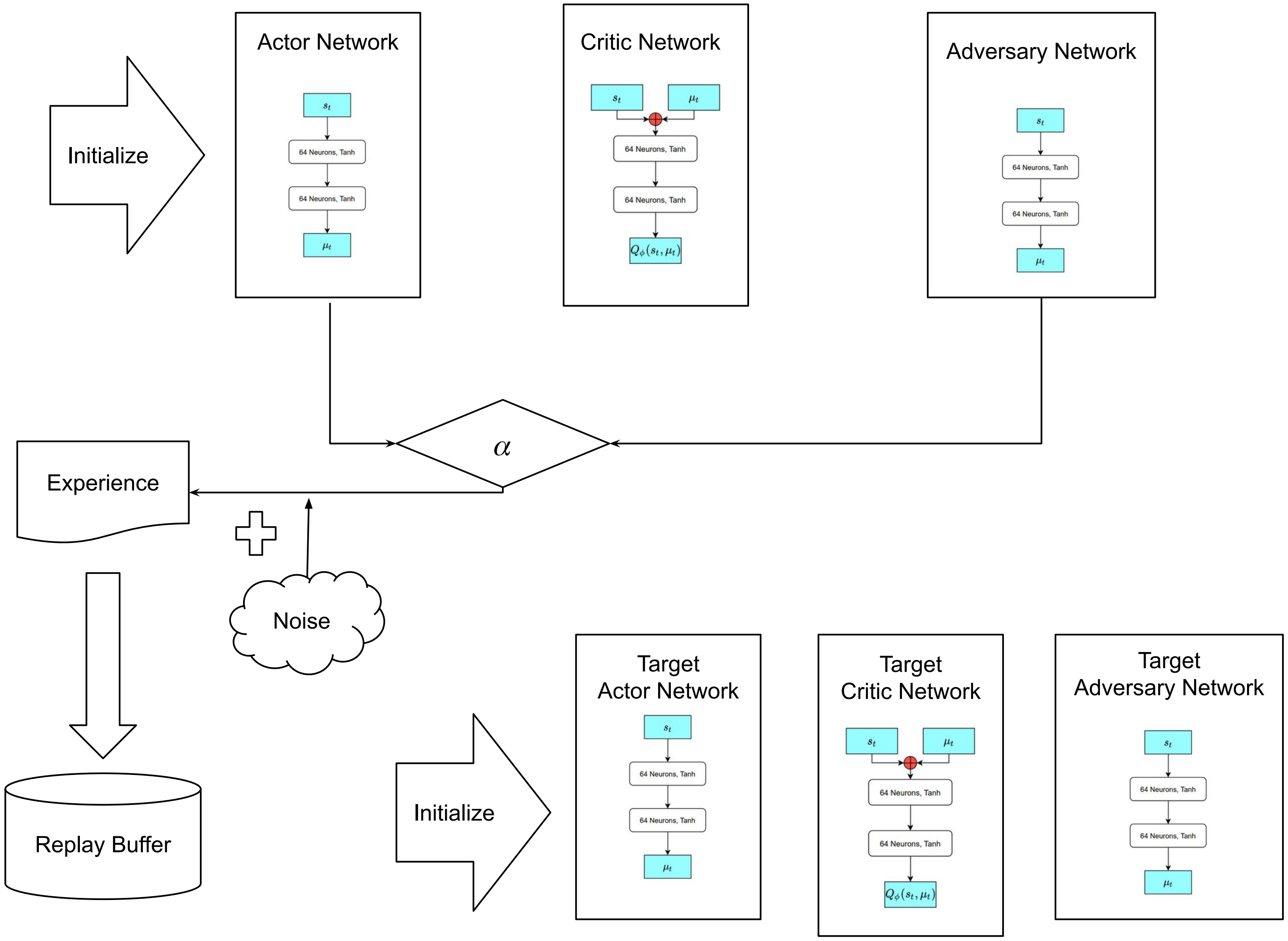
[1]

Network architecture (left: actor & adversary, right: critic)

[1] Deshpande, A. M., Minai, A. A., Kumar, M. (2021). Robust Deep Reinforcement Learning for Quadcopter Control

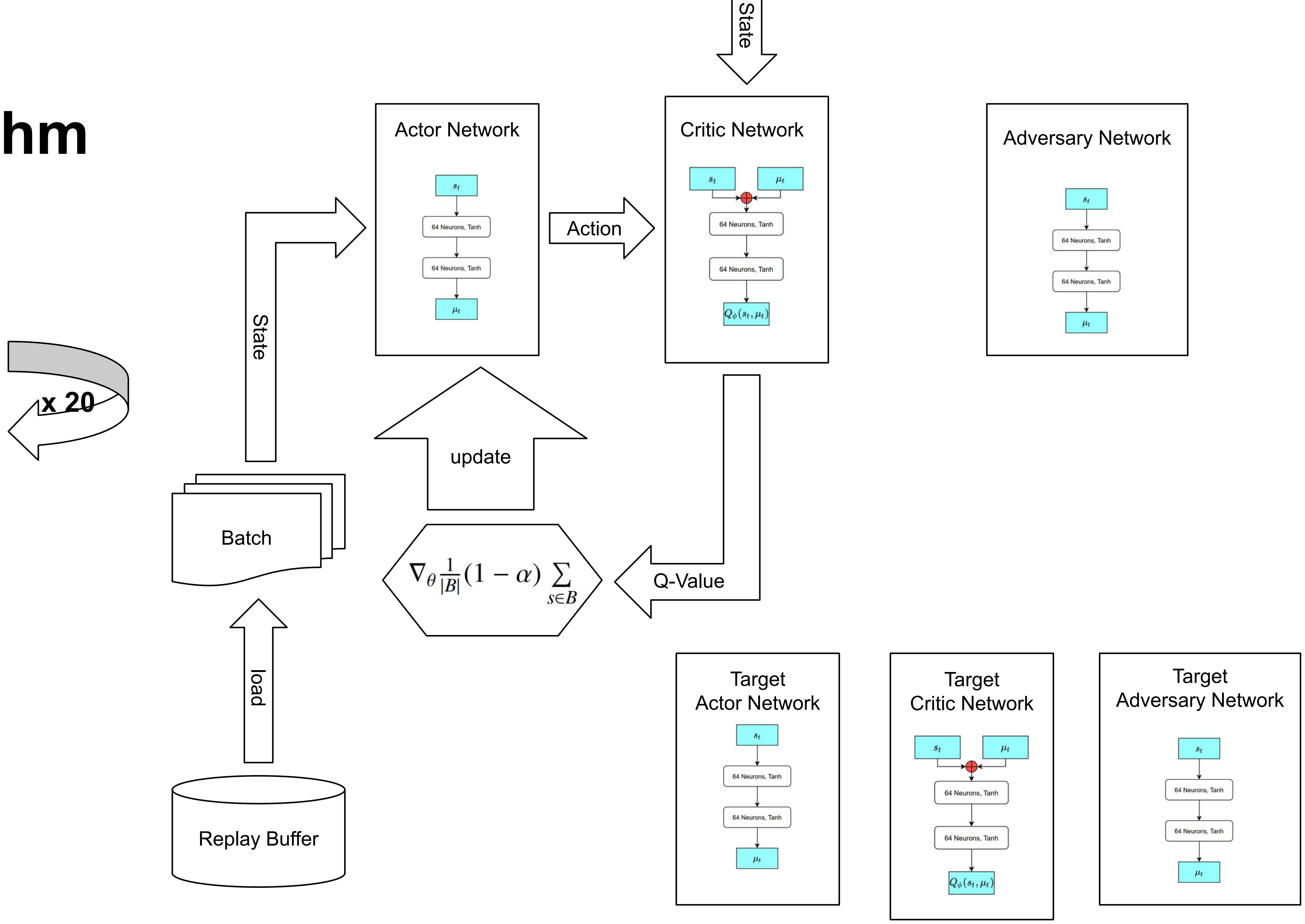
Training Algorithm

Initialization and start of episode step



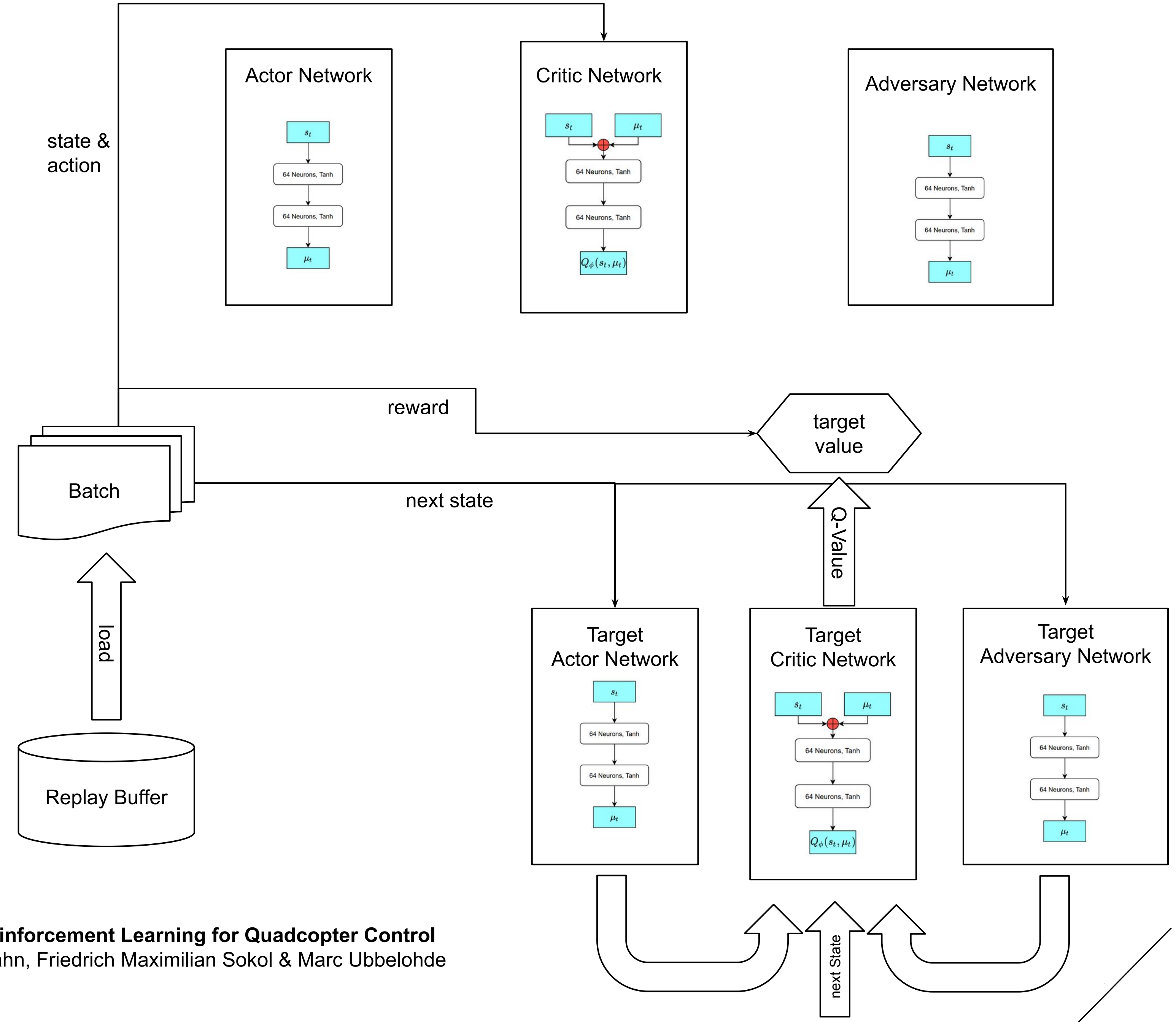
Training Algorithm

Training the Actor-Network



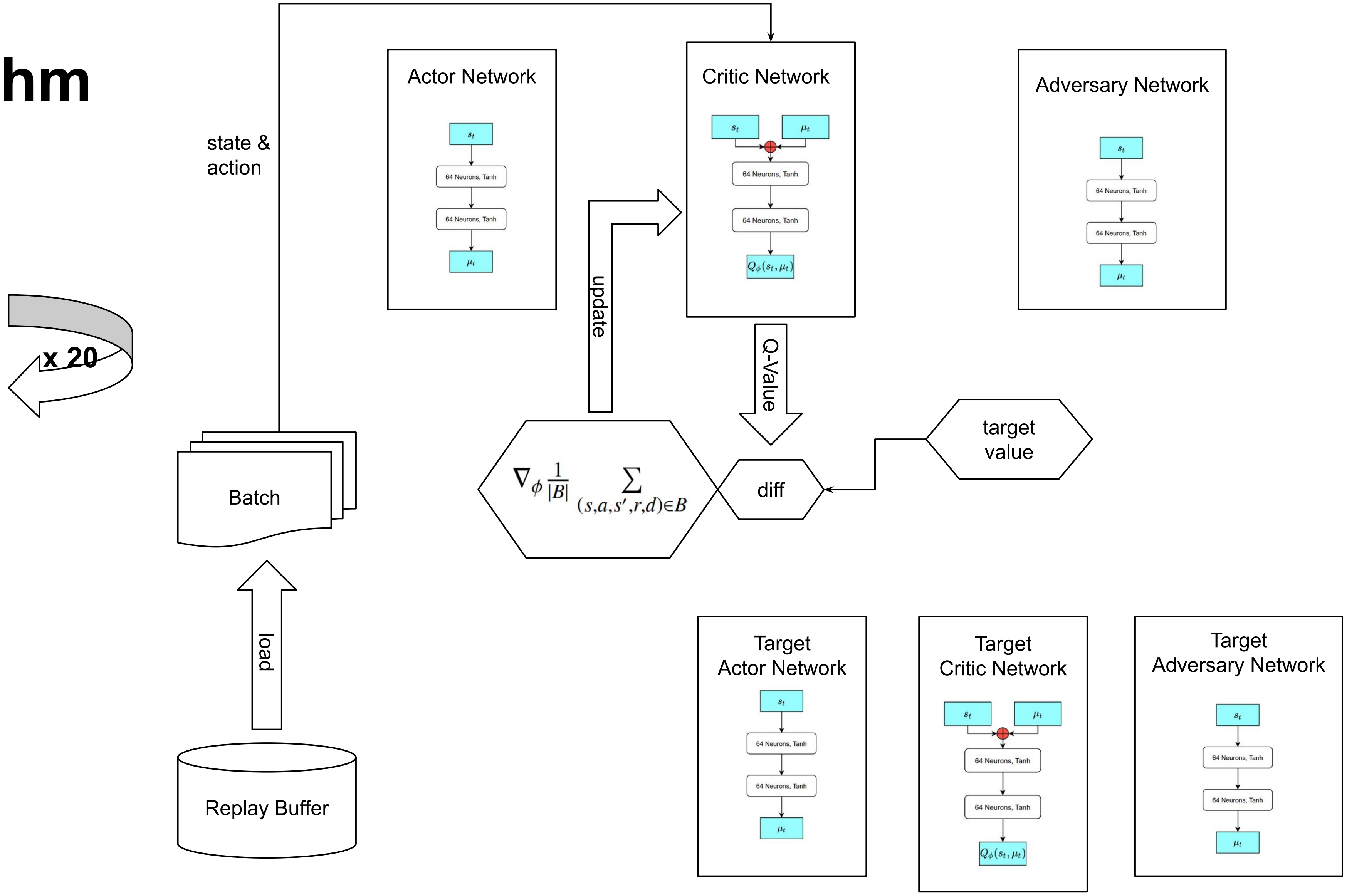
Training Algorithm

Training the Critic-Network



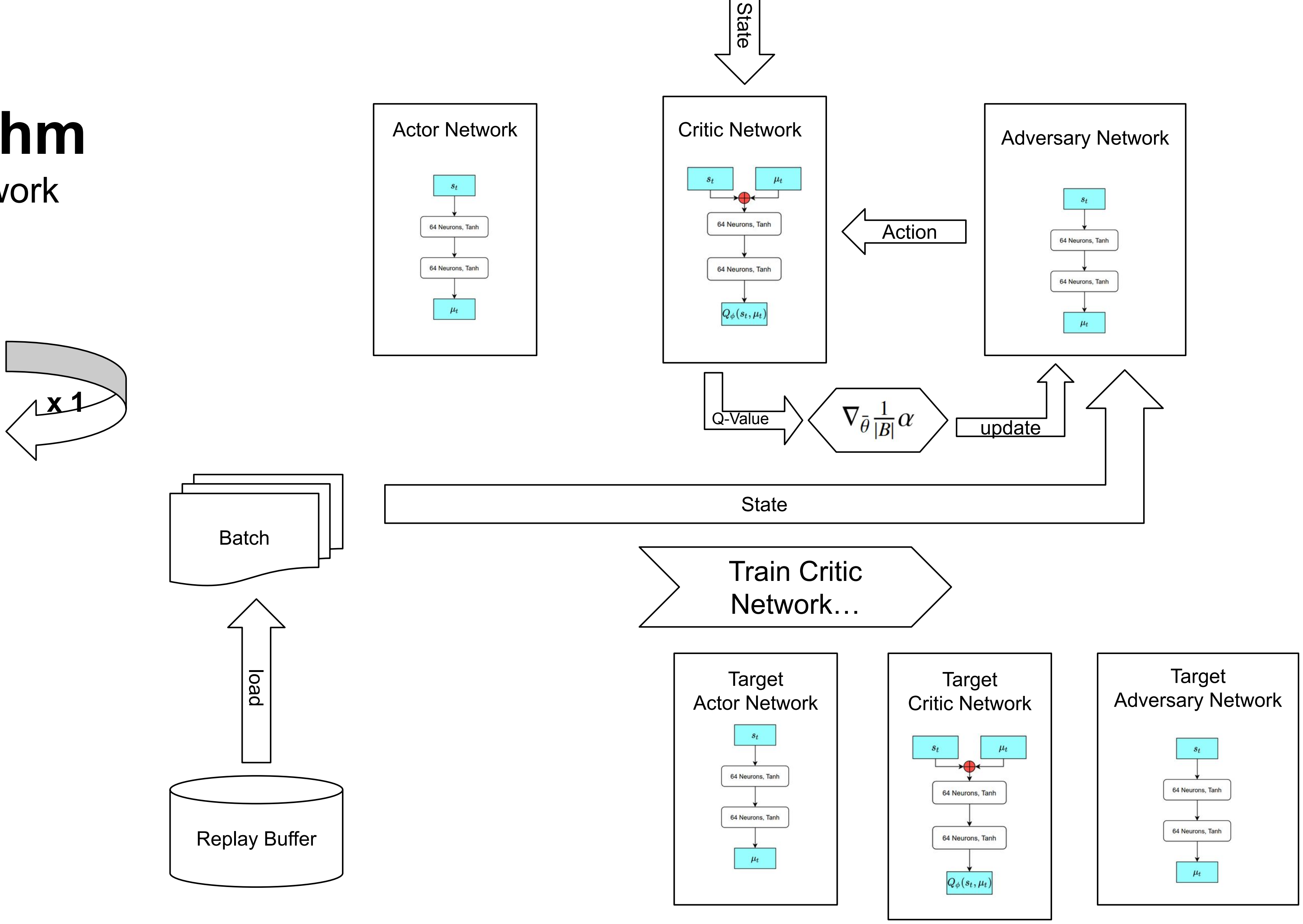
Training Algorithm

Training the Critic-Network



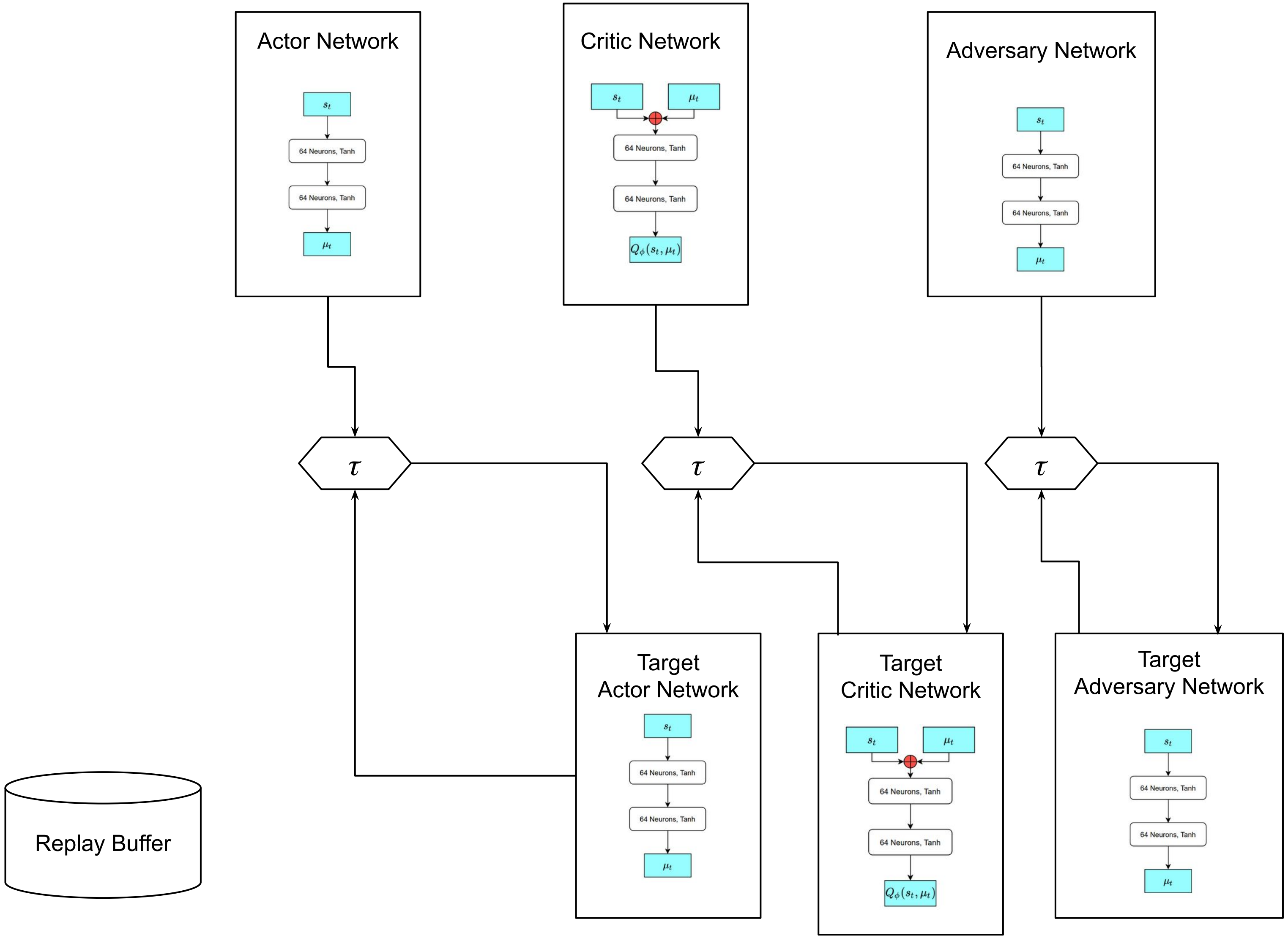
Training Algorithm

Training the Adversary-Network



Training Algorithm

Updating Target Networks



Testing Setup

- robust and non-robust network trained
- policies are no longer adjusted
- no adversary policy used
- mass and action perturbations (10 Episodes for each permutation)
- noise interpreted as model variations or turbulences

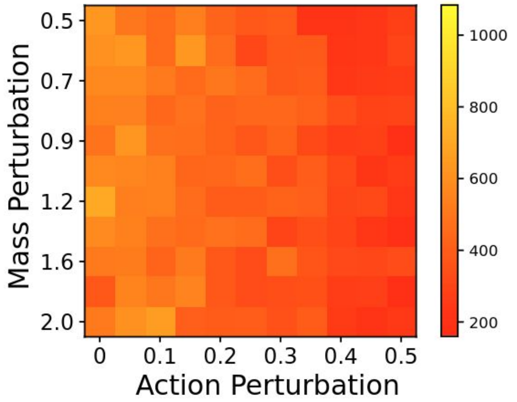
Parameter	Set of Values
Quadcopter relative mass $\frac{m_{test}}{m_{train}}$ (Mass Perturbations)	$\{0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0\}$
Probability of external perturbations δ (Action Perturbations)	$\{0, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5\}$

Test parameters [1]

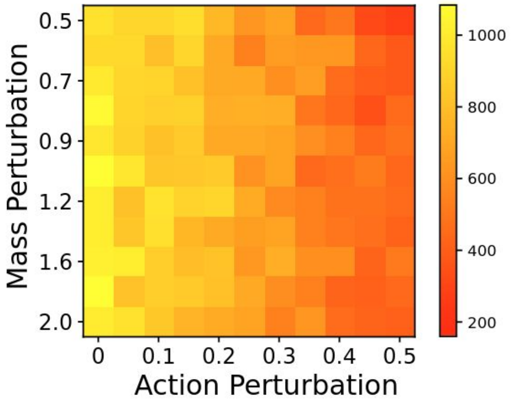
[1] Deshpande, A. M., Minai, A. A., Kumar, M. (2021). Robust Deep Reinforcement Learning for Quadcopter Control

Results

- figures of merit: average return values per permutation (10 episodes each)
- robust algorithm performs much better (even without perturbations)
- authors argue that the robust algorithms is better prepared for model uncertainties (no clear gradient recognizable)



DDPG



AR-DDPG

[1]

[1] Deshpande, A. M., Minai, A. A., Kumar, M. (2021). Robust Deep Reinforcement Learning for Quadcopter Control

Conclusion and Recommendations

- robust algorithm preferred
- big step compared to previous algorithms
- use different activation functions in neural networks
- adjust starting position (decrease size)
- stability is taken into account
- orientation was somewhat neglected

Questions?

Source: Dall-E