

# Uczenie ze wzmocnieniem - projekt

Implementacja AI do gry w Trackmanię z wykorzystaniem uczenia ze wzmocnieniem

Wojciech Klusek, Aleksander Kuś

26 stycznia 2024

## Cel Projektu:

- Stworzenie programu do gry w grę Trackmania 2020, wykorzystującego uczenie ze wzmocnieniem.
- Kryterium sukcesu projektu było skuteczne przejechanie przez pojazd sterowany programem mapy testowej.

## Trackmania 2020:

- Odsłona popularnej serii gier wyścigowych, która zyskała uznanie dzięki swojej dynamicznej rozgrywce.
- Charakteryzuje się połączeniem wyścigów z elementami platformowymi, co sprawia, że jest dużym wyzwaniem zręcznościowym.
- Jest dobrym środowiskiem do implementacji i testowania algorytmów AI, ponieważ wymaga od graczy precyzyjnego sterowania i umiejętności szybkiego podejmowania decyzji.

# Trackmania 2020



Figure: Zrzut ekranu z gry Trackmania 2020.

## Środowisko *tmrl*:

- Rozproszony framework dla robotyki, zaprojektowany, do treningu AI z wykorzystaniem uczenia ze wzmocnieniem w aplikacjach czasu rzeczywistego
- Jest dostarczany z potokiem autonomicznej jazdy dla gry wideo TrackMania 2020.
- Sztuczna inteligencja umieszczona w tym środowisku nie ma wiedzy na temat jazdy samochodem.
- Celem AI jest nauczenie się jak najszybszego ukończenia toru poprzez eksplorację otoczenia.

# Architektura środowiska tmrl

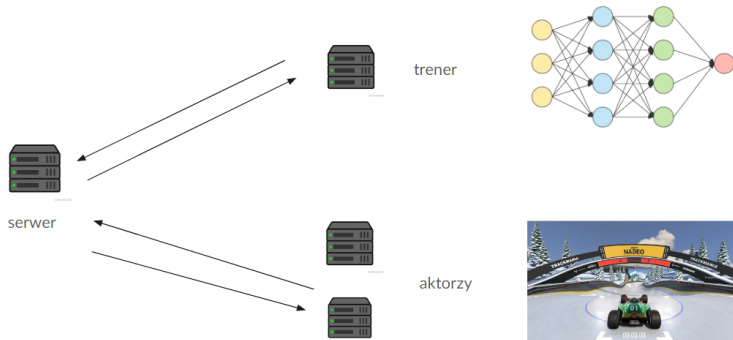


Figure: Rysunek ilustrujący architekturę środowiska *tmrl*.

# Mapa testowa

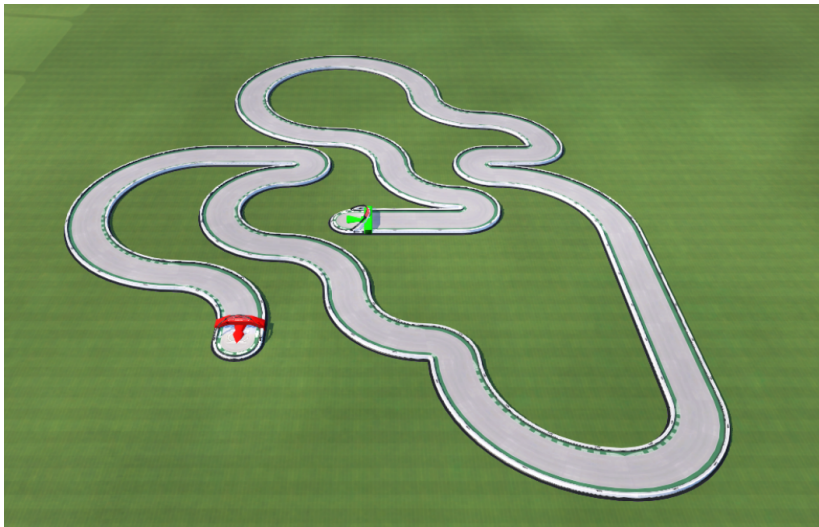


Figure: Mapa testowa dla zaimplementowanego programu.

Do implementacji programu opisanego w treści zadania zostały użyte dwa algorytmy

- 1 DDPG - Deep Deterministic Policy Gradient,
- 2 SAC - Soft Actor Critic.

Oba opisane algorytmy korzystały z dwóch sieci neuronowych:

- 1 Konwolucyjna sieć neuronowa do przetwarzania obrazów.
- 2 Wielowarstwowy perceptron do generowania akcji.

# Algorytm DDPG

DDPG jest algorytmem typu *actor-critic*, który wykorzystuje dwie główne sieci: sieć agenta  $\mu(s)$  do aproksymacji optymalnej polityki i sieć krytyka  $Q$  do oceny optymalnej wartości funkcji. Proces aktualizacji obejmuje także aktualizacje sieci docelowych -  $\mu_{\theta_{\text{targ}}}$ ,  $Q_{\phi_{\text{targ}}}$ , które są wolniejszymi wersjami głównych sieci, aby zapewnić stabilność nauki.

## Algorytm:

- ➊ Wybierz próbkę:  $B = \{(s, a, r, s', d)\}$  z  $D$
- ➋ Oblicz cel:  $y(r, s', d) = r + \gamma(1 - d)Q_{\phi_{\text{targ}}}(s', \mu_{\theta_{\text{targ}}}(s'))$
- ➌ Aktualizuj funkcję  $Q$ :  $\nabla_{\theta} \frac{1}{|B|} \sum Q_{\phi}(s, \mu_{\theta}(s))$
- ➍ Aktualizuj politykę:  $\nabla_{\theta} \frac{1}{|B|} \sum Q(s, \mu(s))$
- ➎ Aktualizuj sieci targetowe:  
 $\phi_{\text{targ}} \leftarrow \rho\phi + (1 - \rho)\phi_{\text{targ}}$   
 $\theta_{\text{targ}} \leftarrow \rho\theta + (1 - \rho)\theta_{\text{targ}}$





Figure: Aktor wykorzystujący algorytm DDPG.

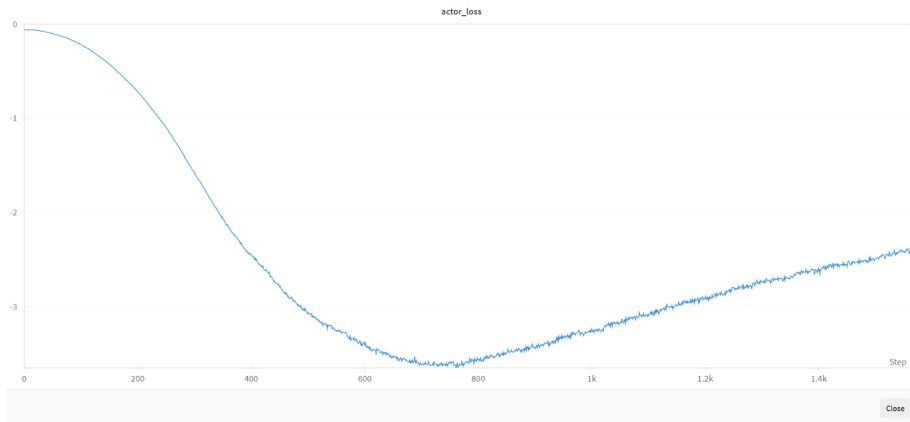


Figure: Funkcja straty aktora DDPG.

# Wyniki cd.

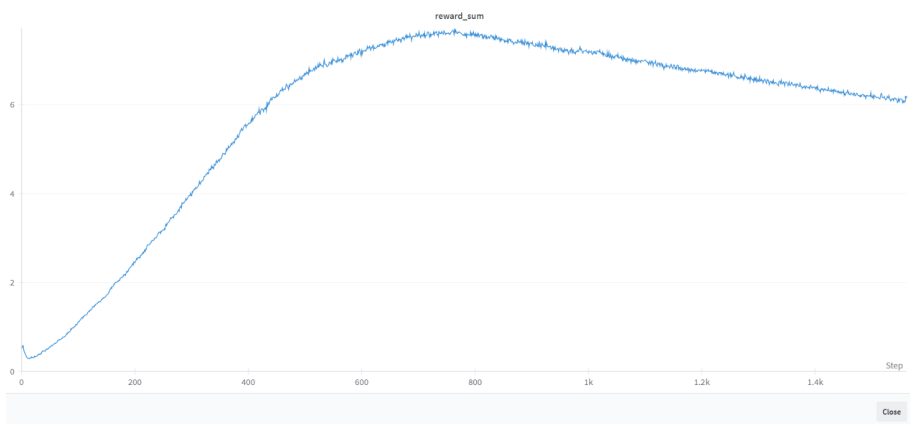


Figure: Nagma DDPG.

# Algorytm SAC

Algorytm Soft Actor-Critic (SAC) jest zaawansowaną metodą uczenia ze wzmocnieniem, która stosuje podejście aktor-krytyk z dodatkiem entropii do polityki, promując tym samym eksplorację.

❶ Wybierz próbkę:  $B = \{(s, a, r, s', d)\}$  z  $D$

❷ Oblicz cel:

$$y(r, s', d) = r + \gamma(1 - d)(\min_{i=1,2} Q_{\phi_{\text{targ},i}}(s', \tilde{a}') - \alpha \log \tilde{\pi}_{\theta}(\tilde{a}'|s'))$$

❸ Aktualizuj funkcję Q:  $\nabla_{\phi_i} \frac{1}{|B|} \sum (Q_{\phi_i}(s, a) - y(r, s', d))^2$  dla  $i = 1, 2$

❹ Aktualizuj politykę:  $\nabla_{\theta} \frac{1}{|B|} \sum (\min_{i=1,2} Q_{\phi_i}(s, \tilde{a}(s)) - \alpha \log \pi_{\theta}(\tilde{a}(s)|s))$

❺ Aktualizuj sieci targetowe:  $\phi_{\text{targ},i} \leftarrow \rho \phi_i + (1 - \rho) \phi_{\text{targ},i}$  dla  $i = 1, 2$

Przejazd agenta SAC

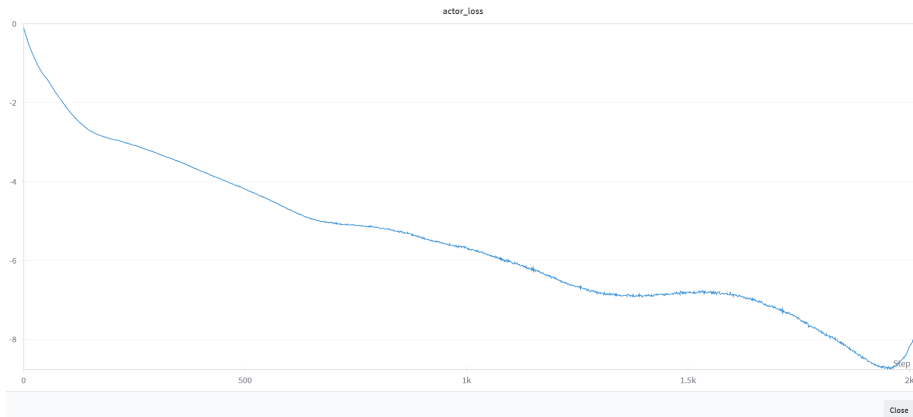


Figure: Funkcja straty aktora SAC.

# Wyniki cd.

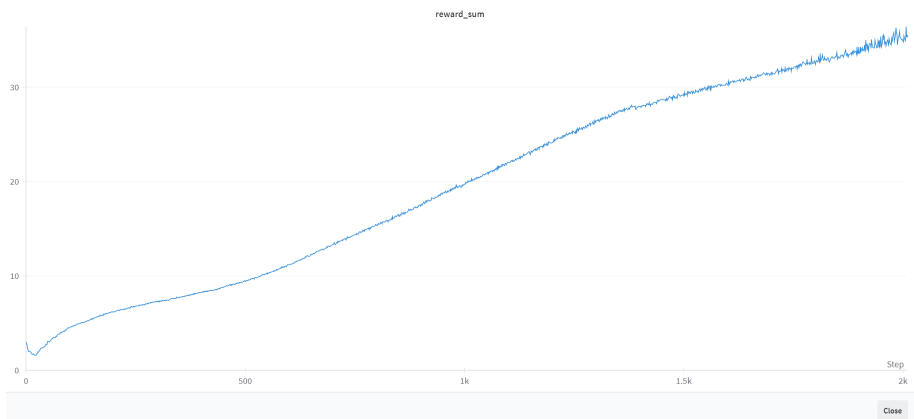


Figure: Nagroda SAC.

Podczas pracy nad projektem wyciągnięto następujące wnioski:

- ❶ Uczenie ze wzmocnieniem może być skutecznie stosowane do złożonych zadań wymagających precyzyjnej kontroli i szybkiego podejmowania decyzji.
- ❷ W kontekście zadanego problemu algorytm SAC poradził sobie zdecydowanie lepiej niż algorytm DDPG.
- ❸ Długi czas treningu oraz niewygodne identyfikowanie błędów na GPU stanowią istotne przeszkody w szybkim iteracyjnym rozwoju algorytmów.



- ① DDPG:  
<https://spinningup.openai.com/en/latest/algorithms/ddpg.html>
- ② SAC:  
<https://spinningup.openai.com/en/latest/algorithms/sac.html>
- ③ Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018.
- ④ N. Mazyavkina, Samir Moustafa, Ilya Trofimov, and Evgeny Burnaev. Optimizing the Neural Architecture of Reinforcement Learning Agents, pages 591–606. 07 2021.