Reinforcement Learning for Navigation Goal

Francesco Fantechi

Relatori: Andrew D. Bagdanov

UNIVERSITA' DEGLI STUDI DI FIRENZE
Facolta di Ingegneria
Corso di Laurea Magistrale in Ingegneria Informatica

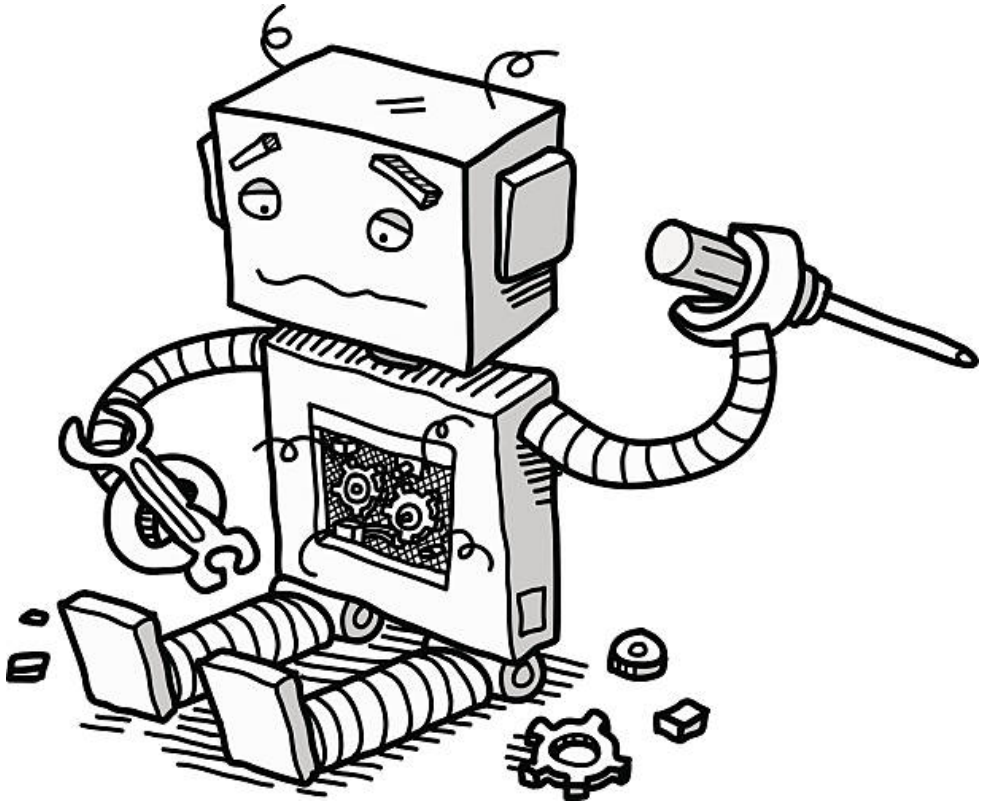
A.A. 2022-2023

Obbiettivo

- Far navigare il robot fino a un punto Goal

Obbiettivo

- Far navigare il robot fino a un punto Goal
- Evitando le collisioni con gli ostacoli

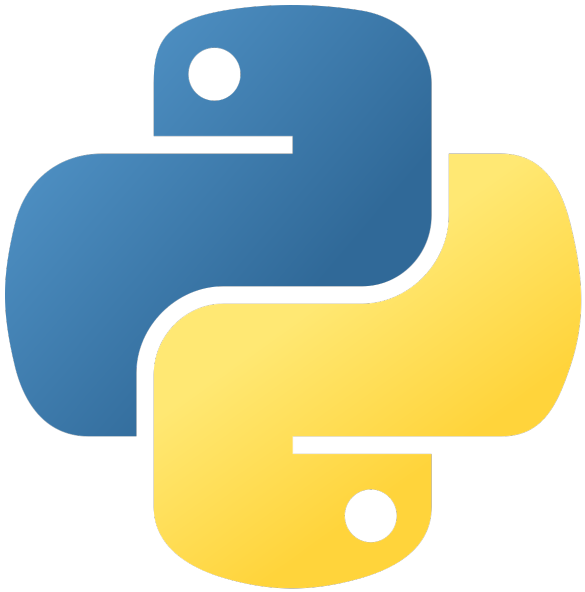


Obbiettivo

- Far navigare il robot fino a un punto Goal
- Evitando le collisioni con gli ostacoli
- Imparando e migliorando dai propri tentativi attraverso un algoritmo di Reinforcement e Curriculum Learning

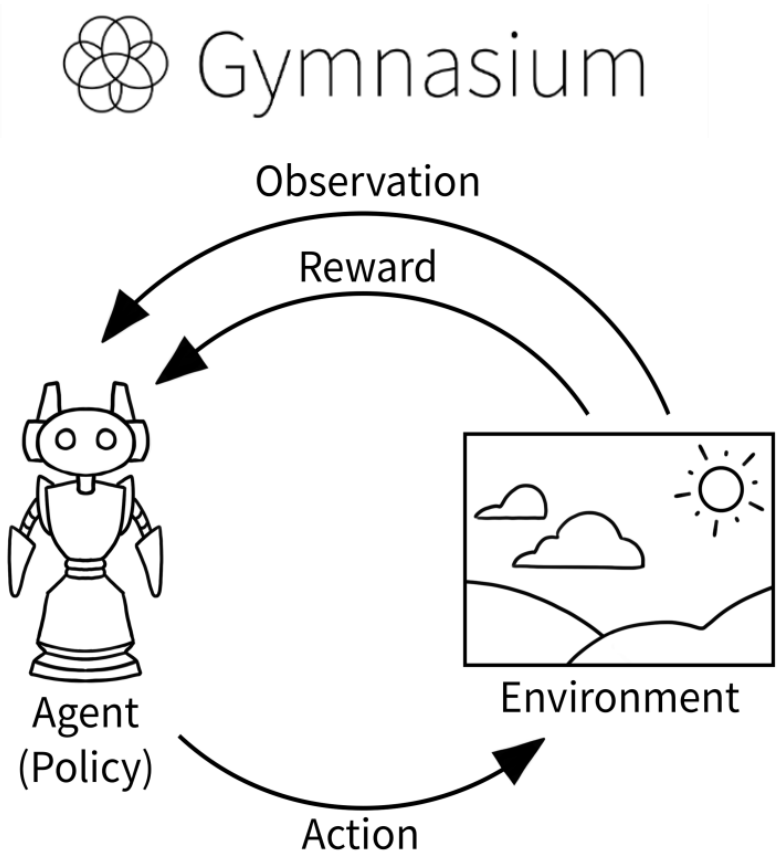
Implementazione

- Il lavoro è stato implementato in linguaggio Python



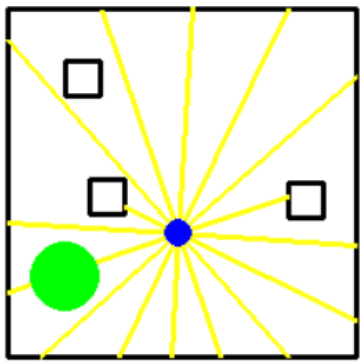
Implementazione

- Il lavoro è stato implementato in linguaggio Python
- Libreria Gymnasium per implementare il ciclo osservazione, stato, azione, reward dell'agente



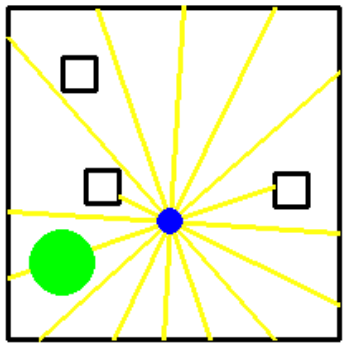
Implementazione

- Il lavoro è stato implementato in linguaggio Python
- Libreria Gymnasium per implementare il ciclo osservazione, stato, azione, reward dell'agente
- Libreria Pygame per la renderizzazione a video



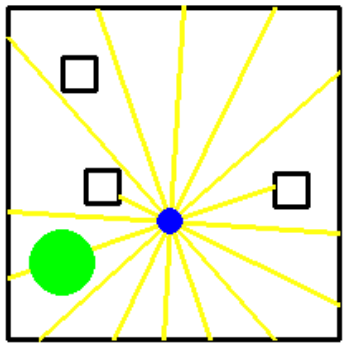
implementazione

- L'Osservazione dell'agente è un vettore costituito da 18 elementi:



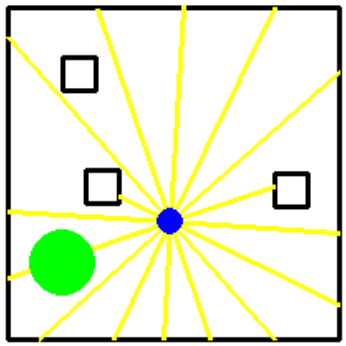
implementazione

- L'Osservazione dell'agente è un vettore costituito da 18 elementi:
  - Le 16 minor distanze dagli ostacoli circostanti in direzione radiale all'agente



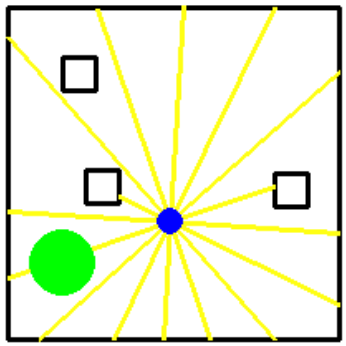
implementazione

- L'Osservazione dell'agente è un vettore costituito da 18 elementi:
  - Le 16 minor distanze dagli ostacoli circostanti in direzione radiale all'agente
  - La distanza che intercorre fra l'agente e il punto Goal



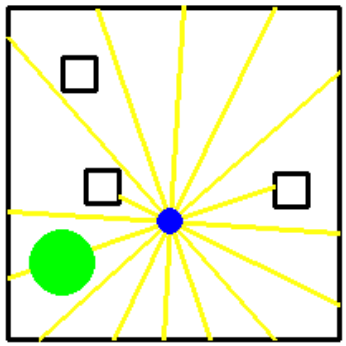
implementazione

- L'Osservazione dell'agente è un vettore costituito da 18 elementi:
  - Le 16 minor distanze dagli ostacoli circostanti in direzione radiale all'agente
  - La distanza che intercorre fra l'agente e il punto Goal
  - La distanza angolare fra l'orientazione dell'agente e la direzione che dovrebbe assumere per frontaggiare il punto Goal



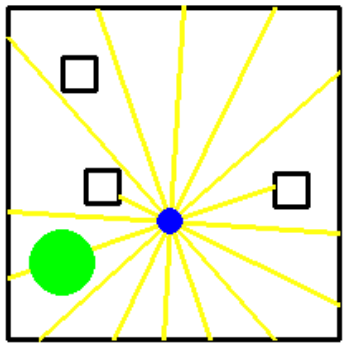
implementazione

- L'Osservazione dell'agente è un vettore costituito da 18 elementi:
  - Le 16 minor distanze dagli ostacoli circostanti in direzione radiale all'agente
  - La distanza che intercorre fra l'agente e il punto Goal
  - La distanza angolare fra l'orientazione dell'agente e la direzione che dovrebbe assumere per frontaggiare il punto Goal
- L'agente puó compiere 3 Azioni: $\uparrow$, $\nwarrow$, $\nearrow$



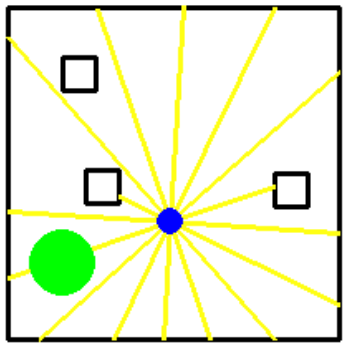
implementazione

- L'Osservazione dell'agente è un vettore costituito da 18 elementi:
  - Le 16 minor distanze dagli ostacoli circostanti in direzione radiale all'agente
  - La distanza che intercorre fra l'agente e il punto Goal
  - La distanza angolare fra l'orientazione dell'agente e la direzione che dovrebbe assumere per frontaggiare il punto Goal
- L'agente puó compiere 3 Azioni: $\uparrow$, $\nwarrow$, $\nearrow$
- Reward dell'agente:



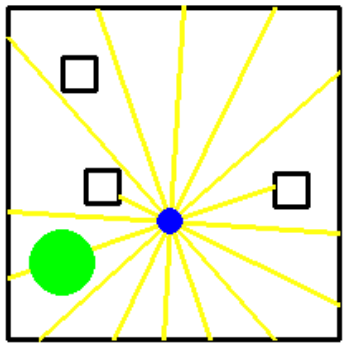
implementazione

- L'Osservazione dell'agente è un vettore costituito da 18 elementi:
  - Le 16 minor distanze dagli ostacoli circostanti in direzione radiale all'agente
  - La distanza che intercorre fra l'agente e il punto Goal
  - La distanza angolare fra l'orientazione dell'agente e la direzione che dovrebbe assumere per frontaggiare il punto Goal
- L'agente puó compiere 3 Azioni: $\uparrow$, $\nwarrow$, $\nearrow$
- Reward dell'agente:
  - $+100 \cdot n$ se raggiunge il Goal ($n \in \mathbb{R}$, $n \geq 1$)



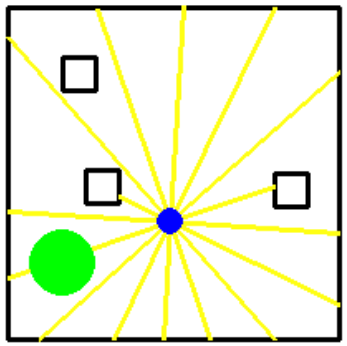
implementazione

- L'Osservazione dell'agente è un vettore costituito da 18 elementi:
  - Le 16 minor distanze dagli ostacoli circostanti in direzione radiale all'agente
  - La distanza che intercorre fra l'agente e il punto Goal
  - La distanza angolare fra l'orientazione dell'agente e la direzione che dovrebbe assumere per frontaggiare il punto Goal
- L'agente puó compiere 3 Azioni: $\uparrow$, $\nwarrow$, $\nearrow$
- Reward dell'agente:
  - $+100 \cdot n$ se raggiunge il Goal ($n \in \mathbb{R}$, $n \geq 1$)
  - $+2 \cdot \Delta$ se avanza verso il Goal ($\Delta \in [0, 1] \ \propto \ \text{avanzamento}$)



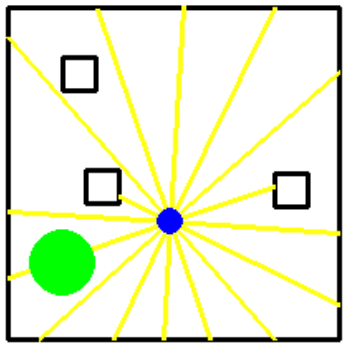
implementazione

- L'Osservazione dell'agente è un vettore costituito da 18 elementi:
  - Le 16 minor distanze dagli ostacoli circostanti in direzione radiale all'agente
  - La distanza che intercorre fra l'agente e il punto Goal
  - La distanza angolare fra l'orientazione dell'agente e la direzione che dovrebbe assumere per frontaggiare il punto Goal
- L'agente puó compiere 3 Azioni: $\uparrow$, $\nwarrow$, $\nearrow$
- Reward dell'agente:
  - $+100 \cdot n$ se raggiunge il Goal ($n \in \mathbb{R},\ n \geq 1$)
  - $+2 \cdot \Delta$ se avanza verso il Goal ($\Delta \in [0, 1] \ \propto \ \text{avanzamento}$)
  - $-1 \cdot \Delta$ se si allontana dal Goal ($\Delta \in [0, 1] \ \propto \ \text{allontanamento}$)



implementazione

- L'Osservazione dell'agente è un vettore costituito da 18 elementi:
  - Le 16 minor distanze dagli ostacoli circostanti in direzione radiale all'agente
  - La distanza che intercorre fra l'agente e il punto Goal
  - La distanza angolare fra l'orientazione dell'agente e la direzione che dovrebbe assumere per frontaggiare il punto Goal
- L'agente puó compiere 3 Azioni: $\uparrow$, $\nwarrow$, $\nearrow$
- Reward dell'agente:
  - $+100 \cdot n$ se raggiunge il Goal ($n \in \mathbb{R},\ n \geq 1$)
  - $+2 \cdot \Delta$ se avanza verso il Goal ($\Delta \in [0, 1] \ \propto \ \text{avanzamento}$)
  - $-1 \cdot \Delta$ se si allontana dal Goal ($\Delta \in [0, 1] \ \propto \ \text{allontanamento}$)
  - -100 se collide con un ostacolo o se non termina in MaxSteps



Deep Q-Learning Algorithm

- Equazione di Ottimalità di Bellman per l'action-value function Q:

$$Q^o(x,u) = r(x,u) + \alpha \cdot \sum_{x' \in X} \varphi(x' \mid x,u) \cdot \max_{u'}(Q^o(x',u'))$$

x $\in$ X stato, u $\in$ U azione, φ valore atteso transizione, r funzione di reward, α fattore di sconto

Deep Q-Learning Algorithm

- Equazione di Ottimalità di Bellman per l'action-value function Q:

$$Q^o(x,u) = r(x,u) + \alpha \cdot \sum_{x' \in X} \varphi(x' \mid x,u) \cdot \max_{u'}(Q^o(x',u'))$$

x $\in$ X stato, u $\in$ U azione, φ valore atteso transizione, r funzione di reward, α fattore di sconto

- Approssimando in modo parametrico l'action-value function e rendendo l'equazione model-free e tempo reale si ottiene:

$$Q_t(x,u) = Q(x,u,w(t))$$

w(t) pesi delle rete neurale al tempo t

$$\widehat{Q}_t = r(t) + \alpha \cdot \max_{u'}(Q(x(t+1),u',w(t))$$

Deep Q-Learning Algorithm

- Equazione di Ottimalità di Bellman per l'action-value function Q:

$$Q^o(x,u) = r(x,u) + \alpha \cdot \sum_{x' \in X} \varphi(x' \mid x,u) \cdot \max_{u'}(Q^o(x',u'))$$

x $\in$ X stato, u $\in$ U azione, φ valore atteso transizione, r funzione di reward, α fattore di sconto

- Approssimando in modo parametrico l'action-value function e rendendo l'equazione model-free e tempo reale si ottiene:

$$Q_t(x,u) = Q(x,u,w(t))$$

w(t) pesi delle rete neurale al tempo t

$$\widehat{Q}_t = r(t) + \alpha \cdot \max_{u'}(Q(x(t+1),u',w(t))$$

- Quindi il nostro agente imparerá andando a minimizzare il proprio erroe quadratico di predizione:

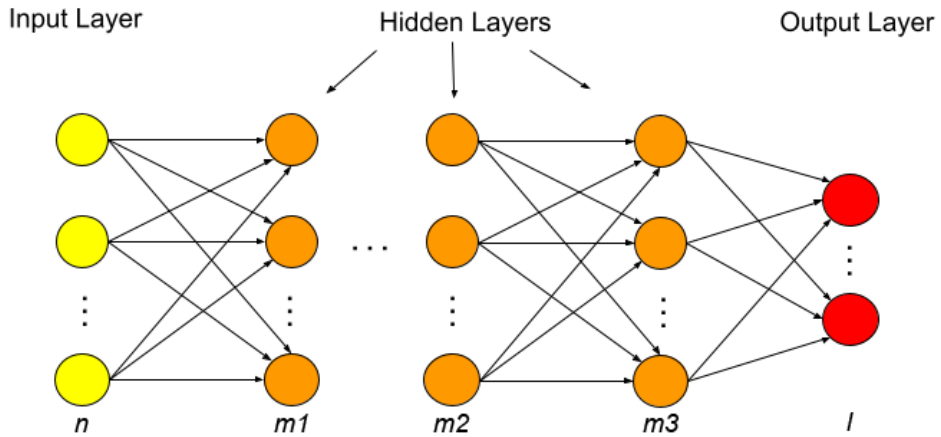
$$[Q(x(t),u(t),w) - \widehat{Q}_t]^2$$

Rete Neurale

- Libreria PyTorch

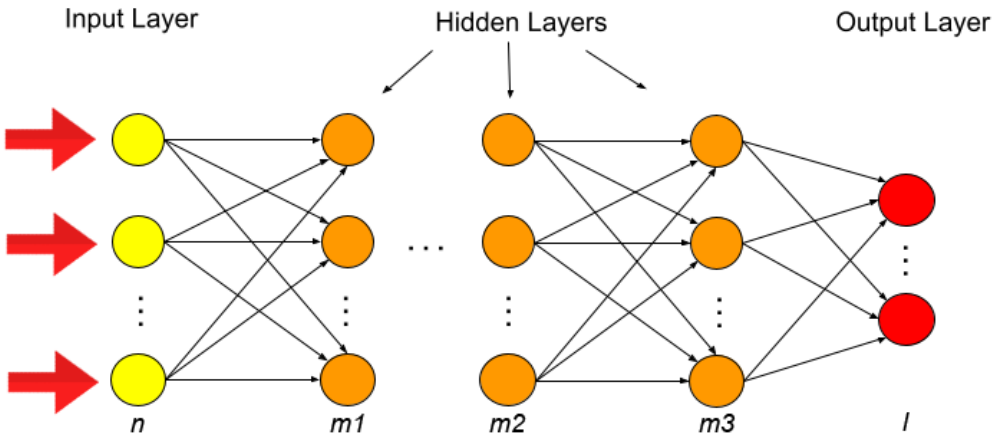
Rete Neurale

- Libreria PyTorch
- Multi layer perceptron (MLP):



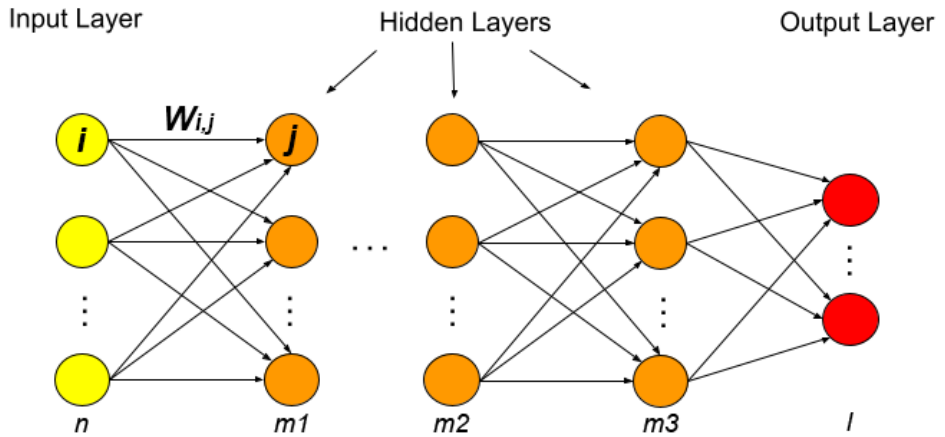
Rete Neurale

- Libreria PyTorch
- Multi layer perceptron (MLP):
  - 18 nodi nell'input layer corrispondenti all'osservazione dell'agente



Rete Neurale

- Libreria PyTorch
- Multi layer perceptron (MLP):
  - 18 nodi nell'input layer corrispondenti all'osservazione dell'agente
  - 2 hidden layers di 128 e 64 nodi rispettivamente

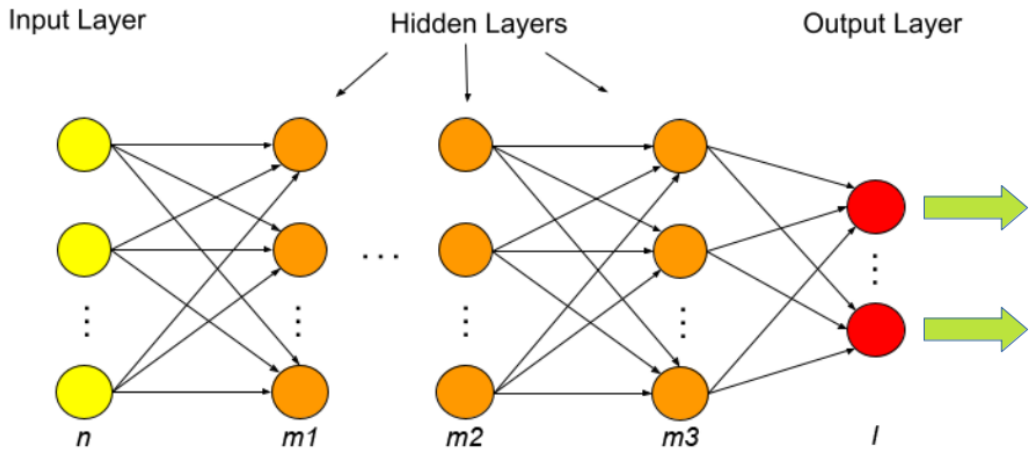


$$\alpha_j = g(\sum_{i=0}^{N} W_{i,j}\alpha_i)$$

con g funzione di attivazione e α_k output del nodo k-esimo

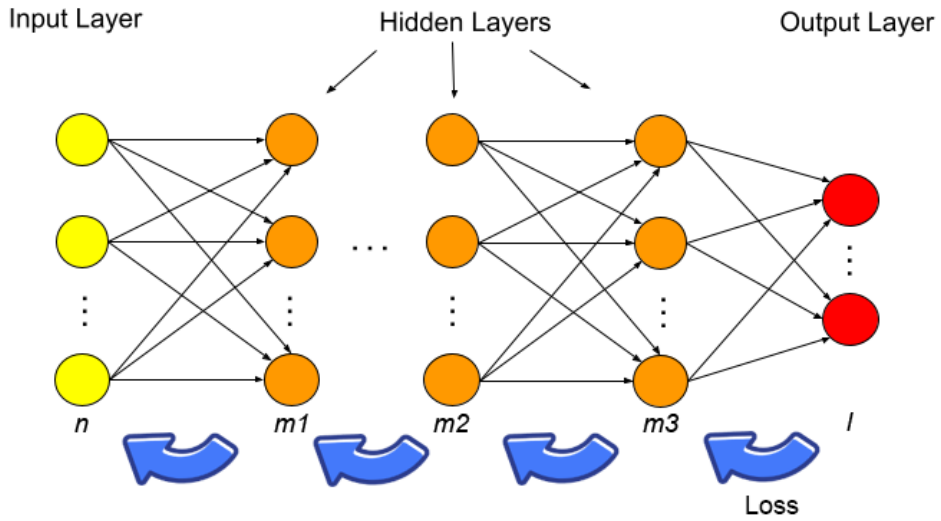
Rete Neurale

- Libreria PyTorch
- Multi layer perceptron (MLP):
  - 18 nodi nell'input layer corrispondenti all'osservazione dell'agente
  - 2 hidden layers di 128 e 64 nodi rispettivamente
  - 3 nodi nell'output layer per restituire un punteggio associato ad ogni azione



Rete Neurale

- Libreria PyTorch
- Multi layer perceptron (MLP):
  - 18 nodi nell'input layer corrispondenti all'osservazione dell'agente
  - 2 hidden layers di 128 e 64 nodi rispettivamente
  - 3 nodi nell'output layer per restituire un punteggio associato ad ogni azione
  - Ottimizzatore Adam



Retropropagazione errore e aggiornamento dei pesi

Addestramento

- L'agente è stato addestrato su migliaia di episodi variando nel corso dell'addestramento i vari parametri del modello e il numero di ostacoli presenti nell'ambiente

Addestramento

- L'agente è stato addestrato su migliaia di episodi variando nel corso dell'addestramento i vari parametri del modello e il numero di ostacoli presenti nell'ambiente
- Per favorire la convergenza:

Addestramento

- L'agente è stato addestrato su migliaia di episodi variando nel corso dell'addestramento i vari parametri del modello e il numero di ostacoli presenti nell'ambiente
- Per favorire la convergenza:
  - Exploration VS Exploitation:

Addestramento

- L'agente è stato addestrato su migliaia di episodi variando nel corso dell'addestramento i vari parametri del modello e il numero di ostacoli presenti nell'ambiente
- Per favorire la convergenza:
  - Exploration VS Exploitation:
    - Metodo $\varepsilon - \textit{Greedy}$

Addestramento

- L'agente è stato addestrato su migliaia di episodi variando nel corso dell'addestramento i vari parametri del modello e il numero di ostacoli presenti nell'ambiente
- Per favorire la convergenza:
  - Exploration VS Exploitation:
    - Metodo $\varepsilon - \textit{Greedy}$
    - ε si riduce esponenzialmente con il passare degli episodi facendo si che l'agente consolidi ció che ha imparato

Addestramento

- L'agente è stato addestrato su migliaia di episodi variando nel corso dell'addestramento i vari parametri del modello e il numero di ostacoli presenti nell'ambiente
- Per favorire la convergenza:
  - Exploration VS Exploitation:
    - Metodo $\varepsilon - \textit{Greedy}$
    - ε si riduce esponenzialmente con il passare degli episodi facendo si che l'agente consolidi ció che ha imparato
  - Replay Buffer per contenere gli esempi dai quali apprendere di modo da renderli indipendenti

Addestramento

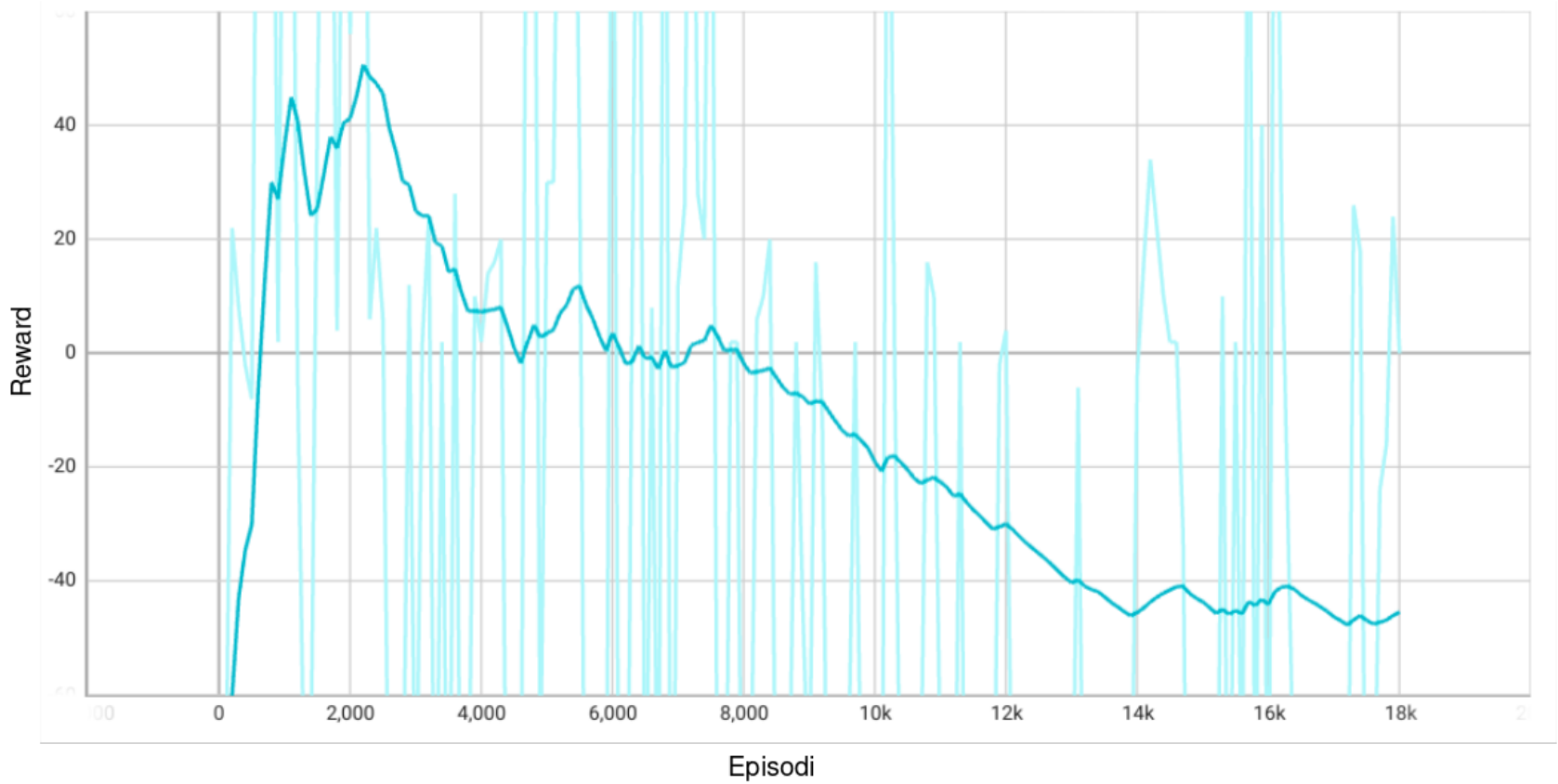
- L'agente è stato addestrato su migliaia di episodi variando nel corso dell'addestramento i vari parametri del modello e il numero di ostacoli presenti nell'ambiente
- Per favorire la convergenza:
  - Exploration VS Exploitation:
    - Metodo $\varepsilon - \textit{Greedy}$
    - ε si riduce esponenzialmente con il passare degli episodi facendo si che l'agente consolidi ció che ha imparato
  - Replay Buffer per contenere gli esempi dai quali apprendere di modo da renderli indipendenti
  - Utilizzo di due reti (Policy e Target) di modo da aggiornare tutti i pesi contemporaneamente e non solo per la coppia stato/azione corrente

Addestramento

- L'agente è stato addestrato su migliaia di episodi variando nel corso dell'addestramento i vari parametri del modello e il numero di ostacoli presenti nell'ambiente
- Per favorire la convergenza:
  - Exploration VS Exploitation:
    - Metodo $\varepsilon - \textit{Greedy}$
    - ε si riduce esponenzialmente con il passare degli episodi facendo si che l'agente consolidi ció che ha imparato
  - Replay Buffer per contenere gli esempi dai quali apprendere di modo da renderli indipendenti
  - Utilizzo di due reti (Policy e Target) di modo da aggiornare tutti i pesi contemporaneamente e non solo per la coppia stato/azione corrente
- Utilizzo della libreria Tensorboard per validare l'addestramento ogni 50 episodi

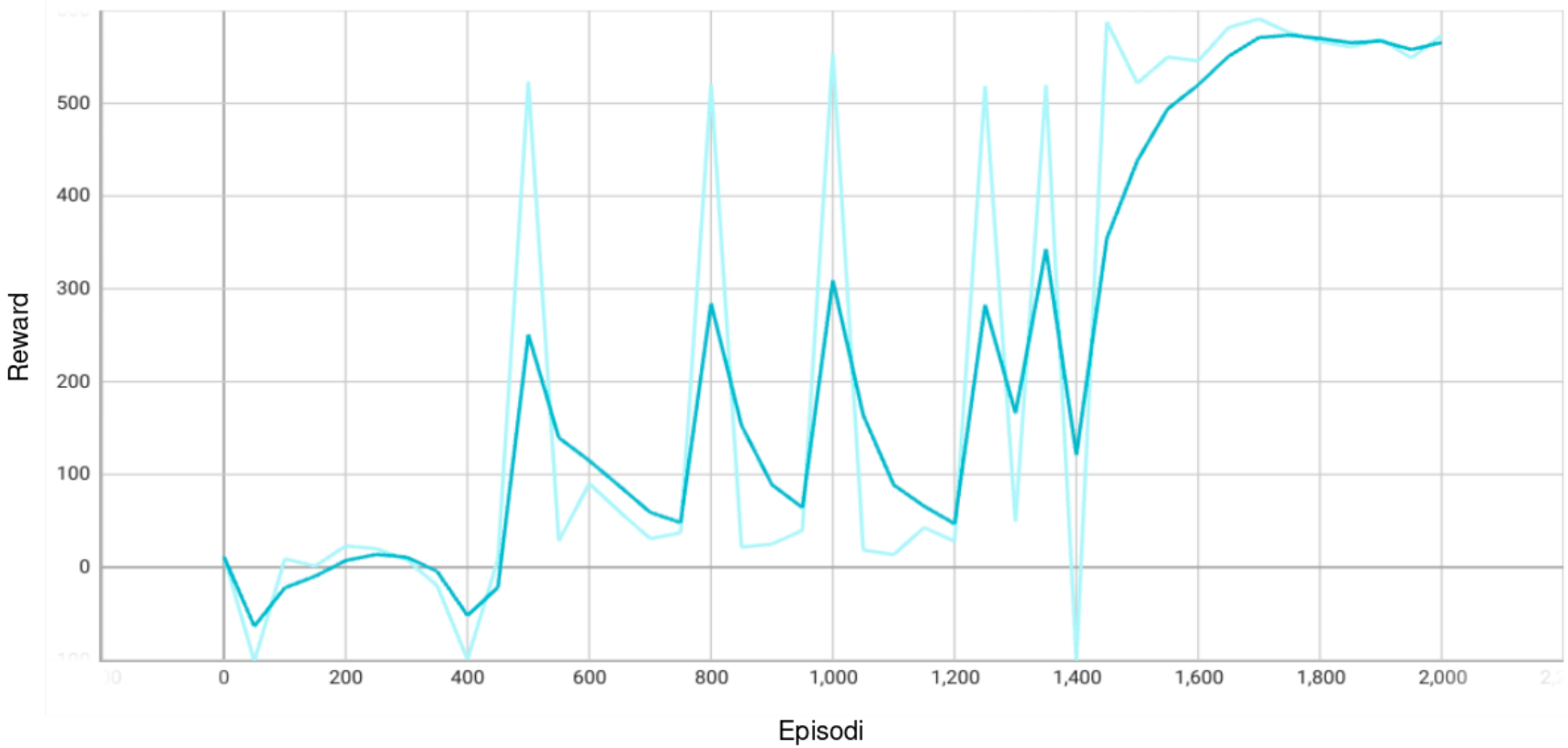
Risultati

- "Terra Sicura", Reward Goal: $+100$, Collision Reward: -100



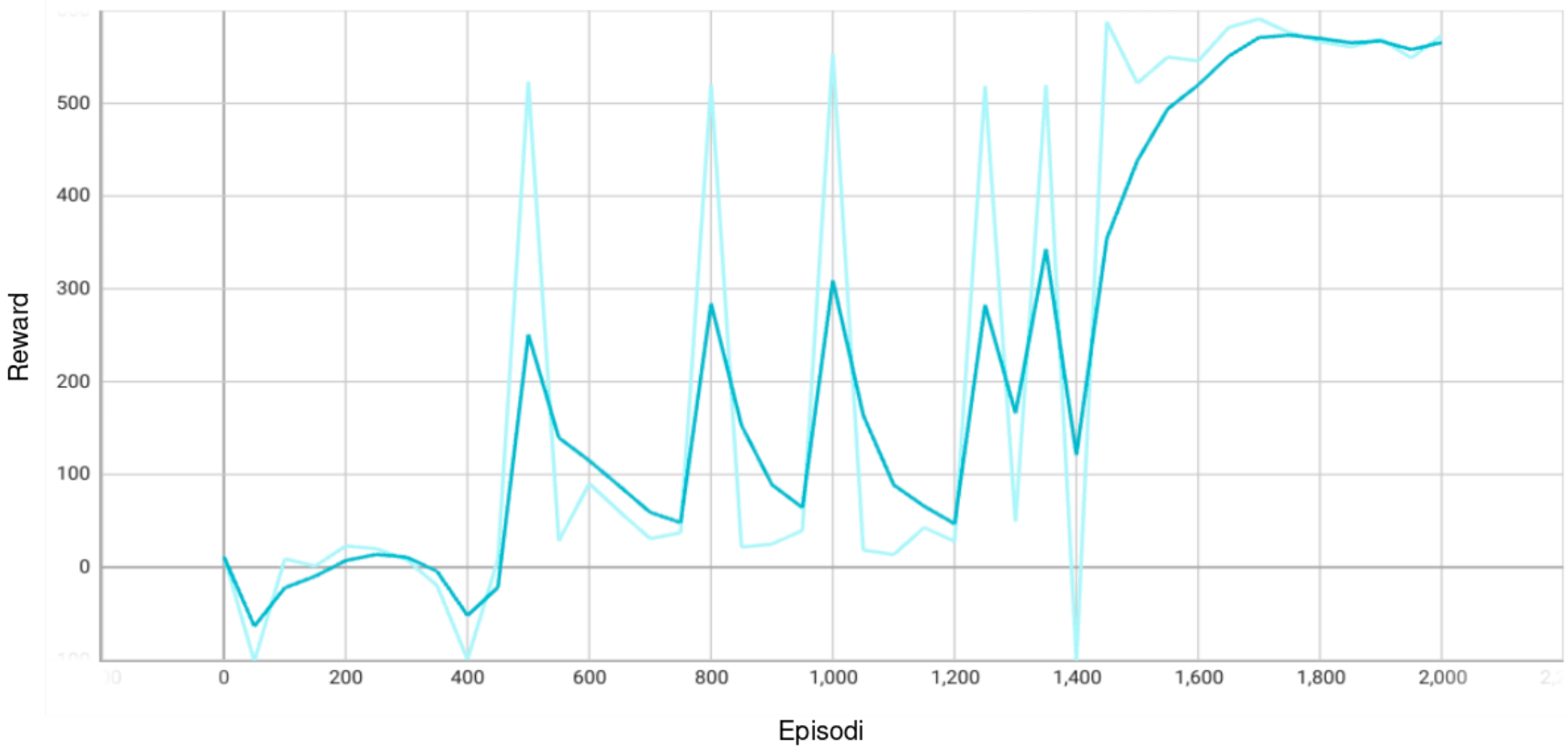
Risultati

- "Terra Sicura", Reward Goal: $+100$, Collision Reward: -100
- Curriculum Learning, Reward Goal: $+500$, Numero Ostacoli: 0



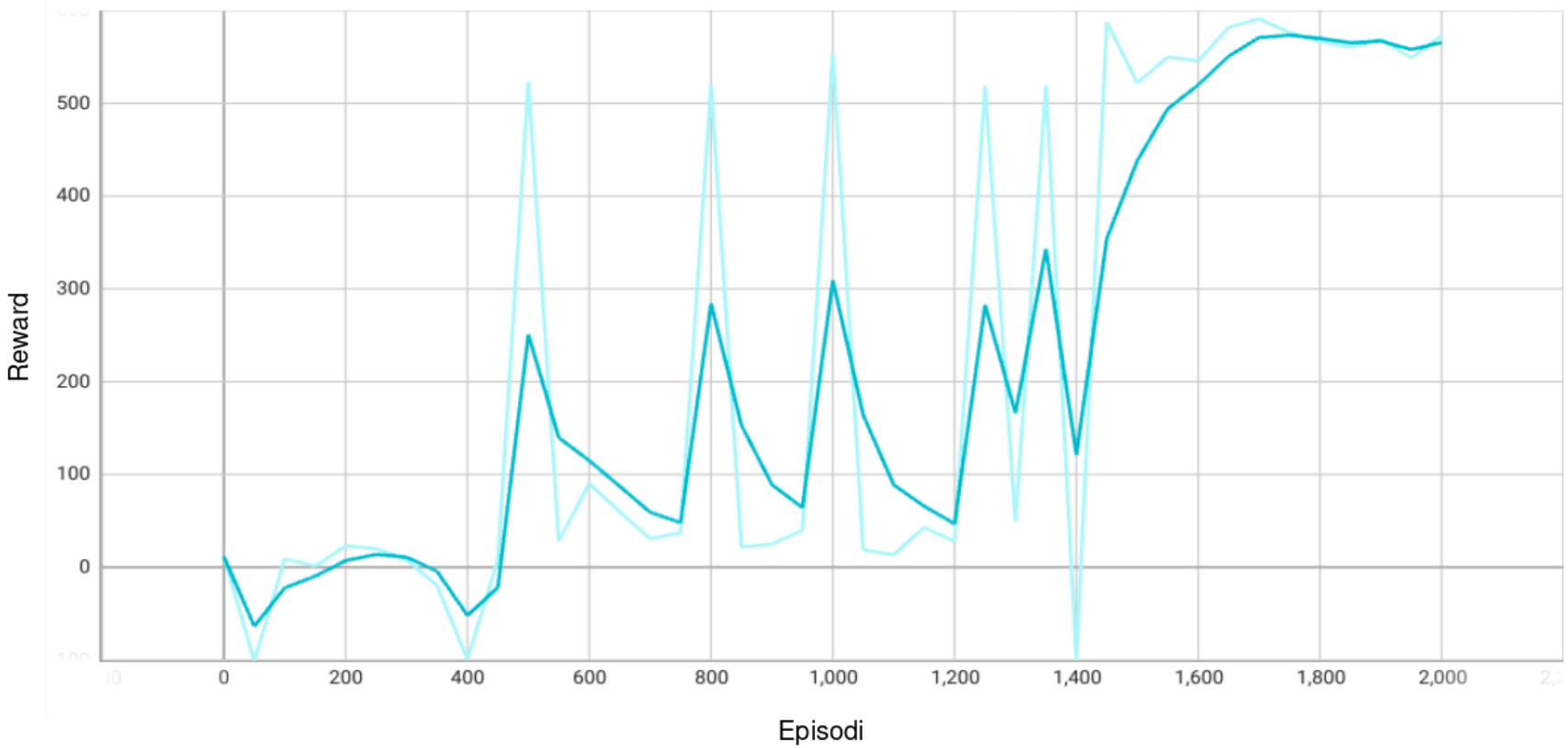
Risultati

- "Terra Sicura", Reward Goal: $+100$, Collision Reward: -100
- Curriculum Learning, Reward Goal: $+500$, Numero Ostacoli: 0
- Coraggio VS Avventatezza



Risultati

- "Terra Sicura", Reward Goal: $+100$, Collision Reward: -100
- Curriculum Learning, Reward Goal: $+500$, Numero Ostacoli: 0
- Coraggio VS Avventatezza
  - Reward Goal ridotto e Numero Ostacoli via via aumentati



Risultati

- Migliori Risultati: