ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

BÁO CÁO TIẾN ĐỘ

Học phần: Tính toán tiến hóa

Bài báo: Dynamic Path Planning for Mobile Robots with Deep Reinforcement Learning

Sinh viên thực hiện:
Hà Trung Kiên
Đào Thành Mạnh

ONE LOVE. ONE FUTURE.

MỤC LỤC:

PHẦN 1: ĐẶT VẤN ĐỀ
PHẦN 2: THUẬT TOÁN SOFT ACTOR-CRITIC (SAC)
PHẦN 3: ĐÁNH GIÁ HIỆU SUẤT
PHẦN 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN TRONG TƯƠNG LAI

1.1. Đặt vấn đề

Hiện nay, robot di động đang phát triển theo hướng tự học và thích nghi thông minh. Với việc sử dụng robot di động ngày càng phổ biến, việc thiết kế các thuật toán hoạch định đường đi chính xác và hiệu quả là một trong những thách thức lớn. Các phương pháp cổ điển bao gồm thuật toán truyền thống, thuật toán đồ họa và thuật toán thông minh bionic. Tuy nhiên, chúng vẫn chưa đủ để khai thác hoạch định đường đi thông minh cho robot di động do nhiều thách thức. Các thuật toán truyền thống như Simulated Annealing (SA), Artificial Potential Field (APF) và Fuzzy Logic dễ triển khai nhưng thường gặp khó khăn trong việc mô hình hóa, không thể sử dụng hiệu quả kiến thức trước đó, dễ rơi vào cực tiểu cục bộ và không thể tìm thấy tối ưu toàn cục.

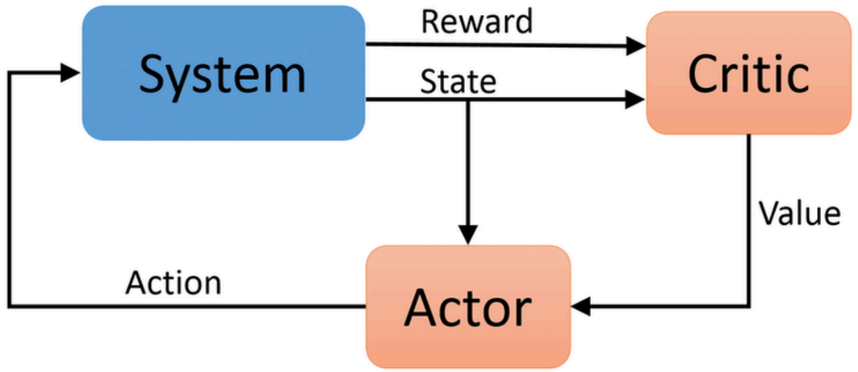
Bài báo đề xuất một thuật toán lập kế hoạch đường đi cho robot di động dựa trên thuật toán Soft Actor-Critic (SAC) trong môi trường phức tạp với chướng ngại vật tĩnh và động. Bài báo tập trung vào việc cải thiện hiệu suất di chuyển của robot bằng cách kết hợp các kỹ thuật học tăng cường sâu (Deep Reinforcement Learning), cho phép robot tự học và thích nghi với môi trường. Mục tiêu của thuật toán là giúp robot di động có thể liên tục thực hiện các hành động để tránh chướng ngại vật và cuối cùng đến được đích cuối cùng

a. Actor-Critic

Actor-Critic là một kiến trúc phổ biến trong học tăng cường sâu (Deep Reinforcement Learning) được sử dụng để huấn luyện các agent (tác nhân) nhằm đưa ra quyết định tối ưu trong môi trường. Kiến trúc này kết hợp hai thành phần chính:

- Actor: Là mô hình thực hiện hành động dựa trên policy hiện tại. Actor cập nhật policy để quyết định hành động nào nên thực hiện trong môi trường. Actor chủ yếu đưa ra hành động dựa trên xác suất trong policy của mình.

- Critic: Là mô hình đánh giá hành động của Actor bằng cách tính toán giá trị phần thưởng kỳ vọng (Expected Reward) hoặc hàm lợi ích (Advantage Function). Critic giúp Actor cải thiện bằng cách phản hồi xem hành động có tốt hay không, dựa trên việc so sánh phần thưởng đạt được so với kỳ vọng.



b. Soft Actor-Critic (SAC) .

Soft Actor-Critic (SAC) là một phiên bản cải tiến của kiến trúc **Actor-Critic** trong học tăng cường, được thiết kế để cân bằng giữa khả năng tối ưu hóa hành động và khả năng khám phá trong môi trường. Điểm khác biệt lớn nhất của SAC so với Actor-Critic truyền thống chính là **entropy**

Entropy trong học tăng cường là một khái niệm dùng để đo lường mức độ ngẫu nhiên hoặc không chắc chắn trong lựa chọn hành động của một tác nhân. Trong bối cảnh học tăng cường, entropy được sử dụng để khuyến khích tác nhân thử nghiệm và khám phá nhiều hành động khác nhau thay vì luôn chọn hành động có phần thưởng (reward) cao nhất.

PHẦN 2: THUẬT TOÁN SOFT ACTOR - CRITIC (SAC)

Đầu vào bài toán:

- Vị trí (State) hiện tại của Robot, điểm đích
- Khoảng cách đến các chướng ngại vật
- Tốc độ và hướng di chuyển của robot và các chướng ngại vật

Đầu ra bài toán:

- Hành động (Action) mà robot thực hiện, bao gồm
  - Tốc độ di chuyển: Robot sẽ điều chỉnh tốc độ dựa trên khoảng cách và hướng tới mục tiêu hoặc tránh chướng ngại vật.
  - Hướng di chuyển: Robot sẽ chọn hướng trong phạm vi 360 độ, phù hợp với việc tiếp cận mục tiêu hoặc tránh va chạm với chướng ngại vật.

PHẦN 2: THUẬT TOÁN SOFT ACTOR - CRITIC (SAC)

3 yếu tố chính của thuật toán:

- **Trạng thái hệ thống (System state)**
  - Thông tin vị trí, vận tốc và hướng di chuyển theo thời gian thực của robot.
  - Thông tin vị trí của chướng ngại vật động.
  - Thông tin vị trí của đích đến.

$$s=\{l_{agent}, d_{obstacle}, d_{target}, v_{agent}, v_{obstacle}\}$$

- **Hành động hệ thống (System action)**
  - Thuật toán điều hướng truyền thống thường chia không gian hành động liên tục thành các hướng rời rạc (đông, tây, nam, bắc). Tuy nhiên, ở đây, không gian hành động được thiết kế liên tục với hướng di chuyển và tốc độ bất kỳ, phù hợp hơn với tình huống thực tế.

- **Hàm phần thưởng (Reward fucntion)**
  - Hàm phần thưởng rất quan trọng để hướng dẫn robot tìm mục tiêu nhanh chóng và tránh va chạm. Hàm tổng phần thưởng bao gồm:
  - r1: Thưởng cho việc đi đúng góc.
  - r2: Thưởng cho việc tìm thấy đích đến.
  - r3: Phạt khi va chạm chướng ngại vật.
  - r4: Thưởng/phạt cho các tình huống kết thúc.
  -

$$r_{total}=r_1+r_2+r_3+r_4$$

PHẦN 2: THUẬT TOÁN SOFT ACTOR - CRITIC (SAC)

Hàm phần thưởng (Reward fucntion)

- **Hướng vận tốc của mỗi robot cần phải trùng với hướng vector từ robot đến mục tiêu. Do đó:**

$$r_1 = \cos(l_{vector}, v)$$

l-vector biểu thị vector từ vị trí của mỗi robot đến vị trí mục tiêu
v biểu thị vector hướng di chuyển của mỗi robot

- **Robot cần phải nhanh chóng tìm thấy đích đến:** Mỗi robot cần nhận được phần thưởng reward > 0 khi di chuyển về phía mục tiêu và bị phạt với giá trị âm khi di chuyển xa khỏi mục tiêu. Do đó:

$$r_2 = -\frac{d}{500} \times 0.4 + 1.0 + \frac{pre_d - 0.95 \times d}{24}$$

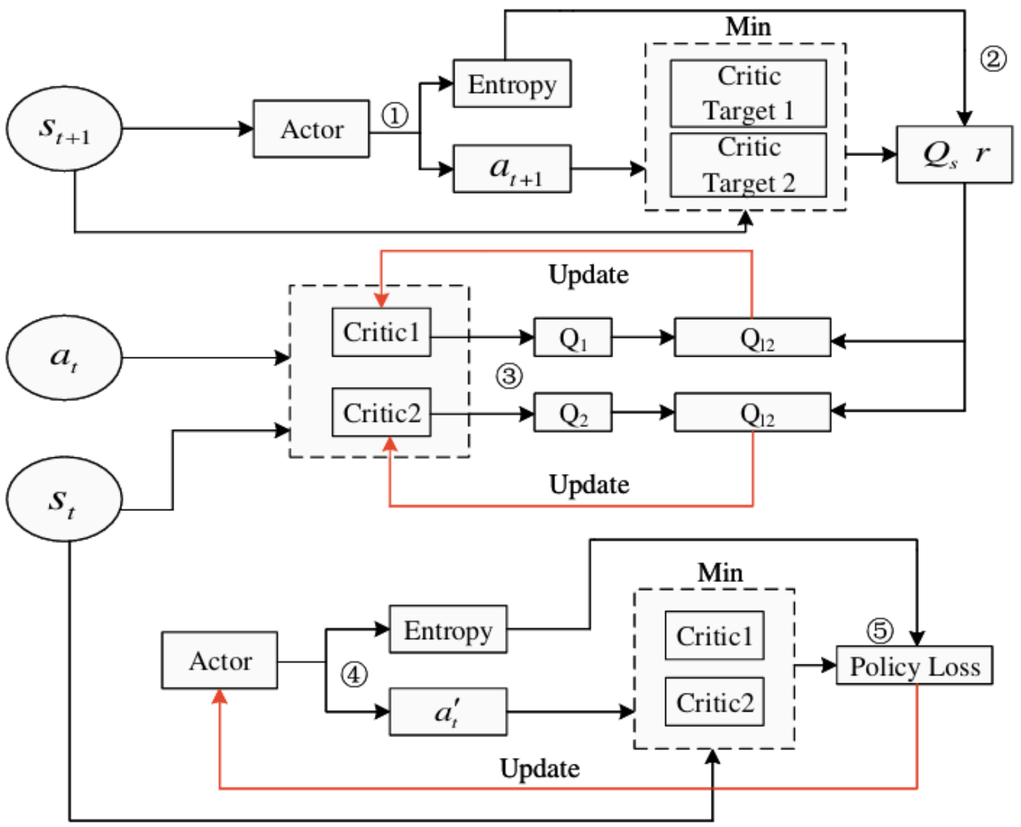
pred_d biểu thị khoảng cách giữa mỗi robot và mục tiêu trong lần lặp trước đó, và d là khoảng cách giữa mỗi robot và mục tiêu trong lần lặp hiện tại.

- **Robot cần tránh va chạm với chướng ngại vật:** Mỗi robot cần điều chỉnh hướng di chuyển của mình để tránh va chạm với chướng ngại vật trong quá trình di chuyển
  - Dangerous area: Nếu có chướng ngại vật trong bán kính 5 mét
  - Safe area: Ngược lại
  - Khi robot phát hiện có chướng ngại vật trong vòng 5 mét trong hướng di chuyển của nó, môi trường sẽ cho điểm phạt reward là -1 để nhắc nhở nó điều chỉnh hướng. Ngoài ra, khi robot di chuyển từ khu vực nguy hiểm sang khu vực an toàn, nó sẽ nhận được phần thưởng reward là 0.5.

- **Các điều kiện kết thúc bao gồm:**
  - Điểm phạt reward = -5 khi robot đã đến được đích
  - Điểm phạt reward = -6 khi robot va chạm với chướng ngại vật
  - Điểm thưởng reward = 100 khi robot chạm tới ranh giới của khu vực tìm kiếm

PHẦN 2: THUẬT TOÁN SOFT ACTOR - CRITIC (SAC)

Mô tả cấu trúc mạng SAC



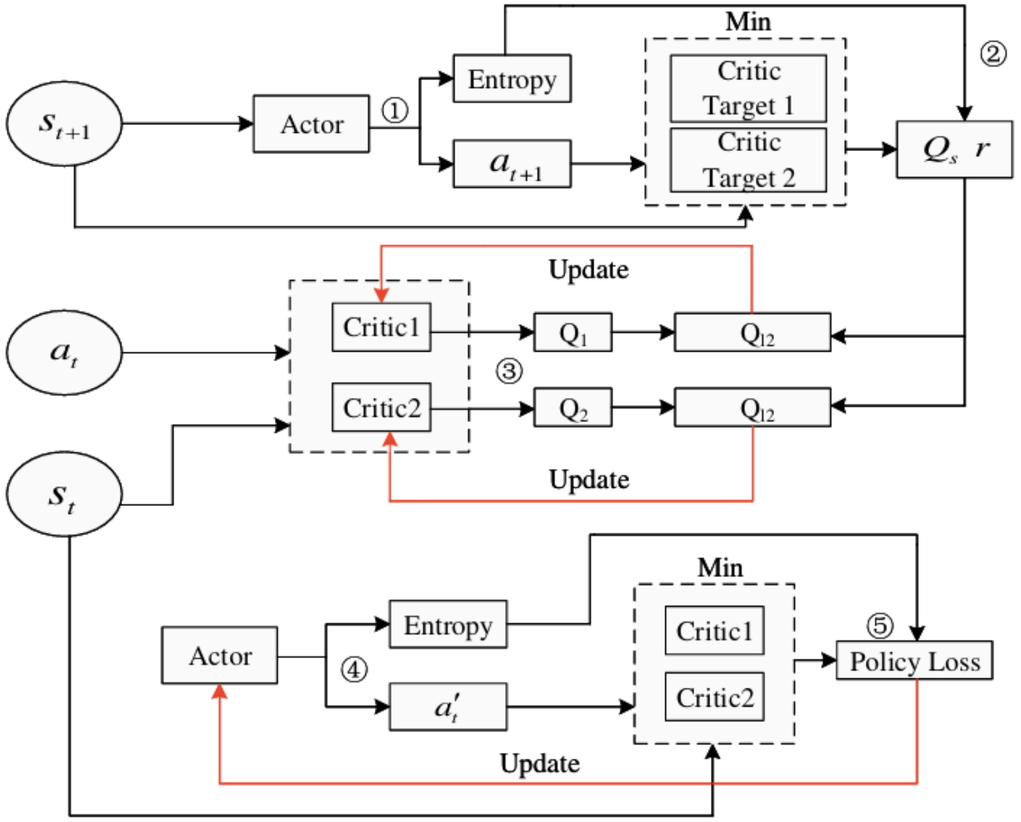
Ở phần (1). Action Entropy được lấy từ đầu ra của Actor, được tính như sau: $H(\pi(\cdot \mid s_{t+1})) = -\log t\pi(a_{t+1} \mid s_{t+1})$

Trong đó: a_{t+1} biểu thị đầu ra hành động của actor từ trạng thái môi trường s_{t+1}

$\pi(\cdot \mid s_{t+1})$ là xác suất xuất ra a_{t+1} từ s_{t+1}

PHẦN 2: THUẬT TOÁN SOFT ACTOR - CRITIC (SAC)

Mô tả cấu trúc mạng SAC



Ở phần (2). Giá trị ước tính được tính như sau:

$$Q_s(r, s_{t+1}) = r + \lambda(V(s_{t+1})) =$$
$$r + \lambda(Q_\pi(s_{t+1}, a_{t+1}) + \alpha H(\pi(\cdot \mid s_{t+1}))) =$$
$$r + \lambda(\min_{j=1,2} Q_{\varphi_{tj}}(s_{t+1}, a_{t+1}) - \alpha \log \pi(a_{t+1} \mid s_{t+1}))$$

Trong đó:

$V_{s_{t+1}}$ biểu thị giá trị của trạng thái s_{t+1}

r từ bộ đệm phát lại (replay buffer)

$Q_\pi(s_{t+1}, a_{t+1})$ là ước tính giá trị của hành động at+1 theo st+1

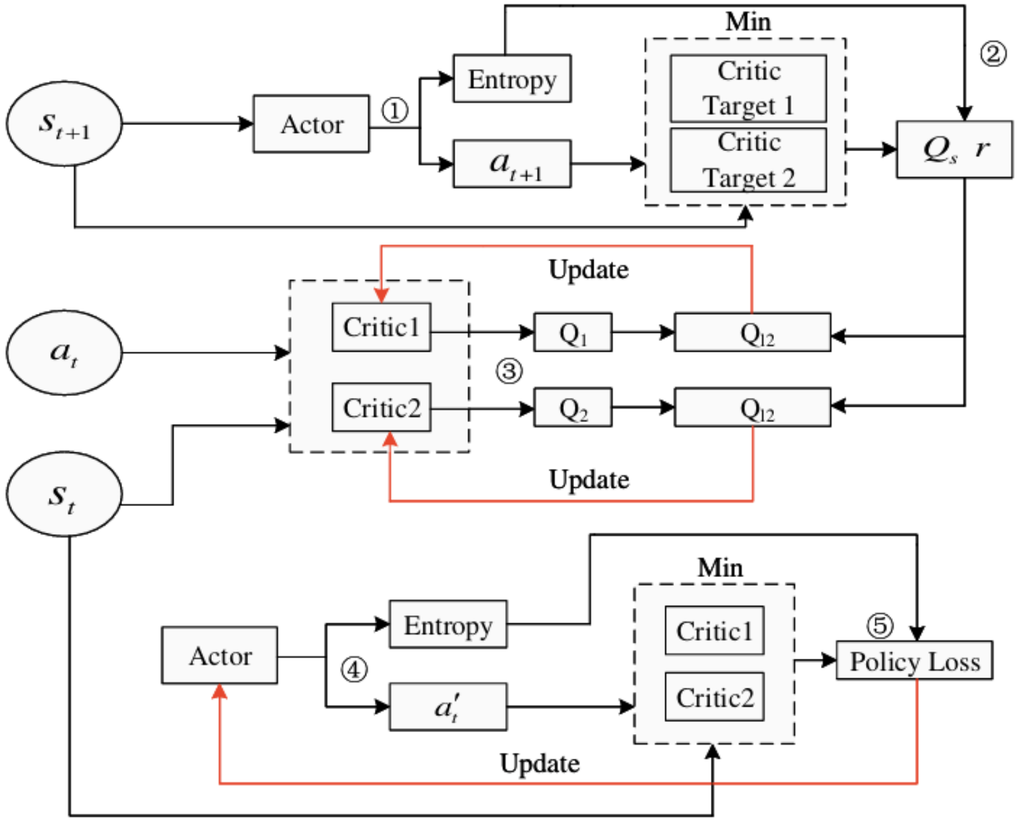
α là trọng số của Entropy

φ_{tj} là tham số của hai mạng Critic Target

$\min_{j=1,2} Q_{\varphi_{tj}}$ chỉ ra việc lấy giá trị nhỏ nhất của đầu ra của hai mạng Critic Target, có thể ngăn ngừa hiệu quả việc ước tính quá mức

PHẦN 2: THUẬT TOÁN SOFT ACTOR - CRITIC (SAC)

Mô tả cấu trúc mạng SAC



Từ (3), có thể thấy SAC có 2 mạng Critic với cùng kiến trúc
Hàm mất mát $L(\varphi_i, D)$ của hai mạng được cho bởi:

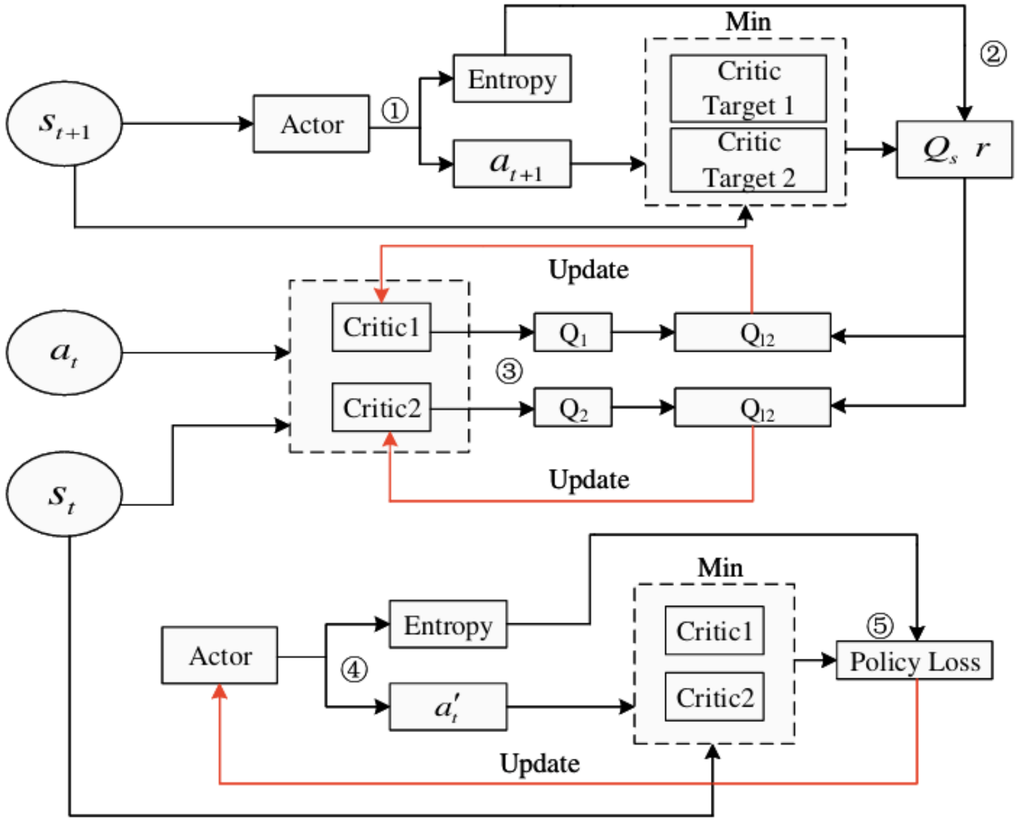
$$L(\varphi_i, D) = E_{(s_t,r,s_{t+1},a_t)-D}[(Q_{\varphi_i}(s_t, a_t) - Q_s(s_{t+1}))^2]$$

$E_{(s_t,r,s_{t+1},a_t)-D}$ trong đó (s_t, r, s_{t+1}, a_t) lấy từ replay buffer D

$Q_{\varphi_i}(s_t, a_t)$ là ước tính giá trị Q-value với trọng số φ_i , trạng thái môi trường s_{t+1}

PHẦN 2: THUẬT TOÁN SOFT ACTOR - CRITIC (SAC)

Mô tả cấu trúc mạng SAC



Sau đó, cập nhật từng mạng Critic Target theo tỷ lệ với tham số ϕ quá trình này là sliding average update:

$$\phi_{ti} \leftarrow \rho\phi_{ti} + (1-\rho)\phi_i, \quad i \in \{1,2\}$$

PHẦN 2: THUẬT TOÁN SOFT ACTOR - CRITIC (SAC)

Tóm tắt quá trình chung của SAC

Algorithm 1 Soft Actor Critic algorithm

1: Initialize parameters of SAC and that of each network
2: **for** each iteration **do**
3: **for** each environment step **do**
4: A mobile robot in the environment state s_t executes action a_t from the actor network and obtains a reward r_{t+1}. The state is changed to s_{t+1}. Put this experience $(s_t, a_t, s_{t+1}, r_{t+1})$ into the replay buffer.
5: **end for**
6: **if** it is time to update **then**
7: **for** each gradient step **do**
8: Gain a certain number of experiences $B(s_t, a_t, s_{t+1}, r_{t+1}))$ from the replay buffer.
9: The critic target network estimates Q values with $Q_s(r, s_{t+1})$
10: Update critic network by $\nabla_{\varphi_i} \frac{1}{|B|} \sum_{(s_t, a_t, r, s_{t+1}) \in B} (Q_{\varphi_i}(s_t, a_t) - Q_s(r, s_{t+1}))^2$
11: Update actor network by $\nabla_\theta \frac{1}{|B|} \sum_{s \in B} (\min_{j=1,2} Q_{\varphi_i}(s_t, a'_t) - \alpha \log \pi_\theta(a_t \mid s_t))$
12: Update critic target network by
13: **end for** $\phi_{ti} \leftarrow \rho \phi_{ti} + (1 - \rho)\phi_i, \quad i \in \{1, 2\}$
14: **end if**
15: **end for**

Trạng thái hiện tại s_t chính là input cho mạng actor
Mạng actor sẽ tính toán giá trị trung bình và phương sai của phân phối Gaussian, sau đó lấy mẫu hành động output theo phân phối xác suất

Mỗi mobile robot thực thi hành động a_t
và nhận phần thưởng reward r_{t+1} từ môi trường
Trạng thái sau đó thay đổi thành s_{t+1}
Kinh nghiệm hiện tại sẽ được đưa vào replay buffer

Khi số lượng kinh nghiệm đạt đến ngưỡng nhất định, mạng critic sẽ được update dựa trên mẫu từ một loạt kinh nghiệm trong replay buffer. Quá trình này được lặp lại cho đến khi mạng target hội tụ

PHẦN 3: ĐÁNH GIÁ HIỆU SUẤT

Để đánh giá hiệu suất của thuật toán được đề xuất, bài báo đã sử dụng môi trường mô phỏng dựa trên Pygame

- **Kích thước môi trường:** 400x400
- **Các đối tượng trong môi trường:**
  - **Robot di động:** Hình vuông màu đỏ
  - **Chướng ngại vật tĩnh:** Hình vuông màu xanh lam
  - **Chướng ngại vật động:** Hình vuông màu trắng
  - **Mục tiêu:** Hình vuông màu vàng

Table 1. Parameter setting of SAC

Parameter	Setting
Discount	0.95
Learning rate	0.0003
Number of samples per minibatch	256
Replay buffer size	10^7
Target smoothing coefficient	0.005
Initial value of α	1.0
Gradient steps	1
Target update interval	1
Exploration function	Stochastic sampling

PHẦN 3: ĐÁNH GIÁ HIỆU SUẤT

Kết quả mô phỏng khẳng định tính hiệu quả của thuật toán SAC trong việc lập kế hoạch đường đi cho robot di động trong môi trường phức tạp. SAC vượt trội so với PPO về khả năng học hỏi, tính ổn định, độ mạnh mẽ và hiệu quả di chuyển. Việc chuẩn hóa trạng thái và sử dụng kinh nghiệm ưu tiên cũng góp phần tăng tốc quá trình học và nâng cao hiệu suất của thuật toán.

PHẦN 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN TRONG TƯƠNG LAI

- Bài báo chưa xem xét đến một số yếu tố thực tế như gia tốc của robot và ma sát của môi trường. Bài báo đề xuất hướng nghiên cứu bổ sung các yếu tố này để mô hình phản ánh chính xác hơn môi trường thực tế.

- Bài báo sử dụng mạng nơ-ron kết nối đầy đủ (fully connected network). Nhóm tác giả đề xuất thử nghiệm sử dụng mạng nơ-ron tích chập (convolutional neural network) để cải thiện hiệu suất lập kế hoạch đường đi.

HUST

THANK YOU