NGHIÊN CỨU THUẬT TOÁN ĐIỀU KHIỂN ROBOT 4 CHÂN

STUDY ON QUADRUPED ROBOT CONTROLLING ALGORITHM

Lưu Thanh Tùng[1,2], Nguyễn Đức Thiên Ân[1,2], Phan Khánh Duy[1,2], Tô Nghĩa Nhân[1,2]
[1]Khoa Cơ khí, Trường Đại học Bách Khoa, Đại học Quốc gia Thành phố Hồ Chí Minh (HCMUT)
[2]Đại học Quốc gia Thành phố Hồ Chí Minh (VNU-HCM)

TÓM TẮT

Trong thời đại hiện đại, việc điều khiển robot 4 chân đang đặt ra những thách thức đáng kể. Các khó khăn bao gồm việc đồng bộ hóa chuyển động của các chân, duy trì cân bằng trên mặt đất không đồng đều và thích ứng với môi trường xung quanh. Các phương pháp điều khiển truyền thống thường gặp khó khăn trong việc xử lý các yếu tố phi tuyến và sự ngẫu nhiên từ môi trường, dẫn đến hạn chế về hiệu suất và sự linh hoạt của robot. Để vượt qua những thách thức này, chúng tôi đề xuất sử dụng thuật toán học tăng cường, cùng với đó sử dụng trình mô phỏng máy tính để thiết lập thuật toán, mở ra cơ hội để điều khiển robot 4 chân một cách linh hoạt và hiệu quả hơn, từ đó có thể điều khiển robot 4 chân di chuyển linh hoạt trên địa hình gồ ghề.

Từ khóa: *Robot 4 chân; Mô phỏng; Học tăng cường.*

ABSTRACT

In the modern era, controlling quadruped robots poses significant challenges. Difficulties include synchronizing the motion of the legs, maintaining balance on uneven terrain, and adapting to the surrounding environment. Traditional control methods often struggle to handle nonlinear and uncertain factors, leading to limitations in the performance and flexibility of robots. To overcome these challenges, we propose using reinforcement learning algorithms, building simulation environment to opening up opportunities to control quadrupedal robots more flexibly and effectively over rough terrain.

Keywords: *Quadruped robot; Simulation; Reinforcement learning.*

1. TỔNG QUAN

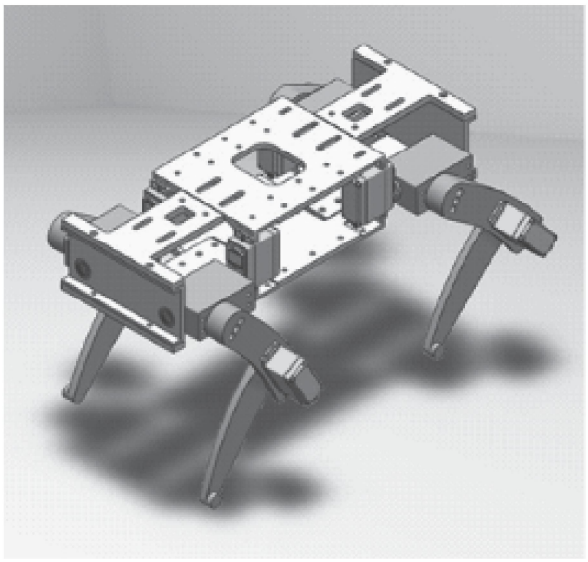
Bài toán di chuyển là một yếu tố quan trọng đối với robot di động và đặc biệt là trong lĩnh vực thiết kế robot 4 chân, như robot chó. Trong quá trình làm việc, robot có thể gặp phải những loại địa hình, hay vật cản chưa từng gặp trước đây, hay việc hoạt động trong các môi trường có tính ngẫu nhiên cao gây khó khăn cho các thuật toán điều khiển thông thường. Các tiến triển gần đây trong lĩnh vực thiết kế và giải thuật điều khiển của robot bốn chân đã đánh dấu một bước phát triển đáng kể, mở ra những cơ hội mới với khả năng di chuyển linh hoạt của chúng.

Trên thế giới, hiện nay, các phương pháp điều khiển robot 4 chân sử dụng học tăng cường tiêu biểu như [1], [2], gồm một bộ điều khiển sử dụng thuật toán học tăng cường và một bộ hành động tham chiếu để thực hiện dáng đi của robot. Hay như [3], [4], [5] sử dụng phương pháp học sinh-giáo viên (student-teacher method), một mạng nơ ron nhân tạo có đầy đủ thông tin từ môi trường (teacher network) và tạo hành động cho robot, cùng lúc đó một mạng nơ ron khác không truy cập đầy đủ được các thông số của môi trường (student network) sẽ học theo mạng nơ ron giáo viên để sinh ra hành động phù hợp cho robot. Một phương pháp khác khá nổi bật gần đây, được trình bày bởi [6], [7], đó là sử dụng những bộ dữ liệu MoCap (Motion Capture), là bộ dữ liệu ghi lại chuyển động của động vật để từ đó làm tham chiếu cho thuật toán học tăng cường. Tuy nhiên, những phương pháp trên đòi hỏi thời gian huấn luyện thuật toán lâu, đồng thời lượng tham số của mô hình học sâu rất lớn và thuật toán huấn luyện rất phức tạp. Việc huấn luyện thuật toán điều khiển đơn giản là một nhu cầu cần thiết và đáng được quan tâm.

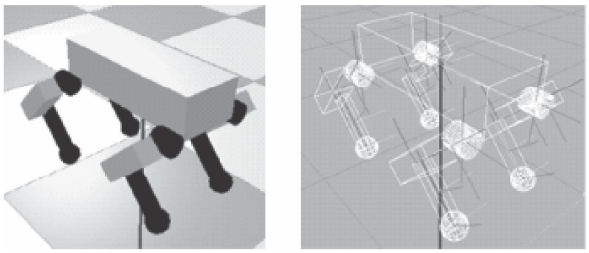
2. PHƯƠNG PHÁP

2.1. Mô hình robot

Để thực hiện công việc phát triển thuật toán, ta cần một mô hình robot đơn giản nhưng đầy đủ để có thể chạy được trong mô phỏng. Thiết kế hoàn thiện của robot 4 chân thường rất phức tạp, nếu sử dụng trực tiếp trong mô phỏng có thể gây chậm trễ và hao tốn tài nguyên tính toán. Vì vậy, một mô hình đơn giản (Hình 1) sẽ giúp dễ dàng thực hiện mô phỏng.



Hình 1. Mô hình robot 4 chân chi tiết



Hình 2. Mô hình robot 4 chân đơn giản.

Mô hình robot sử dụng trong mô phỏng có các thông số cơ bản trong bảng sau:

ISSN 2615 - 9910 (bản in), ISSN 2815 - 5505 (online)
TẠP CHÍ CƠ KHÍ VIỆT NAM, Số 314, tháng 4 năm 2024
cokhivietnam.vn / tapchicokhi.com.vn

Bảng 1. Thông số cơ bản của mô hình mô phỏng.

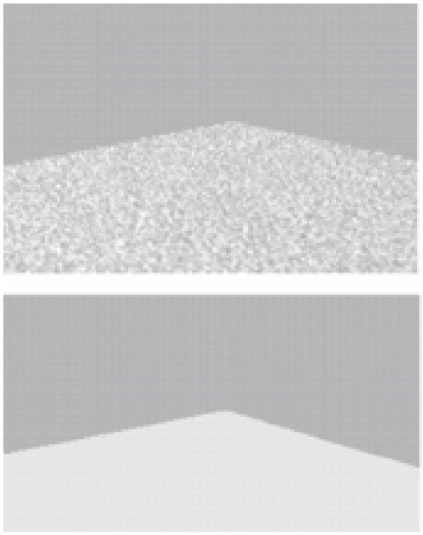
Thông số	Giá trị
Khối lượng thân	2 kg
Kích thước thân	0,4 × 0,15 × 0,1 m
Khối lượng vai	0,1 kg
Kích thước vai	0,025 × 0,05 m
Khối lượng đùi	0,1 kg
Kích thước đùi	0,05 × 0,05 × 0,15 m
Khối lượng bắp	0,1 kg
Kích thước bắp	0,03 × 0,03 × 0,1 m

Các kích thước này được lấy từ bản vẽ thiết kế robot, để đảm bảo mô hình mô phỏng sát với mô hình thực tế nhất có thể.

2.2. Môi trường mô phỏng

Môi trường mô phỏng được xây dựng trên trình mô phỏng Pybullet [8]. Mô hình robot sau khi thiết kế chi tiết sẽ được chuyển thành dạng file .urdf để tải lên mô phỏng. Trong môi trường mô phỏng của Pybullet, các bài toán va chạm sẽ được tính toán trong thời gian thực để đảm bảo mô phỏng sát với thực tế.

Các loại địa hình sẽ được mô hình hóa để huấn luyện thuật toán bao gồm địa hình phẳng và địa hình gồ ghề. Trong địa hình gồ ghề, các chỏm đá nhỏ được sinh ngẫu nhiên với cao độ lớn nhất là 5 (cm). Hình 2 thể hiện các địa hình huấn luyện thuật toán học tăng cường.



Hình 3. Trên: Địa hình gồ ghề với các mỏm đá nhỏ; Dưới: Địa hình mặt phẳng.

2.3. Thuật toán học tăng cường

Thuật toán học tăng cường Vanilla Policy Gradient (VPG) được sử dụng để huấn luyện thuật toán. Một số khái niệm để xây dựng thuật toán bao gồm:

- Bộ điều khiển π.
- Trạng thái của môi trường s.
- Đầu ra của thuật toán a.
- Phần thưởng từ môi trường r.
- Bộ điều khiển, là một hàm số có đầu vào là trạng thái quan sát được từ môi trường và đầu ra là một hành động. Policy thường được ký hiệu là: π(a|s).

- Quỹ đạo, là một chuỗi các hành động, trạng thái môi trường và phần thưởng từ môi trường, thường được biểu diễn dưới dạng một tập hợp nhiều trạng thái, hành động và phần thưởng xếp theo thứ tự như sau:

$$\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \dots, s_t, a_t, r_t). \quad (1)$$

Hàm chuyển tiếp, giả sử trạng thái của ☞

môi trường tại một thời điểm chỉ phụ thuộc vào trạng thái của môi trường tại thời điểm trước đó, thì hàm chuyển tiếp được định nghĩa là: p(s'|s,a).

Tổng tích lũy phần thưởng, là giá mà môi trường trả về khi được tính trên một quỹ đạo, với γ là hệ số lạm phát: γ ∈(0,1), có giá trị như sau:

$$G(\tau|t) = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \quad (2)$$

Xác suất để ta bắt gặp một quỹ đạo có T bước thời gian là:

$$P(\tau|\pi) = p(s_0)\prod_{t=0}^{T-1} p(s_{t+1}|s_t) = p(s_0)\prod_{t=0}^{T-1} p(s_{t+1}|s_t, a_t)\pi(a_t|s_t) \quad (3)$$

Từ đó, ta có thể đưa ra khái niệm phần thưởng kỳ vọng trong một quỹ đạo là:

$$J(\tau) = \sum_{\tau}^{\infty} P(\tau|\pi)G(\tau) = E[G(\tau)] \quad (4)$$

Vì vậy, để tìm một bộ điều khiển tốt nhất để tối đa phần thưởng mà ta nhận về khi thực hiện một quỹ đạo chính là thực hiện bài toán tối ưu sau:

$$\pi^* = \arg\max_{\pi} J(\tau) \quad (5)$$

Để trực tiếp tìm được bộ điều khiển tốt nhất, ta thực hiện đạo hàm $\nabla_\theta E_{\tau\sim\pi_\theta}[G(\tau)]$, sau đó cập nhật tham số mô hình theo công thức Gradient Ascent cơ bản, với θ là thông số mạng nơ ron của bộ điều khiển π:

$$\theta \leftarrow \theta + \beta\, \nabla_\theta E_{\tau\sim\pi_\theta}[G(\tau)] \quad (6)$$

Trong đó, $\nabla_\theta E_{\tau\sim\pi_\theta}[G(\tau)]$ được biến đổi như sau:

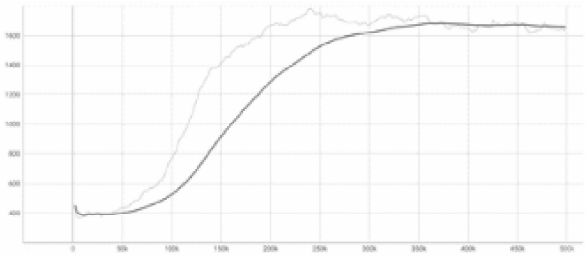
$$\nabla_\theta E_{\tau\sim\theta}[G(\tau)] = E_{\tau\sim\theta}[\sum \nabla_\theta \log \pi_\theta(a_t|s_t)\, G(\tau|s_t)] \quad (7)$$

Theo lý thuyết của VPG [9], [10], cập nhật thuật toán theo (6) sử dụng thuật toán tối ưu Gradient Ascent sẽ đạt được bộ điều khiển tối ưu π*.

3. THÍ NGHIỆM

3.1. Huấn luyện thuật toán

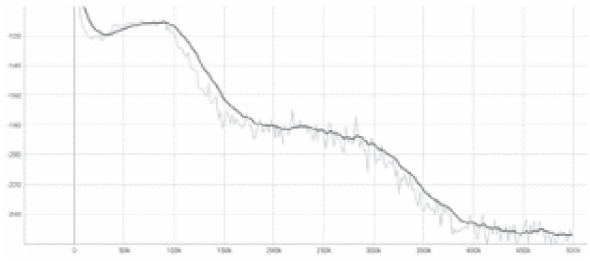
Thuật toán được huấn luyện 500.000 bước, với learning rate là 0,005. Tổng thời gian trong mô phỏng là 6 giờ và thời gian thực là 2 giờ đồng hồ. Bộ điều khiển được chọn là một mạng nơ ron đơn giản có 3 lớp và lớp ẩn có 256 nút. Ta sử dụng thuật toán tối ưu Adam để tối ưu thuật toán VPG. Các kết quả huấn luyện được thể hiện trên các Hình 3, Hình 4, Hình 5. Bộ dữ liệu để huấn luyện thuật toán sẽ được thu thập song song với việc thực toán chạy trong mô phỏng. Để tiết kiệm thời gian và tối ưu tài nguyên tính toán, 8 môi trường mô phỏng được chạy cùng lúc để thu thập dữ liệu cho thuật toán.



Hình 4. Tổng phần thưởng tích lũy của mô hình.

Hình 5. Độ dài một lần mô phỏng.

ISSN 2615 - 9910 (bản in), ISSN 2815 - 5505 (online)
TẠP CHÍ CƠ KHÍ VIỆT NAM, Số 314, tháng 4 năm 2024
cokhivietnam.vn / tapchicokhi.com.vn

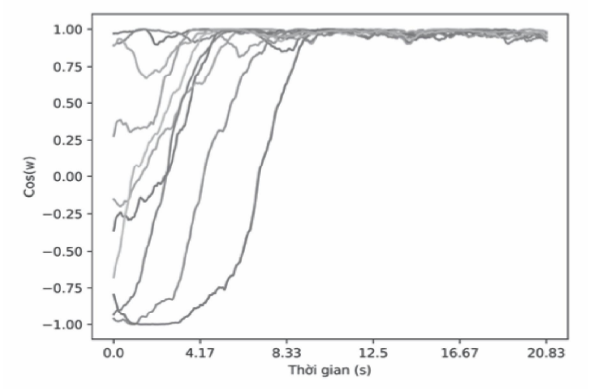


Hình 6. Mất mát của bộ điều khiển.

4. KẾT QUẢ VÀ THẢO LUẬN

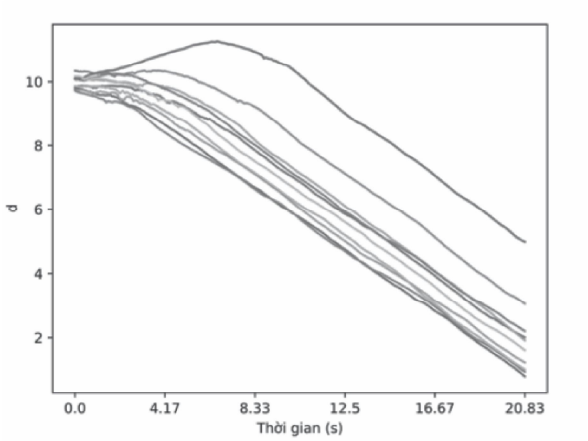
Trong số các thông số đánh giá của thuật toán thì tổng phần thưởng tích lũy là quan trọng bậc nhất, thông số này phản ánh khả năng đạt được mục tiêu đã đề ra của thuật toán. Giá trị càng cao chứng tỏ trong một vòng lặp mô phỏng, thuật toán của ta đã điều khiển robot đi đúng hướng.

Một yếu tố khác có thể dùng để đánh giá thuật toán chính là bằng cách nhìn vào đáp ứng của thuật toán đối với một điểm mục tiêu bất kỳ trong môi trường mô phỏng. Vì đầu vào của thuật toán bao gồm một véc-tơ chỉ phương trong hệ tọa độ robot, chỉ đến mục tiêu, vì thế, ta có thể thực hiện tính toán và khảo sát khoảng cách cosin (cosine distance) giữa góc quay mặt hiện tại và véc-tơ chỉ phương để biết được đáp ứng thuật toán.



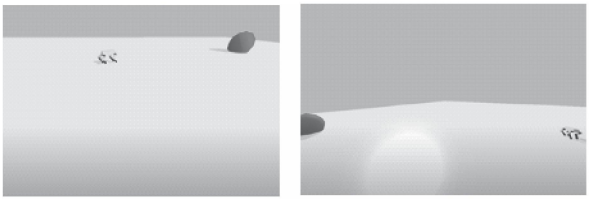
Hình 7. Đáp ứng khoảng cách cosin của góc quay robot.

Theo như quan sát, hầu hết các đường đáp ứng đều đạt đến giá trị lớn hơn 0,75 sau 5 giây, tuy chỉ có hai trường hợp cá biệt đó là đường xanh lá và đỏ là đáp ứng chậm so với các đường còn lại, đó là vì khi này mục tiêu nằm sau lưng robot và thuật toán điều khiển cần tốn một khoảng thời gian để cho robot quay đầu.



Hình 8. Khoảng cách giữa robot và mục tiêu theo thời gian.

Từ hình trên, có thể thấy robot di chuyển với vận tốc gần như tuyến tính, và cần nhiều thời gian hơn để có thể đi đến được vị trí của mục tiêu nếu vị trí mục tiêu không thuận lợi với hướng quay ban đầu của robot.



Hình 9. Trái: Di chuyển đến mục tiêu trên đường thẳng; Phải: Quay đầu sau đó tiến về mục tiêu.

5. KẾT LUẬN

Thuật toán huấn luyện đã điều khiển robot di chuyển trên địa hình phẳng và gồ ghề ☞

đến mục tiêu với độ tin cậy cao. Kết quả thử nghiệm trên mô phỏng cho thấy robot có thể hoàn thành nhiệm vụ di chuyển đến mục tiêu trong thời gian ngắn và dễ dàng vượt qua địa hình gập ghềnh trong mô phỏng. Với độ phức tạp thấp và thời gian huấn luyện không cao, đây chính là nền móng cho việc triển khai thuật toán vào thiết bị phần cứng dễ dàng và nhanh gọn hơn.

Lời cảm ơn:

Nghiên cứu này được tài trợ bởi Trường Đại học Bách khoa, Đại học Quốc gia Thành phố Hồ Chí Minh trong khuôn khổ đề tài mã số SVKSTN-2023-CK-26. Chúng tôi xin cảm ơn Trường Đại học Bách khoa, ĐHQG-HCM đã hỗ trợ cho nghiên cứu này.❖

Ngày nhận bài: **17/02/2024**
Ngày phản biện: **27/3/2024**

Tài liệu tham khảo:

[1]. Tan, J., Zhang, T., Coumans, E., Iscen, A., Bai, Y., Hafner, D., ... & Vanhoucke, V. (2018). Sim-to-real: *Learning agile locomotion for quadruped robots*. Preprint arXiv: 1804.10332.

[2]. Lee, J., Hwangbo, J., Wellhausen, L., Koltun, V., & Hutter, M. (2020). *Learning quadrupedal locomotion over challenging terrain.* Science robotics, 5(47), eabc5986.

[3]. Ma, Y., Farshidian, F., Miki, T., Lee, J., & Hutter, M. (2022). *Combining learning-based locomotion policy with model-based manipulation for legged mobile manipulators.* IEEE Robotics and Automation Letters, 7(2), 2377-2384.

[4]. Miki, T., Lee, J., Hwangbo, J., Wellhausen, L., Koltun, V., & Hutter, M. (2022). *Learning robust perceptive locomotion for quadrupedal robots in the wild.* Science Robotics, 7(62), eabk2822.

[5]. Agarwal, A., Kumar, A., Malik, J., & Pathak, D. (2023). *Legged locomotion in challenging terrains using egocentric vision*. Conference on robot learning (pp. 403-415). PMLR.

[6]. Peng, X. B., Andrychowicz, M., Zaremba, W., & Abbeel, P. (2018). *Sim-to-real transfer of robotic control with dynamics randomization.* 2018 IEEE international conference on robotics and automation (ICRA) (pp. 3803-3810). IEEE.

[7]. Bohez, S., Tunyasuvunakool, S., Brakel, P., Sadeghi, F., Hasenclever, L., Tassa, Y., ... & Heess, N. (2022). *Imitate and repurpose: Learning reusable robot movement skills from human and animal behaviors.* arXiv preprint arXiv:2203.17138.

[8]. Coumans, E., & Bai, Y. (2016). *Pybullet, a python module for physics simulation for games, robotics and machine learning.*

[9]. Sutton, R. S., McAllester, D., Singh, S., & Mansour, Y. (1999). *Policy gradient methods for reinforcement learning with function approximation.* Advances in neural information processing systems, 12.

[10]. Schulman, J. (2016). *Optimizing expectations: From deep reinforcement learning to stochastic computation graphs.* Doctoral dissertation, UC Berkeley.