Học cách di chuyển nhanh hơn cho robot bốn chân sử dụng học tăng cường sâu không có mô hình

Trần Văn Chiến*
Khoa Điện tử- Viễn Thông, trường Đại học Công Nghệ, Đại học Quốc Gia Hà Nội
Hà Nội, Việt Nam
chienvt197@gmail.com
*

***Tóm tắt*—Robot bốn chân có sự nhanh nhẹn, linh hoạt và ổn định, tuy nhiên, điều khiển chuyển động của chúng luôn là một bài toán khó. Các phương pháp tiếp cận trước đây chủ yếu sử dụng dáng đi được xác định trước dẫn đến hành vi vụng về và không tự nhiên. Trong bài báo này, chúng tôi trình bày một cách tiếp cận hiệu quả sử dụng học tập củng cố sâu với kiến thức tiên nghiệm để tối ưu hóa dáng đi của robot bốn chân. Chúng tôi áp dụng kỹ thuật tối ưu hóa chính sách gần phân tán (DPPO) để tối ưu hóa việc tìm kiếm dáng đi tốt hơn. Nó không yêu cầu mô hình hóa các robot phức tạp, ưu điểm là có tốc độ hội tụ mạng và hiệu quả học tập tốt. Kết quả mô phỏng chứng minh được điều đó so với các phương pháp học tăng cường sâu khác mà không có kiến thức tiên nghiệm. Bên cạnh đó, dáng đi đạt được của chúng tôi có tốc độ cao hơn nhanh hơn 50% so với dáng đi nước kiệu mà không cần tối ưu hóa.**

I. GIỚI THIỆU

Nhìn chung, robot dạng chân có thể được phân loại thành dạng một chân và nhiều chân. Trong bài báo này, chúng tôi tập trung vào vấn đề kiểm soát chuyển động của robot bốn chân.

Có hai cách tiếp cận điển hình để thiết kế một dáng đi mô phỏng theo sinh học: hàm chuyển động và quỹ đạo bước. Những kết quả mô phỏng thể hiện rằng cả hai cách tiếp cận này đều có độ ổn định thấp. Và bởi vì một bộ điều khiển được thiết kế dựa trên một loại địa hình cụ thể, robot có lẽ sẽ không thực hiện tốt trên một địa hình với nhiều dốc và ma sát cao.

Hướng tới vấn đề này, trong bài báo này, chúng tôi đề xuất một cách tiếp cận giúp học một hành vi di chuyển dựa trên kiến thức đã biết trước của một robot bốn chân như Doggo, khi đó bộ điều khiển các tham số thì tự động được huấn luyện với DRL. Cách tiếp cận được đề xuất chỉ cần theo dõi trạng thái bên trong của robot mà không cần xây dựng mô hình động học phức tạp của nó. Kết quả thử nghiệm chứng minh rằng việc di chuyển robot với các chính sách đã học của chúng tôi có thể di chuyển nhanh hơn 1,5 lần so với dáng đi nước kiệu không được tối ưu hóa. Bên cạnh đó, so với DRL mà không sử dụng kiến thức tiên nghiệm, thuật toán của chúng tôi hội tụ cũng nhanh hơn.

Bài báo này được tổ chức như sau: Phần II giới thiệu về DPPO. Phần III trình bày việc triển khai thuật toán cho Doggo. Phần IV đề xuất một phương pháp học tập củng cố sâu dựa trên kiến thức trước đó để tối ưu hóa dáng đi và tốc độ của robot Doggo. Phần V trình bày kết quả thí nghiệm của chúng tôi và Phần VI kết thúc bài báo này.

II. NỀN TẢNG LÝ THUYẾT

[1] DPPO là phiên bản phân tán của PPO, có thể coi là phiên bản gần đúng của TRPO (Tối ưu hóa chinh sách khu vực tin cậy - thứ mà đảm bảo rằng chính sách cập nhật mới không khác xa chính sách cũ hoặc chúng ta có thể nói rằng chính sách mới nằm trong vùng tin cậy của chính sách cũ.

Việc thu thập dữ liệu và tính toán gradient được phân bố thông qua các đối tượng. Chúng tôi đã thử nghiệm với cả cập nhật đồng bộ và không đồng bộ và đã nhận thấy rằng việc trung bình các gradient và áp dụng chúng đồng bộ dẫn đến kết quả tốt hơn trong thực tế.

Mã giả cho thuật toán DPPO được cung cấp dưới đây.

Algorithm 1 Distributed Proximal Policy Optimization (chief)

for $i \in \{1, ..., N\}$ **do**
  for $j \in \{1, ..., M\}$ **do**
    Wait until at least W - D gradients wrt. θ are available average gradients and update global θ
  end for
  for $j \in \{1, ..., B\}$ **do**
    Wait until at least W - D gradients wrt. ϕ are available average gradients and update global ϕ
  end for
end for

W là số lượng đối tượng; D đặt một ngưỡng cho số lượng đối tượng có gradient phải có sẵn để cập nhật các thông số. M, B là số lần lặp con với các cập nhật chính sách và đường cơ sở được cung cấp cho một loạt điểm dữ liệu. T là số điểm dữ liệu được thu thập trên mỗi đối tượng trước khi tính toán cập nhật thông số. K là số bước thời gian để tính toán trả về K-bước và rút ngắn lan truyền ngược qua thời gian (đối với RNN).[2]

Algorithm 2 Distributed Proximal Policy Optimization (worker)

for $i \in \{1, ..., N\}$ **do**
  for $w \in \{1, ..., T/K\}$ **do**
    Run policy π_θ for K timesteps, collecting $\{s_t, a_t, r_t\}$ for t $\in \{(i-1)K, ..., iK-1\}$
    Estimate return $\hat{R}_t = \sum_{t=(i-1)K}^{iK-1} \gamma^{t-(i-1)K} r_t + \gamma^K V_\phi(s_{iK})$
    Estimate advantages $\hat{A}_t$ = $\hat{R}_t$ - $V_\phi(s_t)$
    Store partial trajectory information
  end for
  $\pi_{old} \leftarrow \pi_\theta$
  for $m \in \{1, ..., M\}$ **do**
    $J_{PPO}(\theta)$ = $\sum_{t=1}^{T} \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_old}(a_t|s_t)} \hat{A}_t$ - $\lambda KL[\pi_{old}|\pi_\theta]$ - $\xi max(0, KL[\pi_{old}|\pi_\theta] - 2KL_{target})^2$
    if $KL[\pi_{old}|\pi_\theta] > 4KL_{target}$ **then**
      break and continue with next outer iteration i+1
    end if
    Compute $\nabla_\theta J_{PPO}$
    Send gradient wrt. to θ to chief
    Wait until gradient accepted or dropped; update parameters
  end for
  for $b \in \{1, ..., B\}$ **do**
    $L_{BL}(\phi)$ = -$\sum_{t=1}^{T} (\hat{R}_t - V_\phi(s_t))^2$
    Compute $\nabla_\phi L_{BL}$
    Send gradient wrt. to θ to chief
    Wait until gradient accepted or dropped; update parameters
  end for
  if $KL[\pi_{old}|\pi_\theta] > \beta_{high} KL_{target}$ **then**
    $\lambda \leftarrow \widetilde{\alpha}\lambda$
  else if $KL[\pi_{old}|\pi_\theta] < \beta_{low} KL_{target}$ **then**
    $\lambda \leftarrow \lambda/\widetilde{\alpha}$
  end if
end for

DPPO đạt được hiệu suất tương tự như TRPO và DPPO cân bằng tốt với số lượng đối tượng được sử dụng, điều này có thể làm giảm đáng kể thời gian xử lý. Vì nó hoàn toàn dựa trên gradient nên nó cũng có thể được sử dụng trực tiếp với các mạng lặp lại như được trình bày trong tác vụ Memory Reacher. DPPO cũng nhanh hơn (trong ép xung) so với việc triển khai A3C của chúng tôi khi sử dụng cùng một số lượng đối tượng.

III. TRIỂN KHAI THUẬT TOÁN

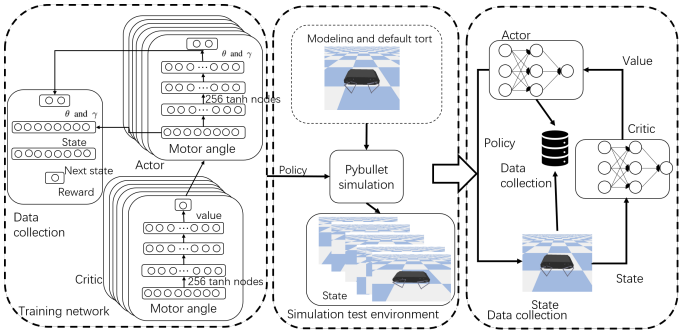
A. *Định nghĩa*

Vấn đề chuyển động của một robot di chuyển bốn chân có thể được mô tả như một quá trình quyết định Markov (MDP). Một MDP được biểu diễn bởi $\{S, A, P, R, \lambda\}$, trong đó S là không gian trạng thái của robot, bao gồm toàn bộ trạng thái khả thi của robot; A là không gian hành động, là một tập hợp của toàn bộ hành động có thể thực hiện của robot;$P : S * A * S \rightarrow [0, 1]$ là hàm xác suất chuyển đổi, cái mà mô hình hóa sự phát triển của các trạng thái dựa trên các hành động; $R : S * A \rightarrow R$ biểu thị phần thưởng được cho sau khi một hành vi và trạng thái; λ thể hiện hệ số suy giảm trong đoạn $[0, 1]$.

Đích đến của học tăng cường là thực hiện một chính sách π, với $\pi : S * A \rightarrow [0, 1]$, một phân bố xác suất hoạt động trên trạng thái.Chính sách tối ưu là một chuỗi các hành động nhằm tối đa hóa lợi nhuận của robot bốn chân. Chính sách tối ưu được hiển thị như π_θ:

$$\pi_\theta = \text{argmax}\, P_{\pi_\theta}[R(s_t, a_t)] \tag{1}$$

Trong đó P_{π_θ} là quỹ đạo trạng thái $\{s_1, s_2, \ldots\ldots s_T\}$ thu được khi chính sách π_θ được sử dụng. Chính sách π_θ muốn lấy được giá trị cực đại của toàn bộ lợi nhuận và quỹ đạo. Cấu trúc mạng được hiển thị như hình 1. DPPO được chứng minh là rất thành công trong việc thu thập các giá trị hành động tối ưu liên tục.



Hình 1: Cấu trúc mạng

Nó là một thuật toán chính sách trực tuyến dựa trên khung làm việc Actor-Critic truyền thống. Mỗi một trong chúng đều được thiết kế bằng một mạng neural. Chúng tôi sử dụng $\hat{A}$ để hiển thị hàm Advantage, biểu thị cho việc hành động chúng ta chọn tốt hơn với những hành động khác chúng ta đã bỏ qua như thế nào, giá trị được cho bởi công thức:

$$\hat{A} = r(s_t, a_t) + \gamma V(s_{t+1} - V(s)) \tag{2}$$

DPPO thu thập tối đa N = 1000 tập, trong môi trường đang được sử dụng bởi chính sách hiện tại. DPPO thu thập kinh nghiệm bao gồm $\{s_t, a_t, s_{t+1}, r_t\}$, trong mỗi vòng lặp DPPO chọn 64 bộ dữ liệu đã được thu thập cho gradient ngẫu nhiên (stochastic gradient descent). DPPO tối ưu mạng tham số đang sử dụng bởi gradient và phần thưởng

$$L(\theta) = E_{\pi_{old}}[\frac{\pi_\theta(a_t|s_t)}{\pi_{old}(a_t|s_t)}\hat{A}] \tag{3}$$

Để giải quyết vấn đề tốc độ học không thể được xác định trong học tăng cường, DPPO sử dụng công thức sau để tối ưu hóa tham số của mạng:

$$L^{CLIP}(\theta) = E_t[min(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)] \tag{4}$$

Trong đó ϵ là siêu tham số, sử dụng phương pháp cắt để giới hạn $r_t(\theta)$ từ $1-\epsilon, 1+\epsilon$. Chúng tôi lấy giá trị cực tiểu của đối tượng đã được cắt và chưa được cắt, nghĩa là chỉ bỏ qua sự thay đổi tỉ lệ xác

suất khi nó làm đối tượng tốt hơn, và chúng tôi sử dụng nó khi nó làm đối tượng xấu hơn.

B. Tối ưu hóa

Robot bốn chân có nhiều dữ liệu trong suốt qua trình huấn luyện và cấu trúc mạng thì phức tạp. Để tăng tốc độ hội tụ DPPO, trong mỗi vòng lặp huấn luyện, chúng tôi chính quy hóa dữ liệu của robot Doggo trong môi trường và truyền vào mạng để huấn luyện. Với hàm kích hoạt tanh, dữ liệu có thể được giữ trong phần nhạy cảm nhất của hàm kích hoạt, điều này có lợi để ngăn không cho gradient biến mất. Chính quy hóa dữ liệu có thể cải thiện tốc độ hội tụ của mạng.

Tại cùng thời điểm đó, chúng tôi điều chỉnh hàm phần thưởng. Khi robot bốn chân gặp tai nạn thay vì hoàn thành nhiệm vụ phía trước đã được chỉ định thì ngừng huấn luyện. Ví dụ, nếu xảy ra tình trạng lật, robot bốn chân quá thấp so với sàn; hoặc cấu trúc chân bị kẹt không bước được. Khi đó tổng giá trị tiền thưởng thu được được nhân với trọng số $\gamma \in [0, 1]$. Nhưng khi robot hoàn thành tác vụ nâng cao được chỉ định và ngừng huấn luyện, trọng số thì không được nhân vào. Bằng cách này, chúng tôi đặt phần thưởng cho chuyển động tịnh tiến bình thường cao hơn phần thưởng khi tai nạn xảy ra. Trong quá trình đào tạo, robot áp dụng chiến lược có giá trị phần thưởng cao hơn để giảm xác suất tai nạn.

IV. ÁP DỤNG HỌC TĂNG CƯỜNG CHO ROBOT BỐN CHÂN NHANH HƠN

Trong phần này, chúng tôi chủ yếu mô tả các phương pháp chúng tôi sử dụng và cách thức làm cho robot bốn chân Doggo chạy nhanh hơn.

A. Tiền kiến thức

Trong phương pháp chúng tôi đề xuất, sử dụng góc quay hiện tại của tất cả tám động cơ làm đầu vào của một mạng neural gồm ba lớp, các lớp được kết nối đầy đủ được thiết lập. Đầu ra của mạng neural là θ và γ. Chúng tôi điều chỉnh kế hoạch chuyển động bằng việc điều chỉnh θ và γ để tối ưu hóa tư thế của robot bốn chân. Mạng neural sử dụng hàm kích hoạt tanh như là một hàm kích hoạt phi tuyến tính để giới hạn θ và γ tới một dải xác định.

Chúng tôi sử dụng hàm $\sin$ để đảm bảo tính tuần hoàn của chuyển động của cơ cấu chân. Điều này đảm bảo dáng đi nước kiệu giống như kiến thức tiên nghiệm và sử dụng mạng neural để tối ưu hóa dáng đi.

Để duy trì tính ổn định của robot Doggo tốt hơn, chúng tôi hy vọng rằng θ và γ không nên quá rộng, để tránh tình huống robot Doggo bị lật hoặc ngã. Sử dụng mạng neural có thể tìm thiểu những chiến lược mà nên được thực hiện trong trường hợp này, nhưng việc tìm hiểu trong tình huống này yêu cầu một lượng lớn tính toán và thời gian, vì thì chúng tôi phải chỉ định giả tạo giá trị cực đại của θ và γ

Bằng phương pháp này, giá trị cực đại của θ và γ thì hạn chế thủ công để tăng tính ổn định của tư thế và gia tốc tốc độ hội tụ của mạng.

B. Mạng học tăng cường sâu

Trong quá trình của học tăng cường, chúng tôi hy vọng rằng robot Doggo sẽ dịch chuyển dọc theo một đường thẳng một cách nhanh chóng, vì thế chúng tôi xác định hàm phần thưởng của học tăng cường.

Ở đây chúng tôi chủ yếu xem xét vị trí của tâm robot, $x_1, y_1, z1$ là các vị trí hiện tại và x_0, y_0, z_0 là các giá trị tọa độ của thời điểm trước. Giá trị trả về thì chủ yếu bao gồm bốn phần, phần thưởng đầu tiên cho hành vi tiến lên của robot, phần thứ hai và thứ ba tương ứng với phạt hành vi xoay trái và xoay phải, lắc trên và dưới của robot, phần thứ tư giới hạn năng lượng cho mỗi lần. Phần này chủ yếu ngăn cản robot hao tổn quá nhiều năng lượng trong mỗi thời điểm, điều này không phù hợp với thế giới vật chất thực. Hàm phần thưởng

này thưởng cho chuyển động thẳng về phía trước nhanh chóng của robot, đồng thời phạt nếu xảy ra hành động xoay trái, xoay phải, hoặc lắc lư của robot, cũng như hạn chế năng lượng để ngăn robot không phù hợp với thế giới thực.

V. KẾT QUẢ

Chúng tôi đã thu thập các tư thế học tăng cường với kiến thức tiên nghiệm, với không cần kiến thức tiên nghiệm và dữ liệu chạy nước kiệu mặc định của robot Doggo. Chúng tôi so sánh kiểu dáng chuyển động của chúng, tốc độ huấn luyện tốc độ thay đổi tình huống có một minh chứng định lượng của cách tiếp cận mà chúng tôi đề xuất, khả năng thích ứng với môi trường được học.

A. Dáng đi của Doggo bốn chân

Chúng tôi đã so sánh các tư thế, như trong hình 2. Có thể thấy rõ rằng dáng đi của robot Doggo có được khi học tăng cường sử dụng kiến thức tiên nghiệm tốt hơn nhiều so vớiviệc không sử dụng. Dáng đi có được bằng cách sử dụng kiến thức tiên nghiệm có độ xoay và độ lắc lư nhỏ hơn, đồng thời tránh va chạm giữa cơ thể hoặc cấu trúc chân của robot Doggo với mặt đất. So sánh với chạy nước kiệu mặc định, tư thế học tăng cường sử dụng kiến thức tiên nghiệm có một sự cải thiện đáng kể trong tốc độ.

B. Tốc độ huấn luyện của học tăng cường

Tiếp theo, chúng tôi so sánh tốc độ huấn luyện của tư thế dùng học tăng cường với kiến thức tiên nghiệm và tư thế dùng học tăng cường mà không có kiến thức tiên nghiệm trong quá trình học, như hình 3. Trong cùng một môi trường huấn luyện, học tăng cường sử dụng kiến thức tiên nghiệm có một giá trị lợi nhuận phần thưởng trả về tốt ở thời điểm bắt đầu, và đạt đến giá trị lợi nhuận cực đại nhanh hơn trong quá trình học hơn là học tăng cường mà không có kiến thức tiên nghiệm. Giá trị cực đại của lợi nhuận thu được cao gấp hai lần so với lợi nhuận thu được khi dùng học tăng cường mà không có kiến thức tiên nghiệm. Điều này thể hiện rằng học tăng cường với kiến thức tiên nghiệm sử dụng thời gian ngắn hơn để tìm kiếm phần thưởng tốt hơn với học tăng cường mà không có kiến thức tiên nghiệm.

C. Tốc độ của Doggo

Cuối cùng, chúng tôi định lượng những thay đổi trong tốc độ di chuyển của cả ba, chúng tôi xác định tốc độ là khoảng cách Doggo di chuyển mỗi hành động mô phỏng.

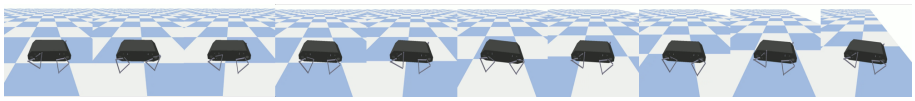
Như trong hình 4. Chúng tôi có thể thấy rõ ràng rằng tốc độ trung bình của tư thế thu được bằng cách áp dụng học củng cố sử dụng kiến thức tiên nghiệm thì cao hơn so với những loại tư thế khác. Cao hơn so với dáng đi nước kiệu mặc định là 50%. Tư thế thu được mà không cần kiến thức tiên nghiệm thì có tốc độ và không ổn định hơn hai dáng đi còn lại.

D. Khả năng thích ứng với môi trường của dáng đi

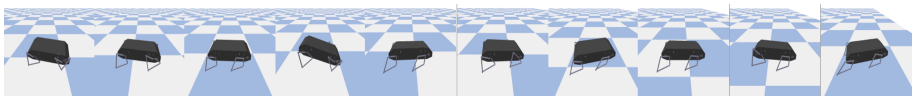
Tư thế chúng tôi đã học được cũng có khả năng thích ứng với môi trường, như trong hình 5. Robot bốn chân trước đây di chuyển trên mặt đất bằng phẳng và robot bốn chân sau di chuyển trên dốc 20 độ. Robot đã hoàn thành hai chuyển động và duy trì sự ổn định của tốc độ.

VI. KẾT LUẬN

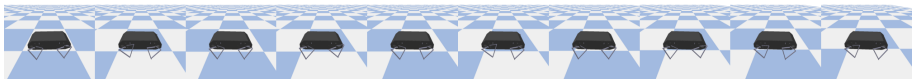
Học tư thế của rô bốt bốn chân có kiến thức tiên nghiệm, các thông số để có được tư thế tốt hơn, được tìm kiếm với hiệu quả cao bằng cách sử dụng phương pháp DPPO. Bởi vì cách tiếp cận này không dựa trên một mô hình robot chính xác, nó rất hữu ích cho những robot khó tạo mô hình. Kết quả mô phỏng cũng chứng minh rằng cách tiếp cận được đề xuất hội tụ nhanh hơn và tư thế được tạo ra di chuyển nhanh hơn 0.5 lần so với một bước chạy nước kiệu nhất định mà không cần



(a) Học tăng cường với kiến thức tiên nghiệm

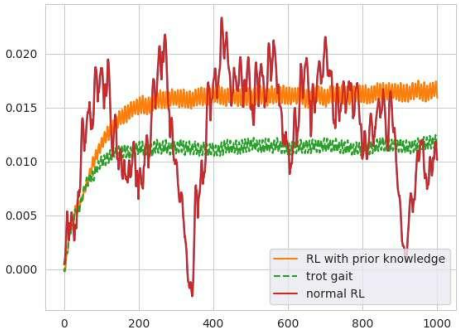


(b) Học tăng cường không kiến thức tiên nghiệm

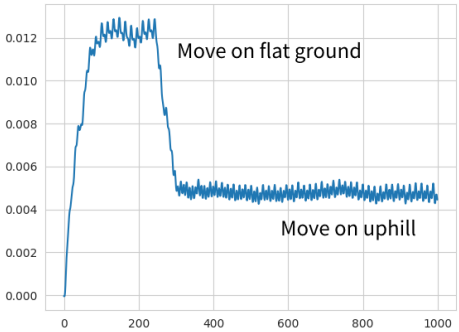


(c) Tư thế chạy nước kiệu mặc định

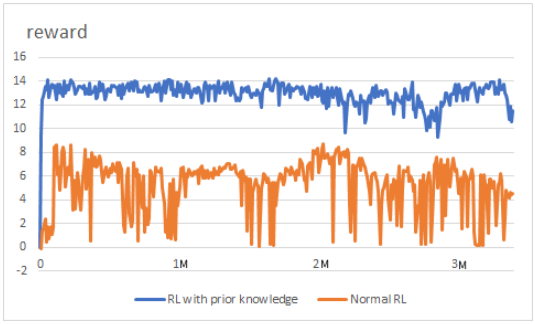
Hình 2: Tư thế của robot Doggo thu được bởi 3 phương pháp. (a) Chạy nước kiệu với học tăng cường có kiến thức tiên nghiệm; (b) Chạy nước kiệu vơi học tăng cường thường và (c) Tư thế chạy nước kiệu mặc định



Hình 4: Tốc độ của ba dáng đi khác nhau. Trục dọc đại diện cho tốc độ và trục ngang đại diện cho số bước mô phỏng.



Hình 5: Tốc độ di chuyển trên mặt đất phẳng và leo dốc.



Hình 3: Đường cong phần thưởng nhận được khi sử dụng phương pháp học tăng cường với kiến thức trước đây về Doggo và đường cong phần thưởng khi sử dụng phương pháp học tăng cường chung. Chúng tôi ngừng huấn luyện khi các tham số ổn định. Trục dọc đại diện cho giá trị trả về và trục ngang đại diện cho số lần lặp.

tối ưu hóa. Cuối cùng, vì cách tiếp cận được đề xuất chỉ cần có kiến thức tiên nghiệm về dáng đi, nên cũng phù hợp với một số robot có chân khác.

TÀI LIỆU THAM KHẢO

[1] N. Heess, B. Dhruva, T., S. Sriram, J. Lemmon, and D. Silver. (2017) Emergence of locomotion behaviours in rich environments. [Online]. Available: https://arxiv.org/abs/1707.02286

[2] N. Heess, D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, Z. W. Tom Erez, M. Ali Eslami, S., M. Riedmiller, and D. Silver. (2018) Realizing learned quadruped locomotion behaviors through kinematic motion primitives. [Online]. Available: https://arxiv.org/pdf/1707.02286v2.pdf