TẠP CHÍ KHOA HỌC & CÔNG NGHỆ ĐẠI HỌC DUY TÂN
DTU Journal of Science and Technology 6(49) (2021) 16-26

TRƯỜNG ĐẠI HỌC DUY TÂN
DUYTAN UNIVERSITY

Ứng dụng phương pháp Học tăng cường xây dựng mô hình xe tự hành

Applying Reinforcement Learning method to building self-driving car model

Nguyễn Dũng[a], Đặng Việt Hùng[a*], Lê Thị Ngọc Vân[a], Trần Huệ Chi[a],
Phan Văn Sơn[a], Nguyễn Quang Vinh[c]
Nguyen Dung[a], Hung Dang Viet[a*], Le Thi Ngoc Van[a], Tran Hue Chi[a],
Phan Van Son[a], Nguyen Quang Vinh[c]

[a]*Khoa Công nghệ Thông tin, Trường Đại học Duy Tân, Đà Nẵng, Việt Nam*
[a]*Faculty of Information Technology, Duy Tan University, 55000, Danang, Vietnam*
[b]*Viện Nghiên cứu và Phát triển Công nghệ Cao, Đại học Duy Tân, Đà Nẵng, Việt Nam*
[b]*Institute of Research and Development, Duy Tan University, Da Nang, 550000, Vietnam*
[c]*Tổng Công ty Điện lực Tp.HCM, Hồ Chí Minh, Việt Nam.*
[c]*Ho Chi Minh city Power Corporation, 700000 Ho Chi Minh city, Vietnam*

(Ngày nhận bài: 03/8/2021, ngày phản biện xong: 02/11/2021, ngày chấp nhận đăng: 02/12/2021)

Tóm tắt

Bài báo áp dụng thuật toán Q-Learning vào huấn luyện xe tự hành và tránh va chạm với chướng ngại vật. Hiện nay xe tự hành là loại xe đang được rất nhiều công ty tham gia nghiên cứu và mong muốn sản xuất đưa vào thực tiễn sử dụng. Q-Learning (Watkins, 1989) là một hình thức Học tăng cường không cần mô hình và có thể được xem như là một phương pháp lập trình động không đồng bộ (DP). Nó cho phép Tác tử khả năng học tập để hành động tối ưu trong môi trường có thuộc tính Markov bằng cách trải nghiệm kết quả của hành động, mà không cần phải xây dựng mô hình xác suất. Bài báo này trình bày quá trình xây dựng chương trình mô phỏng hệ thống xe tự hành dựa vào thuật toán Q-Learning. Kết quả cho thấy thuật toán Q-Learning thành công trong việc xây dựng một kĩ thuật tự huấn luyện để thích nghi với yêu cầu nào đó.

Từ khóa: Q-Learning; Học tăng cường; Markov; xe tự hành.

Abstract

This paper applies Q-Learning algorithm to training a self-driving cars (SDC) model to avoid moving obstacles. Currently, SDC is one of the trendy fields that many companies do research to produce and put into practice. Q-Learning (Watkins, 1989) is a form of model-free reinforcement learning (RL). It can also be viewed as an asynchronous dynamic programming (DP) method. It gives agents an ability to learn how to act optimally in Markov environment by experiencing the results of the action, without building problem model maps. In this work, we build a self-driving car simulation program based on the Q-Learning algorithm. The results show that Q-learning can successfully equipt an agent to self-train for achieving some target.

Keywords: Q-Learning: Reinforcement Learning (RL); Markov; self-driving car.

* *Corresponding Author:* Dang Viet Hung, Faculty of Information Technology, Duy Tan University, 55000, Danang, Vietnam; Institute of Research and Devolopment, Duy Tan University, 55000, Danang, Vietnam.
Email: dangviethung@duytan.edu.vn

1. Giới thiệu

Học máy nghiên cứu cách thức để mô hình hóa bài toán cho phép máy tính tự động hiểu, xử lý và học từ dữ liệu để thực thi nhiệm vụ được giao cũng như cách đánh giá giúp tăng tính hiệu quả. Dưới góc nhìn của trí tuệ nhân tạo, động lực chính của học máy là nhu cầu thu nhận tri thức. Thật vậy, trong nhiều trường hợp, kiến thức chuyên gia khan hiếm hoặc tiến độ thực hiện chậm vì một số nhiệm vụ cần đưa ra quyết định nhanh chóng dựa trên xử lý dữ liệu khổng lồ và thiếu ổn định dẫn đến việc buộc phải dùng đến máy tính. Tom Mitchell, giáo sư nổi tiếng của Đại học Carnegie Mellon University - CMU định nghĩa cụ thể và chuẩn mực hơn về học máy như sau: "Một chương trình máy tính được xem là học cách thực thi một lớp nhiệm vụ thông qua trải nghiệm, đối với thang đo năng lực nếu như dùng năng lực ta đo thấy năng lực thực thi của chương trình có tiến bộ sau khi trải qua trải nghiệm (máy đã học)" [1]

Reinforcement Learning (RL) là một lớp phương pháp thuộc học máy, một lĩnh vực của trí tuệ nhân tạo, có thể trang bị cho một Tác tử (có năng lực tính toán và ra quyết định hành động để thay đổi trạng thái hiện có) một khả năng tự thu nhận thông tin, tự huấn luyện để kết hợp các hành động thành một chuỗi nhằm đạt được mục đích nào đó. Trong các kĩ thuật RL, Q-learning là một giải pháp được ưa chuộng đối với những bài toán có miền trạng thái rời rạc, không cần xây dựng mô hình xác suất chuyển đổi giữa các trạng thái. Bài báo này sẽ triển khai Q-learning cho mô hình xe tự hành trong nỗ lực di chuyển không va chạm với các chướng ngại vật chuyển động[1].

Các bài toán liên quan đến xe tự hành được quan tâm khá nhiều trong thời gian gần đây dù lần đầu được đề xuất giải quyết rất lâu trước đó, vào năm 1925, bởi Francis Houdina [3]. Ông đã sử dụng khái niệm thuật ngữ "a radio-controlled car" (một chiếc xe điều khiển bằng sóng radio). Tuy nhiên nó đã không được thành công như mong đợi. Sau đó đến năm 1969 John McCarthy đã đưa ra khái niệm tự trị và đặt tên là ROBO-CHAUFFevo [4]. Ông là một trong những người sáng lập trí tuệ nhân tạo, ông đã đưa ra thuật ngữ 'Ô tô điều khiển bằng máy tính'. Ý tưởng của ông nhắc đến một chiếc xe có khả năng tự động điều hướng qua các con đường bằng cách sử dụng cùng một góc nhìn như con người có được khi lái xe.

Các nhà khoa học và các nhà sản xuất xe ô tô bắt đầu tiếp cận bài toán, vào năm 2003 Toyota đã ra mắt Prius hybrid [5]. Chiếc xe sử dụng các cảm biến (sensor) và camera hoạt động tốt để hoạt động trong các bãi đỗ xe tự động điều này mang lại niềm phấn khích lớn. Xu hướng sau đó được tiếp nối bởi BMW cũng như cách này hệ thống đỗ xe tự động. Tiếp đó là sự ra đời của Tesla Autopilot vào năm 2015 và Gm Super Cruise – 2017 được hỗ trợ tự động với hệ thống phanh, kiểm soát tốc độ và thay đổi làn đường làm gia tăng sự thoải mái của người lái và hành khách [6]. Và hãng Google mong muốn sẽ ra mắt một chiếc xe tự hành thực sự vào 2021, hoàn toàn không cần sự tương tác của con người [5], hứa hẹn mang lại sự thú vị và thỏa mãn cho việc trải nghiệm sau rất nhiều năm nghiên cứu.

Trong bài báo này, chúng tôi quan tâm đến vấn đề vận hành chuyển động của xe. Bài toán cụ thể được mô phỏng giống như xe được người điều khiển, không gian quan sát được là vùng hạn chế trước mũi xe. Trong quá trình di chuyển nếu người lái xe phát hiện chướng ngại vật thì điều khiển xe sang trái hoặc sang phải để tránh chướng ngại vật. Trong trường hợp xe đi qua đoạn đường cong thì người lái xe điều khiển xe đi theo chiều cong của đường để tránh va vào lề.

Để mô phỏng ý tưởng trên chúng tôi xây dựng hệ thống gồm xe, bốn chướng ngại vật và

đường biên giới hạn. Bốn chướng ngại vật chuyển động theo một đường tròn với bán kính định trước. Đường biên là đường giới hạn phạm vi chuyển động của xe. Xe được phép lựa chọn một trong ba hành động đó là: đi thẳng, rẽ trái và rẽ phải. Xe sẽ tự huấn luyện chuyển động sao cho không va chạm vào chướng ngại vật và đường biên.

2. Cơ sở lý thuyết

2.1. Thuộc tính Markov [2]

Trong bài toán quyết định Markov, Tác tử ra quyết định do một tín hiệu từ môi trường gọi là trạng thái của môi trường. Ta định nghĩa thuộc tính môi trường và các tín hiệu trạng thái của chúng là thuộc tính Markov.

Trạng thái được hiểu là bất cứ thông tin gì có ích với Tác tử, giả thiết trạng thái được đưa ra bởi một số hệ thống tiền xử lý của môi trường. Để đơn giản biểu thức toán học, chúng ta giả sử tập các trạng thái và các mục tiêu là hữu hạn. Quan sát cách thức một môi trường tổng quát có thể đáp ứng tại thời điểm *t+1* đối với hành động được thực hiện tại thời điểm *t*. Trong hầu hết các trường hợp, nguyên nhân của sự đáp ứng này có thể phụ thuộc vào mọi thứ đã xảy ra trước đó. Khi đó biến động của môi trường có thể được định nghĩa bằng cách đặc tả xác suất phân bố khả năng như sau:

$$\Pr\left\{ s_{t+1} = s',\ r_{t+1} = r \mid s_t,\ a_t, r_t, s_{t-1}, a_{t-1}, ..., r_1, s_0,\ a_0 \right\} \quad (1)$$

với mọi *s', r* và mọi giá trị có thể của các sự kiện trước $s_t,\ a_t, r_t, s_{t-1}, a_{t-1}, ..., r_1, s_0,\ a_0$.

Nếu tín hiệu trạng thái có thuộc tính Markov thì đáp ứng của môi trường tại thời điểm *t+1* chỉ phụ thuộc vào trạng thái và hành động tại thời điểm *t*, trong trường hợp này, biến động của môi trường được thể hiện qua hàm:

$$\Pr\left\{ s_{t+1} = s',\ r_{t+1} = r \mid s_t,\ a_t \right\}, \quad (2)$$

Nói cách khác, một trạng thái có thuộc tính Markov (là một trạng thái Markov) khi và chỉ khi giá trị ở hai biểu thức (1) và (2) bằng nhau với mọi *s', r* và $s_t,\ a_t, r_t, s_{t-1}, a_{t-1}, ..., r_1, s_0,\ a_0$. Trong trường hợp này môi trường cũng được gọi là có thuộc tính Markov.

Nếu một môi trường có thuộc tính Markov thì biến động tại mỗi bước của nó sẽ cho phép dự đoán trạng thái và mục tiêu kỳ vọng tiếp theo được đưa ra từ trạng thái và hành động hiện tại. Bằng cách lặp phương trình (2) này, chúng ta có thể dự đoán tất cả các trạng thái và mục tiêu kỳ vọng trong tương lai mà chỉ với kiến thức từ trạng thái hiện tại trong thời điểm hiện tại. Các trạng thái Markov cung cấp khả năng tốt nhất cho việc lựa chọn hành động, khi đó chính sách tốt nhất cho việc lựa chọn hành động sẽ là hàm của một trạng thái Markov.

Nhiều trường hợp trong RL khi tín hiệu trạng thái không có thuộc tính Markov, chúng ta cũng sẽ xấp xỉ trạng thái này thành trạng thái Markov vì chúng ta luôn mong muốn trạng thái là tốt để dự đoán hàm mục tiêu cũng như việc lựa chọn hành động trong tương lai. Với tất cả những lý do đó, cách tốt nhất là xem trạng thái tại mỗi bước thời gian như là một xấp xỉ của trạng thái Markov.

Thuộc tính Markov là rất quan trọng trong các bài toán quyết định Markov vì các quyết định và các giá trị được giả thiết chỉ là hàm phụ thuộc vào trạng thái hiện tại. Giả thiết này không có nghĩa là áp dụng hoàn toàn cho mọi tình huống RL kể cả những tình huống không thỏa mãn Markov. Tuy nhiên lý thuyết phát triển cho các thuộc tính Markov vẫn giúp chúng ta có thể hiểu được hành vi của các giải thuật

RL và các giải thuật thì vẫn có thể áp dụng thành công cho mọi nhiệm vụ với các trạng thái không thỏa mãn Markov.

Với giả thiết như vậy, tương tác giữa Tác tử và môi trường có thể được mô hình dưới dạng bài toán quyết định Markov. Việc tìm kiếm sách lược điều khiển tối ưu trong các bài toán quyết định Markov tương ứng với những tiêu chí tối ưu khác nhau dẫn tới việc xây dựng các phương trình tối ưu Bellman và các thuật toán quy hoạch động. Thông thường, phương pháp quy hoạch động dùng để giải các phương trình tối ưu Bellman khi biết các thuộc tính thống kê của môi trường. Khác với quy hoạch động, phương pháp RL tìm kiếm trực tiếp các chính sách quyết định tối ưu từ các giá trị phản hồi thu nhận được trong các quá trình tương tác với môi trường và trạng thái của môi trường.

2.2. Quy trình quyết định Markov

Một quy trình quyết định Markov [2] là một tập gồm 5 thành phần dữ liệu: $(S, A, P(\cdot,\cdot), R(\cdot,\cdot), \gamma)$. Trong đó,

- S là một tập hữu hạn các trạng thái;

- A là một tập hữu hạn các hành động (ngoài ra, $A(s)$ là tập hữu hạn các hành động có sẵn từ trạng thái s);

- $p(s' \mid s, a) = \Pr(s_{t+1} = s', s_t = s, a_t = a)$ là xác suất thực hiện hành động a trong trạng thái s tại thời gian t sẽ dẫn đến trạng s' tại thời gian $t+1$;

- $r(s, a, s')$ là phần thưởng trực tiếp (hoặc phần thưởng trực tiếp mong đợi) nhận được sau khi chuyển tiếp sang trạng thái s' từ trạng thái s nếu thực hiện hành động a;

- $\gamma \in [0,1)$ là hệ số chiết khấu, sẽ đại diện cho sự khác biệt quan trọng giữa các phần thưởng tương lai và các phần thưởng hiện tại.

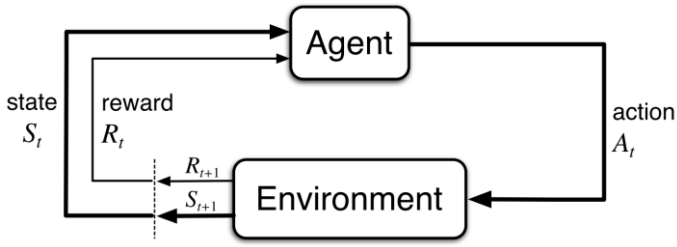
Bài toán cốt lõi của quy trình quyết định Markov đó là tìm một "chính sách" π mà xác định phương thức lựa chọn hành động khi ở trong trạng thái s gọi là $\pi(s)$ sao cho tối đa hóa hàm tích lũy các phần thưởng ngẫu nhiên:

$$\sum_{t=0}^{\infty} \gamma^t r_{a_t}(s_t, s_{t+1}), \text{ (trong đó ta chọn } a_t = \pi(s_t)) \quad (3)$$

2.3. Phương pháp Học tăng cường

Phương pháp Học tăng cường [2] (RL) là một lĩnh vực con của học máy, nghiên cứu cách thức một *tác tử* trong một *môi trường* nên chọn thực hiện các *hành động* nào để cực đại hóa một khoản *thưởng* nào đó về lâu dài. Các thuật toán RL cố gắng tìm một *chính sách* ánh xạ các *trạng thái* của thế giới tới các hành động mà *Tác tử* nên chọn trong các trạng thái đó. *Môi trường* thường được biểu diễn dưới dạng một tập các trạng thái hữu hạn, và các thuật toán RL cho ngữ cảnh này có liên quan nhiều đến các kỹ thuật quy hoạch động. Khác với học có giám sát, trong RL không có các cặp dữ liệu vào/kết quả đúng, các hành động gần tối ưu cũng không được đánh giá đúng sai một cách tường minh. Hơn nữa, ở đây hoạt động được quan tâm, trong đó việc tìm kiếm sự cân bằng giữa khám phá (trạng thái chưa trải nghiệm) và khai thác (trạng thái đã biết).

Một cách hình thức, mô hình RL bao gồm: Tập các trạng thái của môi trường S, tập các hành động A, tập các khoản "thưởng" R với giá trị vô hướng và Tác tử (agent), một chương trình máy tính hoạt động như một đối tượng trong thế giới thực.



Hình 1. Mô hình RL [2]

Cụ thể, Tác tử tương tác với môi trường ở một chuỗi các bước thời gian rời rạc, $t=0,1,2,3,...$ Ở mỗi bước thời gian t, Tác tử nhận được trạng thái của nó là $s_t \in S$ và tập các hành động có thể $A(s_t)$. Nó chọn thực hiện một hành động $a_t \in A(s_t)$ và nhận được từ môi trường trạng thái mới s_{t+1} và một khoản thưởng r_{t+1}.

Ở mỗi bước thời gian, Tác tử thực hiện một ánh xạ từ trạng thái đến xác suất lựa chọn một hành động sẵn có. Việc lựa chọn này được gọi là chính sách của Tác tử và được ký hiệu π_t hay $\pi_t(a|s)$ là xác suất mà $a_t = a$ nếu $s_t=s$. Phương pháp RL chỉ ra cách Tác tử thay đổi chính sách của nó như là kết quả từ kinh nghiệm của nó học được. Mục tiêu của Tác tử là tối đa tổng khoản thưởng tích lũy nhận được trong thời gian dài.

Do đó, RL đặc biệt thích hợp cho các bài toán có sự được mất giữa các khoản thưởng ngắn hạn và dài hạn. RL đã được áp dụng thành công cho nhiều bài toán, trong đó có điều khiển robot, điều vận thang máy, viễn thông, các trò chơi backgammon và cờ vua.

2.4. Các thành phần trong học tăng cường

Ngoài tác nhân và môi trường, phương pháp RL còn có các thành phần chính gồm: Chính sách, tín hiệu thưởng, hàm giá trị và tùy chọn một mô hình môi trường [2].

2.4.1. Chính sách

Chính sách là phương thức xác định hành vi của Tác tử ở một thời điểm nhất định. Nói chung, chính sách là một ánh xạ từ các trạng thái của môi trường đối với những hành động được thực hiện khi ở những trạng thái đó.

Ở mỗi bước thời gian, Tác tử thực hiện một ánh xạ từ trạng thái đến xác suất lựa chọn một hành động sẵn có. Việc lựa chọn này được gọi là chính sách của Tác tử và được ký hiệu π_t hay $\pi_t(a|s)$ là xác suất mà $A_t = a$ nếu $S_t=s$. Phương pháp RL chỉ ra cách Tác tử thay đổi chính sách của nó như là kết quả từ kinh nghiệm của nó học được.

2.4.2. Hàm phản hồi

Trong RL, mục đích hoặc mục tiêu của Tác tử chính là tín hiệu thưởng đặc biệt từ môi trường đến Tác tử. Ở mỗi bước thời gian, giá trị thưởng, hay còn gọi là giá trị phản hồi, là một số thực $r_t \in \mathbb{R}$. Một cách không chính thức, mục tiêu của Tác tử là tối đa hóa tổng giá trị thưởng nhận được. Điều này có nghĩa là việc tối đa hóa không chỉ đối với giá trị thưởng tức thời mà là phần thưởng tích lũy trong một thời gian dài.

Mục đích của Tác tử là cực đại hóa các mục tiêu được tích luỹ trong tương lai. Giá trị phản hồi r_t được biểu diễn dưới dạng hàm số đối với các mục tiêu. Trong các bài toán quyết định Markov, hàm phản hồi sử dụng biểu thức dạng tổng. Các nhà nghiên cứu đã tìm ra ba biểu diễn thường được sử dụng của hàm phản hồi:

Với những bài toán này ta có chuỗi các hành động là vô hạn. Một hệ số suy giảm γ (hệ số chiết khấu), $0 \le \gamma \le 1$ được đưa ra và hàm phản hồi được biểu diễn dưới dạng tổng của các giá trị mục tiêu giảm dần:

$$G_t = r_{t+1} + \gamma^1 r_{t+2} + \gamma^2 r_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \quad (4)$$

Hệ số γ cho phép xác định mức độ ảnh hưởng của những bước chuyển trạng thái tiếp theo đến giá trị phản hồi tại thời điểm đang xét. Giá trị của γ cho phép điều chỉnh giai đoạn Tác tử lấy các hàm tăng cường. Nếu γ gần 0, thì Tác tử chỉ xem xét mục tiêu gần nhất, giá trị γ càng gần với 1 thì Tác tử sẽ quan tâm đến các mục tiêu xa hơn trong tương lai.

Như vậy, thực chất bài toán quyết định Markov trong trường hợp này chính là việc lựa chọn các hành động để làm cực đại biểu thức (4).

2.4.3. Hàm giá trị

Trong mọi trạng thái s_t, một Tác tử lựa chọn một hành động dựa theo một chính sách điều khiển, $\pi : a_t = \pi(s_t)$. Hàm giá trị tại một trạng thái của hệ thống được tính bằng kỳ vọng toán học của hàm phản hồi theo thời gian. Hàm giá trị là hàm của trạng thái và xác định mức độ thích hợp của chính sách điều khiển π đối với Tác tử khi hệ thống đang ở trạng thái *s*. Hàm giá trị của trạng thái *s* trong chính sách π được tính như sau:

$$v_\pi(s) = E_\pi\{G_t \mid s_t = s\} = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s\right] \quad (5)$$

Trong đó, $E_\pi[.]$ biểu thị giá trị kỳ vọng của một biến ngẫu nhiên mà Tác tử theo chính sách π và *t* là bước thời gian bất kỳ.

Bài toán tối ưu bao gồm việc xác định chính sách điều khiển π^* sao cho hàm giá trị của trạng thái hệ thống đạt cực đại sau một số vô hạn hoặc hữu hạn các bước.

$$\pi^* = \{\pi_0(s_0), \pi_1(s_1), \ldots, \pi_{N-1}(s_{N-1})\}, \quad (6)$$

Sử dụng các phép biến đổi:

$$v_\pi(s) = E_\pi\{G_t \mid s_t = s\}$$

$$= E_\pi\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s\right\}$$

Một chính sách tối ưu, kí hiệu $\pi*$, sẽ là cho giá trị thưởng lớn nhất, hay:

$$v_{\pi^*}(s) \geq v_\pi(s) \quad (7)$$

$$\pi^* = \arg\max_\pi \{v_\pi(s)\} \quad (8)$$

Để đơn giản chúng ta viết $v_* = v_{\pi^*}$. Hàm giá trị tối ưu của một trạng thái tương ứng với chính sách tối ưu là:

$$v_*(s) = \max_\pi \{v_\pi(s)\} \quad (9)$$

Đây là phương trình tối ưu Bellman (hoặc phương trình của quy hoạch động). Tóm lại v_π là hàm giá trị trạng thái cho chính sách π. Giá trị của trạng thái kết thúc thường bằng 0. Tương tự, định nghĩa $Q_\pi(s,a)$ là giá trị của việc thực hiện hành động *a* trong trạng thái *s* dưới chính sách điều khiển π, được tính bằng kỳ vọng toán học của hàm phản hồi bắt đầu từ trạng thái *s*, thực hiện hành động *a* trong chính sách π:

$$Q_\pi(s,a) = E_\pi\{G_t \mid s_t = s, a_t = a\} = E_\pi\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s, a_t = a\right\} \quad (10)$$

Q_π được gọi là hàm giá trị hành động cho chính sách π. Và các hàm giá trị v_π, Q_π có thể được ước lượng từ kinh nghiệm. Đối với phương pháp Q-learning, công thức cập nhật được triển khai cụ thể như sau:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left[R_{t+1} + \gamma \max Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)\right] \quad (11)$$

Với γ là hệ số chiết khấu cho giá trị Q của bước trước và α là hệ số học để điều chỉnh mức độ tối ưu của quá trình học. Trong đó, tỉ lệ học tập $\alpha \in [0,1]$ xác định các thông tin mới thu được sẽ ghi đè lên các thông tin cũ, khi $\alpha = 1$ sẽ làm cho Tác tử chỉ xem xét các thông tin gần nhất. Hệ số chiết khấu $\gamma \in [0,1]$ xác định tầm quan trọng của các phần thưởng trong tương lai. Khi $\gamma = 0$, sẽ làm cho Tác tử tham lam bằng cách chỉ xem xét phần thưởng hiện tại, trong khi γ gần 1 sẽ làm cho Tác tử phấn đấu cho một phần thưởng cao dài hạn.

3. Thực nghiệm và kết quả

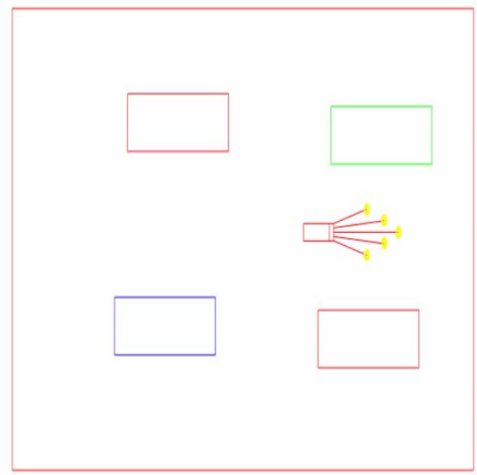
Nhóm chúng tôi đã triển khai bài toán bằng phương pháp Q-learning theo các bước như sau: Xây dựng bộ trạng thái, xác định tập hành động, khởi tạo bảng Q.

3.1. Xây dựng chương trình thực nghiệm.

Trong hệ thống này chúng tôi xây dựng chướng ngại động và chướng ngại vật tĩnh. Chướng ngại vật động là bốn vật thể chuyển động quanh đường tròn có bán kính cho trước. Chuyển động của chướng ngại vật có thể tùy ý, việc chọn chuyển động tròn chỉ nhằm tránh tình huống các chướng ngại vật này va chạm với nhau. Chướng ngại vật tĩnh là các đường biên giới hạn, ngoài ra, để xác định trạng thái xe có va chạm vào chướng ngại vật hay không. Khi xe di chuyển nếu đầu xe chạm vào một trong các chướng ngại vật được xem là va chạm.

3.1.1. Xây dựng bộ trạng thái và hàm phản hồi tương ứng

Việc đánh giá và tổ chức đúng bộ trạng thái cho hệ thống là việc làm rất quan trọng, nó quyết định đến việc thành công và thất bại khi triển khai hệ thống và nó giảm sự phụ thuộc của hệ thống vào môi trường. Trong bài toán này nếu chúng tôi chọn bộ trạng thái là tọa độ của xe và tọa độ của chướng ngại vật thì việc thay không gian hoặc bổ sung chướng ngại vật sẽ ảnh hưởng đến kết quả huấn luyện. Do đó, chúng tôi xác định bộ trạng thái của hệ thống là bộ trạng thái của các **sensor**. Các sensor này nhận giá trị 1 và 0 ứng với trường hợp dò được tín hiệu chướng ngại vật ở gần và không có tín hiệu. Hình 3 mô tả mô hình vật lý của việc huấn luyện xe tự hành, bao gồm năm sensor trang bị trước mũi xe. Nếu một phần nào đó của một chướng ngại vật nằm trong tầm phát hiện của sensor (đoạn nối các điểm tròn với xe) thì giá trị trả về của sensor là 1, ngược lại là 0.



Hình 3. Mô hình vật lý hệ thống xe tự hành

Để xây dựng bộ trạng thái trên không gian nhiều chiều chúng tôi xây dựng bộ trạng thái trên các tập khác nhau. Trạng thái s được xác định: s(u, v, x, y, z) trong đó u, v, x, y, z ∈{0, 1}, ứng với 5 sensor chướng ngại trên xe, với hai trạng thái là có phát hiện chướng ngại (giá trị 1) và không phát hiện chướng ngại (giá trị 0). Trạng thái s và s' trong công thức (11) sẽ là s(ttd(1), (ttd(2), ttd (3), ttd(4), ttd(5)) (trạng thái

của hệ thống trước khi thực hiện hành động và s(tts(1), tts(2), tts(3), tts(4), tts(5)) (trạng thái của hệ thống sau khi thực hiện hành động) một cách tương ứng.

Trong tập trạng thái, Tác tử cần biết phần thưởng cho mỗi trạng thái. Đối với bài toán này, chúng tôi xây dựng hệ phần thưởng R âm (phạt) nếu xe xảy ra trạng thái va chạm với chướng ngại vật. Hàm phản hồi trả về giá trị 0 nếu sau khi thực hiện hành động, đầu xe hướng đến một vùng không gian không có chướng ngại vật, nghĩa là không sensor nào phát hiện chướng ngại vật, hay s(u, v, x, y, z) = θ =[0 0 0 0 0]^T. Ngược lại hàm phản hồi nhận giá trị -1 (giá trị phạt) nếu như sau khi thực hiện hành động đầu xe không thoát khỏi vùng chứa chướng ngại vật, hay s(u, v, x, y, z) ≠θ. Khi có va chạm, chương trình sẽ dừng episode đó lại và bắt đầu một episode mới.

3.1.2. Xây dựng tập hành động và bảng Q

Đối với hệ thống xe tự hành, để chuyển động, xe thực hiện một trong ba hành động đó là: Đi thẳng, rẽ trái, rẽ phải. Bộ hành động được mã hóa thành ba giá trị tương ứng: Giá trị 1 tương ứng cho hành động rẽ trái, giá trị 2 cho hành động đi thẳng và giá trị 3 cho hành động rẽ phải.

Với 3 hành động được lựa chọn trong mỗi trạng thái, bảng Q sẽ có chiều tương ứng là (s_num) x (a_num) = 2^5 x 3. Bảng Q thực chất là nơi chứa kiến thức của Tác tử sau khi học. Hệ thống dựa vào kết quả lưu trong bảng Q để đưa ra quyết định thực hiện hành động trong tập hành động nhằm đạt được mức thưởng tối đa trong dài hạn. Độ lớn của bảng Q được xác định dựa vào tích số của số lượng hành động với số lượng trạng thái.

3.1.3. Xây dựng hệ vật lý cho bài toán

+ Hàm mô tả hoạt động của xe.

```
function [xp]=DoAction(action,x,V,dt,GocquayXe,tts)
    if action ==1
        GocquayXe=-GocquayXe;
    elseif action ==2;
        GocquayXe=0;
    end
     xp(4)=x(4)+GocquayXe*3.14159/180;
     xp(2) = x(2) + V*cos(xp(4))*dt;
     xp(3) = x(3) + V*sin(xp(4))*dt;
    xp(1)=x(1)+1;
R = [cos(xp(4)) -sin(xp(4))
        sin(xp(4)) cos(xp(4))];
       xe_new = R*xe;
     xe_new(1,:)=xe_new(1,:)+xp(2);
     xe_new(2,:)=xe_new(2,:)+xp(3);
end
```

+ Hàm mô tả hoạt động của các chướng ngại vật động

```
Function Action_barr(phi)
       phi=phi/50;
       RR = [cos(phi) -sin(phi); sin(phi) cos(phi)];
       Barrnew1 = (barr + RR*[ra 0]'*ones(1,5));
       phi = phi+pi/2;
       RR = [cos(phi) -sin(phi); sin(phi) cos(phi)];
       Barrnew2 = (barr + RR*[ra 0]'*ones(1,5));
       phi = phi+pi/2;
       RR = [cos(phi) -sin(phi); sin(phi) cos(phi)];
       Barrnew3 = (barr + RR*[ra 0]'*ones(1,5));
       phi = phi+pi/2;
       RR = [cos(phi) -sin(phi); sin(phi) cos(phi)];
       Barrnew4 = (barr + RR*[ra 0]'*ones(1,5));
end
```

Trong đó: Mảng x chứa vị trí của xe trước khi thực hiện hành động, mảng xp chứa vị trí của xe sau khi thực hiện hành động, *ra* là bán kính đường tròn mà các chướng ngại vật quay quanh, V là vận tốc xe, và dt là bước thời gian xe thực hiện hành động và nhận phản hồi từ phần thưởng cũng như sự thay đổi môi trường. Chương trình tạo ra 4 chướng ngại vật hình chữ nhật Barrnewi, được sinh ra từ 4 phép quay hơn kém nhau π/2. Chú ý rằng 4 chướng ngại vật này còn bị tác động bởi một phép quay khác theo thời gian để không bị đứng yên.

3.1.4. Chương trình chính sử dụng phương pháp Q_Leaning

Các bước của thuật toán có thể được tóm tắt như sau:

1. Khởi tạo bảng giá trị Q, Q(s,a).

2. Quan sát trạng thái hiện tại s.

3. Lựa chọn hành động a cho trạng thái dựa vào một trong các chiến lược lựa chọn hành động (ε-greedy).

4. Thực hiện hành động và quan sát giá trị r cũng như trạng thái mới s'.

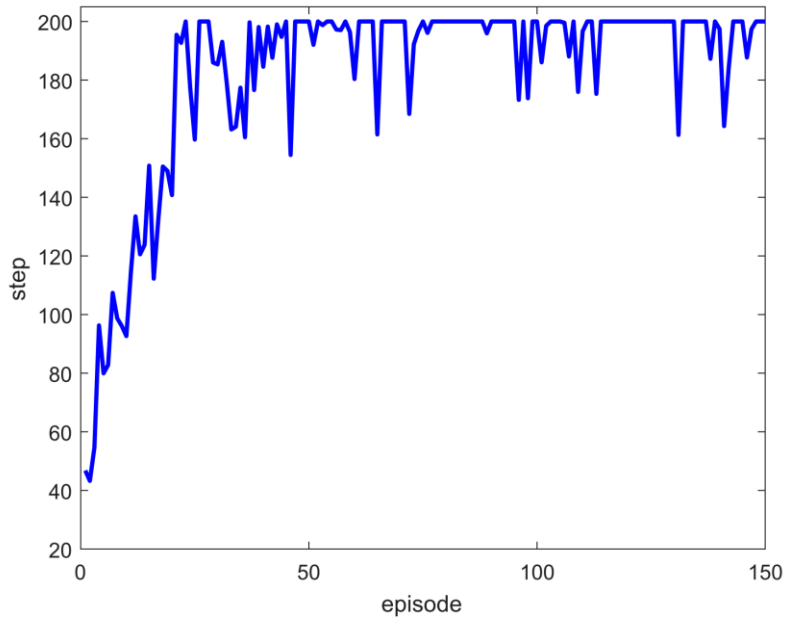
5. Cập nhật giá trị Q cho trạng thái sử dụng giá trị tăng cường được quan sát và giá trị tăng cường lớn nhất có thể cho trạng thái tiếp theo với công thức (11).

6. Thiết lập trạng thái đến trạng thái mới. Quay lại bước 2 nếu số lần lặp tối đa chưa đạt đến.

Trong quá trình huấn luyện, khi Tác tử rơi vào trạng thái nào thì nhận được giá trị phản hồi của trạng thái đó. Sau đó chương trình thực hiện lặp lại quá trình thực hiện hành động và cứ sau mỗi bước thực hiện hành động thì lại điều chỉnh lại giá trị trong bảng Q ứng với trạng thái cũ và hành động vừa được thực hiện. Xe sẽ được huấn luyện trong nhiều lần học gọi là số episode. Trong mỗi episode, xe được phép ra quyết định và nhận phản hồi thay đổi trạng thái nhiều lần để tự thu nhận thông tin và học cách ra quyết định đúng. Số lần ra quyết định và nhận phản hồi này được gọi là số step, được xác định giá trị trần bởi người lập trình. Tuy nhiên, số step trong mỗi episode có thể ít hơn nếu quá trình huấn luyện episode đó có va chạm xảy ra.

3.2. Kết quả chạy chương trình với Q-learning

Đối với môi trường vật lý mô phỏng trên, Tác tử là xe tự hành sẽ tự huấn luyện kĩ năng tránh vật thể bằng phương pháp Q-learning mô tả trong phần 2.3 và 3.1. Các thông số tỉ lệ học và hệ số chiếc khấu được chọn là $\boldsymbol{\alpha = 0.8}$ và $\boldsymbol{\gamma = 0.9}$. Ngoài ra, luật lựa chọn hành động được sử dụng là $\boldsymbol{\varepsilon}$-greedy. Số bước lặp steps được thiết lập là 200 và số lần huấn luyện episode là 150. Chương trình được chạy 20 lần, lấy kết quả trung bình và biểu diễn trong Hình 3.



Hình 3. Kết quả huấn luyện xe tự hành

Hình 3 mô tả kết quả huấn luyện xe tự hành, cụ thể là số step trung bình đạt được qua từng episode của 20 lần thực hiện ngẫu nhiên. Quan sát Hình 3 cho thấy trong 2 episode đầu tiên, xe chỉ đi được dưới 45 step. Từ episode thứ 3 đến episode thứ 21, khả năng tránh vật thể của xe liên tục được cải thiện và đạt được số step cao dần. Tuy đạt được mức step tối đa bắt đầu từ episode thứ 21, mức tối đa này không phải lúc nào cũng đạt ở các episode tiếp theo. Mức này càng lúc càng có xác suất đạt cao hơn khi episode tăng.

Nguyên nhân dẫn đến các kết quả trên là: Ở những episode đầu tiên, xe chưa có kiến thức tránh chướng ngại vật tĩnh và động, nên gây ra va chạm rất sớm. Lưu ý rằng chương trình sẽ dừng episode nếu xảy ra va chạm và vận tốc xe là không đổi là 50m/s. Do đó số step thấp ứng với Hình 3 cho thấy xe không đáp ứng tốt với việc tránh chướng ngại. Càng huấn luyện, nghĩa là ứng với các episode lớn hơn, số step tăng cao dần theo thời gian. Điều này có nghĩa rằng mặc dù xe di chuyển liên tục trong môi trường có các hướng ngại vật tĩnh và động, xe đã tự biết điều chỉnh kiến thức học hỏi được và ra các quyết định càng lúc càng chính xác, tránh được các vật thể. Giá trị step đạt bão hòa ở mức 200 với xác suất cao dần cho thấy xe có thể vận hành tốt trong môi trường phức hợp và đạt được số step tối đa trong các lần huấn luyện về sau. Kết quả này chứng tỏ rằng giải thuật đã được cài đặt thành công. Với sự trang bị giải thuật Học tăng cường Q-learning, xe tự hành đã có thể tự huấn luyện để đạt được kĩ năng tránh vật thể tĩnh và động.

Một điểm lưu ý nữa là với hệ vật lý môi trường được xây dựng phức tạp, vận tốc xe là cao và không đổi, góc quay của xe chỉ giới hạn trong ba lựa chọn, xe thỉnh thoảng rơi vào những tình huống không thể tránh được va chạm. Đó là lý do ở các episode lớn, vẫn tồn tại va chạm sớm trước khi số step lớn nhất được đạt tới.

4. Kết luận

Bài báo đã ứng dụng được phương pháp Học tăng cường và thuật toán Q-learning để xây dựng xe tự hành, xây dựng chương trình hoàn chỉnh áp dụng cho giải thuật Q-learning. Trong thực tế, xe tự hành không hoàn toàn được huấn luyện theo kiểu tự rút kinh nghiệm trong môi trường vật lý thực vì các va chạm có thể dẫn

đến hỏng hóc ngay lập tức. Thay vào đó, mô hình mô phỏng được xây dựng, huấn luyện xe trong môi trường mô phỏng và sử dụng kiến thức này cho Tác tử trong môi trường thực. Do đó, chương trình mô phỏng và kết quả có ý nghĩa nhất định đối với vấn đề xe tự hành thực tế. Kết quả chạy chương trình do nhóm xây dựng cho thấy tính đúng đắn của giải pháp Q-learning áp dụng vào việc xây dựng một Tác tử tự hành có khả năng tránh được các vật thể tĩnh và chuyển động.

Tài liệu tham khảo

[1] Mitchell, T. (1997), *Machine Learning*, McGraw Hill. ISBN 0-07-042807-7.

[2] Richard S. Sutton and Andrew G. Barto. (2016), *Reinforcement Learning: An Introduction*, The MIT Press Cambridge, Massachusetts London, England.

[3] George Heinzelman. *(*2019), *Autonomous Vehicles, Ethics of Progress,* Ethical Issues in Technology, Prof. Jason Bronowitz Arizona State University.

[4] Ronan Glon and Stephen Edelstein. (2020), *The history of self-driving cars.* Link: https://www.digitaltrends.com/cars/history-of-self-driving-cars-milestones/

[5] Kelsey Piper (2020), *It's 2020. Where are our self-driving cars?*. Link: https://www.vox.com/future-perfect/2020/2/14/21063487/self-driving-cars-autonomous-vehicles-waymo-cruise-uber

[6] Henry Payne (2020), *GM working on semi-autonomous Ultra Cruise to operate on all roads*, The Detroit News. Link: https://www.detroitnews.com/story/business/autos/general-motors/2020/05/20/gm-working-semi-autonomous-ultra-cruise-operate-all-roads/5227248002/