Học Tăng Cường theo chương trình cho xe tự lái trong môi trường mô phỏng Carla

Nguyễn Đam Trường[*] *Nguyễn Hoàng Vũ*[**] *Nguyễn Duy Hậu*[***]

**Email: 18021333@vnu.edu.vn*

***Email: 18021435@vnu.edu.vn*

****Email: 18020463@vnu.edu.vn*

Tóm tắt – Những tiến bộ của các phương pháp học có giám sát và học tăng cường đã mang lại sự thay đổi lớn để áp dụng các phương pháp vào trong việc vận hành một chiếc xe tự lái. Tuy nhiên còn nhiều thách thức trong quá trình vận hành một chiếc xe tự lái trong một môi trường nhiều tác nhân và nhiều tác động của ngoại cảnh. Các thuật toán học tập có giám sát có thể được xây dựng trên một lượng lớn các dữ liệu để chuyển đổi sang các môi trường mới, mặc dù vậy thì việc thì việc thu thập dữ liệu mới cho mỗi một môi trường mới rất không thực tế và tốn kém. Học tăng cường có thể giảm thiểu vấn đề phụ thuộc vào dữ liệu. Ngoài ra khi sử dụng môi trường mô phỏng để lập trình xe tự lái có thể chủ động đưa ra một số các vấn đề cụ thể để xe có thể giải quyết, không chỉ có vậy môi trường mô phỏng còn giúp tiết kiệm chi phí và thời gian đào tạo của một chiếc xe tự lái.

Từ khóa – Học tăng cường, Xe tự lái, Học theo chương trình, Điều khiển xe, Môi trường Carla

I. Giới thiệu

Trong vài năm tới, việc xe tự lái phổ biến là một điều tất yếu của xu hướng. Các phương tiện tự lái sẽ di chuyển từ điểm này đến điểm khác đảm bảo an toàn, với thời gian nhanh hơn. Nhờ có các xe tự lái phát triển mà số lượng tử vong do tai nạn giao thông giảm khi gặp phải những yếu tố bất ngờ hoặc liên quan đến tài xế lái xe. Để có nhưng thành tích như trên thì xe tự lái phải nắm vùng những các nhiệm vụ về nhận thức (tức là hiểu được môi trường lái xe xung quanh), lập kế hoạch (có thể tạo ra đường đi đến đích một các tối ưu nhất có thể) và điều khiển (có thể điều khiển xe một các nhất quán kết hợp động lực học và các cảm biến). Ba nhiệm vụ tiên quyết này có thể được giải quyết cùng nhau hoặc giải quyết một cách riêng biệt trong môi trường. Các nhiệm vụ được giải quyết riêng biệt sẽ sinh ra các mô đun tối ưu riêng biệt, dẫn điến việc khi kết hợp các mô đun sẽ gây khó khăn không những thế ta chưa thể biết chắc rằng liệu các mô đun tối ưu thì bài toán có được giải quyết một cách tối ưu hay không.

Ngoài ra, khi kết hợp các mô đun riêng lẻ sẽ gây ra các hiện tượng như là nhiễu tín hiệu truyền. Hai yếu tố là việc kết hợp phức tạp giữa các mô đun riêng lẻ và tín hiệu truyền đi có thể bị nhiễu đã thúc đẩy quá trình đào tạo kết hợp cùng nhau. Quá trình end-to-end trong xe tự lái được nghiên cứu và phát triển, quá trình này không cần bất kỳ mô

đun trung gian nào để truyền tải có thể hạn chế nhiễu và trong quá trình tối ưu nhiệm vụ lái xe thì chính là quá trình tối ưu quá trình end-to-end.

Học bắt chước là một cách tiếp cận hàng đầu cho quá trình end-to-end của xe tự lái, do có sự đơn giản trong thiết kê và tính ổn định, mặc dù yêu cầu một lượng lớn thông tin để học các chính sách cạnh tranh. Học sâu tăng cường đang được thu hút vì những kết quả đáng khích lệ của nó trong lĩnh vực này, mà không yêu cầu chính xác thông thạo quỹ đạo: thay vào đó, chỉ cần một môi trường thực hoặc môi trường mô phỏng(như Carla). Hơn nữa học tăng cường có thể áp dụng các hành động tốt hơn.

Trong bài báo cáo này, chúng tôi sẽ nêu ra một số điều sau:

- Giới thiệu cơ bản vào ngắn gọn môi trường mô phỏng Carla, các cảm biến có trong Carla, cách thức và cơ chế hoạt động của Carla
- Kết hợp Học sâu tăng cường (cụ thể là Tối ưu hóa chính sách gần - PPO) với học theo chương trình và cách xe tự lái học end-to-end trong môi trường Carla
- Đánh giá khi cho xe chạy trên các môi trường khác nhau trong môi trường Carla, cụ thể sẽ huấn luyện xe trong một thị trấn và kiểm tra trong các thị trấn khác

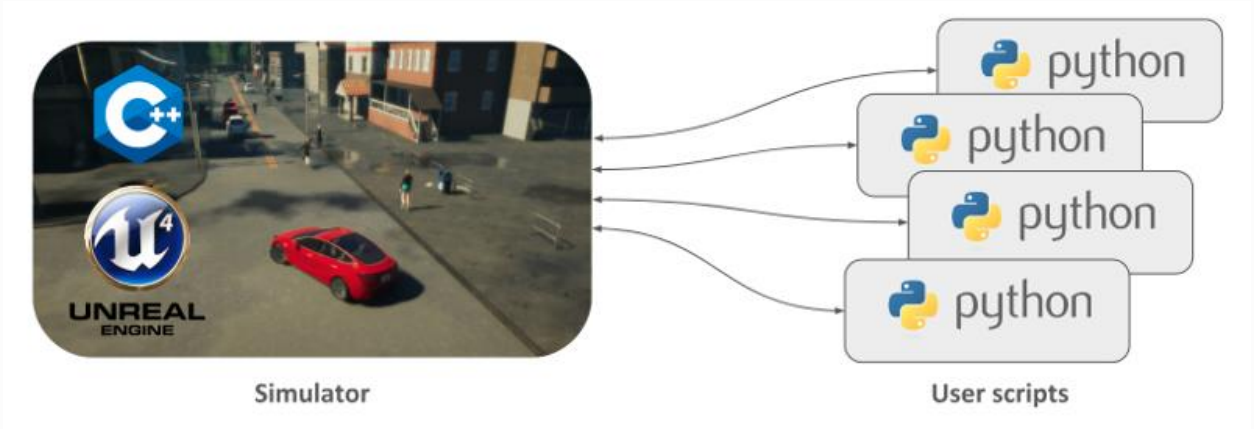
II. Môi trường Carla và các tác nhân bên trong môi trường

2.1. Môi trường Carla

Môi trường giả lập Carla là một trình giả lập thành phố thu nhỏ với đường xá và các tòa nhà gần giống với thực tế. Môi trường giả lập này được phát triển dùng để đào tạo các hệ thống xe tự hành không người lái áp dụng trí tuệ nhân tạo. Môi trường hỗ trợ các yếu tố ngẫu nhiên như người qua đường , xe khác chuyển làn đi ngược chiều, thời thiết, vật cản bất ngờ xuất hiện… làm tăng yếu tố trực quan của môi trường.CARLA dựa trên Unreal Engine để chạy mô phỏng và sử dụng tiêu chuẩn OpenDRIVE (1.4 như ngày nay) để xác định đường và cài đặt đô thị. Nền tảng mô phỏng hỗ trợ đặc điểm kỹ thuật linh hoạt của các bộ cảm biến, điều kiện môi trường, toàn quyền kiểm soát tất cả các tác nhân tĩnh và động, tạo bản đồ và hơn thế nữa…

Trình mô phỏng CARLA bao gồm một kiến trúc máy khách-máy chủ có thể mở rộng.
Máy chủ chịu trách nhiệm về mọi thứ liên quan đến bản thân mô phỏng: kết xuất cảm biến, tính toán vật lý, cập nhật trạng thái thế giới và các tác nhân của nó, v.v. Vì nó hướng đến kết quả thực tế, phù hợp nhất sẽ là chạy máy chủ với GPU chuyên dụng, đặc biệt là khi xử lý máy học.
Phía máy khách bao gồm tổng số các mô-đun máy khách điều khiển logic của các tác

nhân trong cảnh và thiết lập các điều kiện thế giới. Điều này đạt được bằng cách tận dụng API CARLA (bằng Python hoặc C ++), một lớp trung gian giữa máy chủ và máy khách không ngừng phát triển để cung cấp các chức năng mới.



Hình 1: Môi trường mô phỏng Carla (Nguồn: Web Carla)

Điều đó tóm tắt cấu trúc cơ bản của trình mô phỏng. Hiểu về CARLA còn hơn thế nữa, vì nhiều tính năng và yếu tố khác nhau cùng tồn tại trong nó. Một số trong số này được liệt kê dưới đây, để có quan điểm về khả năng của những gì CARLA có thể đạt được.

Các tính năng nổi bật:

- **Khả năng mở rộng thông qua kiến trúc** nhiều **máy chủ** : nhiều Client trong cùng một hoặc trong các Node khác nhau có thể kiểm soát các tác nhân khác nhau.
- **API linh hoạt** : CARLA có một API mạnh mẽ cho phép người dùng kiểm soát tất cả các khía cạnh liên quan đến mô phỏng, bao gồm tạo ra giao thông, hành vi của người đi bộ, thời tiết, cảm biến, v.v.
- **Bộ cảm biến lái xe tự động** : người dùng có thể định cấu hình bộ cảm biến đa dạng bao gồm LIDAR, nhiều camera, cảm biến độ sâu và GPS trong số những bộ cảm biến khác.
- **Mô phỏng nhanh để lập kế hoạch và kiểm soát** : chế độ này vô hiệu hóa kết xuất để cung cấp khả năng thực thi nhanh chóng mô phỏng giao thông và các hành vi của đường mà đồ họa không cần thiết.
- **Tạo bản đồ** : người dùng có thể dễ dàng tạo bản đồ của riêng mình theo tiêu chuẩn OpenDrive thông qua các công cụ như RoadRunner .

- **Mô phỏng các tình huống giao thông** : công cụ ScenarioRunner của chúng tôi cho phép người dùng xác định và thực thi các tình huống giao thông khác nhau dựa trên các hành vi mô-đun.
- **Tích hợp ROS** : CARLA được cung cấp tích hợp với ROS thông qua liên kết ROS
- **Đường cơ sở về Lái xe tự lái** : chúng tôi cung cấp các đường cơ sở về Lái xe tự lái như là các tác nhân có thể chạy được trong CARLA, bao gồm nhân tố AutoWare và nhân tố Học bắt chước có điều kiện .

2.2. Các cảm biến được dùng trên tác nhân

Tên	Câu lệnh và đầu ra	Tổng quát
Camera RGB	Câu lệnh: sensor.camera.rgb Đầu ra: carla.Image per step	Cung cấp tầm nhìn rõ ràng về môi trường xung quanh. Trông giống như một bức ảnh chụp hiện trường bình thường.
Collision sensor	Câu lệnh: sensor.other.collision Đầu ra: carla.CollisionEvent per collision	Truy xuất các va chạm giữa cha mẹ của nó và các tác nhân khác.
IMU sensor	Câu lệnh: sensor.other.imu Đầu ra: carla.IMUMeasurement per step	Bao gồm một gia tốc kế, một con quay hồi chuyển và một la bàn.

III. Học sâu tăng cường kết hợp học theo chương trình

Với sự ra đời của học sâu, nhiều kỹ thuật thị giác máy tính truyền thống đã được thay thế bằng mạng nơ-ron tích hợp sâu (CNN). Học end-to-end là một trong những mô hình học tự lái, trong đó người dùng cung cấp hình ảnh đầu vào từ camera phía trước tới mạng nơ-ron nhất định và mạng này xuất ra các tín hiệu điều khiển xe như ga, lái và phanh.

3.1. Môi trường học.

Môi trường học bao gồm $\mathcal{E} = \{S, O, A, P, r, \gamma\}$, chính thức là Quy trình quyết định Markov có thể quan sát được một phần (POMDP), xác định nhiệm vụ mà tác nhân phải

giải quyết. Môi trường được xây dựng bằng cách kết hợp trình mô phỏng lái xe CARLA (phiên bản 0.9.9) với thư viện phòng tập thể dục của OpenAI, trong đó:

+ **Không gian trạng thái :** S được CARLA định nghĩa ngầm, chứa thông tin chân thực về toàn thế giới. Xe tự lái không thể quan sát trạng thái của môi trường $s_t \in S$, do đó các trạng thái bị ẩn. Tại mỗi thời gian t, với trạng thái s_t thì tác nhân chỉ có thể quan sát được không gian o_t.

+ **Không gian quan sát O:** một không gian quan sát $o_t \in O$ là một chồng K = 4 gồm 4 khung hình được lấy từ K bước thời gian cuối cùng trong timesteps.
Cụ thể, $o_t = \{[I,G,V,N]_k\}^4_{k=1}$, trong đó: I là hình ảnh có kích thước 90x360x3 thu được bằng cách ghép (dọc theo chiều rộng) ba hình ảnh 90x120x3 từ cảm biến ảnh RGB lần lượt ở trái, giữa, phải của xe. G là vecto 9 chiều mã hóa các đối tượng địa lý ở đường gồm: giao lộ, ngã ba, có phải là đèn giao thông, tốc độ giới hạn, màu sắc đèn đường. V là vecto 4 chiều nhúng các tính năng của xe bao gồm: tốc độ, ga, phanh, di chuyển tương tự, và cuối cùng, N là vecto 5 chiều chứa các điểm điều hướng.

+ **Không gian hành động A:** được thực hiện bởi hai hành động liên tục, cụ thể là giá trị ga và phanh và góc lái. Mỗi hành động có giá trị trong khoảng [-1;+1].

+ **Động lực học chuyển tiếp $P(s_{t+1}|s_t, a_t)$:** xác định trạng thái của môi trường tại $s_t \in S$ phát triển theo thời gian do được áp dụng các dự đoán hành động $a_t \in A$. Động lực học chuyển tiếp được xác đinh bởi CARLA và không được tác nhân học một cách rõ ràng.

+ **Hàm đánh giá r:** phạt bất kỳ vụ va chạm nào, cũng như đi theo lô trình sai lộ trình. Chức năng phần thưởng của rất đơn giản, vì nó liên quan đến tốc độ, hướng, vi phạm và va chạm của xe theo cách trực quan:

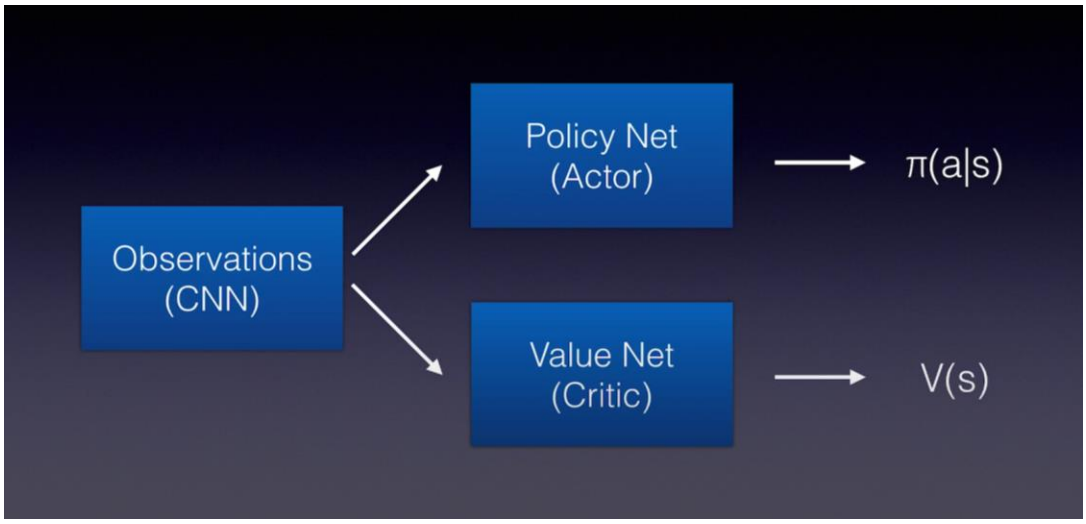
$$r_t = \begin{cases} -Cp & ,nếu\ va\ chạm \\ Slimit - Vspeed & ,nếu\ đi\ quá\ tốc\ độ \\ Vspeed * \frac{Vsim}{\left(\frac{dw}{2}\right)^2} & ,các\ trường\ hợp\ còn\ lại \end{cases}$$

Trong đó: Vspeed là tốc độ hiện tại của xe, Vsim là độ tương đồng cosin của xe với các điểm tham chiếu tiếp theo w, dw là khoảng cách giữa vị trí của xe là điểm tham chiếu, Slimit là tốc độ giới hạn đặt cho xe, cuối cùng là cp là mức phạt khi va chạm với đồ vật, con người hoặc phương tiện.

+ **Hệ số giảm thiểu** $\gamma \in [-1,1]$: Phần thưởng trong tương lai được giảm thiểu theo hệ số γ tại mỗi bước thời gian.

3.2. Thuật toán học tăng cường tối ưu hóa chính sách gần

PPO là thuật toán được phát triển từ thuật toán TRPO bởi OpenAI với 3 mục đích chính là dễ áp dụng, dễ cân chỉnh tham số và hiệu năng tốt. Thuật toán thuộc loại actor-critic vì vậy cũng có cấu trúc đặc trưng của loại này. Mô hình sẽ gồm 2 mạng neuron riêng biệt actor và critic nhưng có chung 1 đầu vào là trạng thái mà agent thu được, trạng thái có thể đã được tách đặc trưng bằng CNNs trước khi trở thành đầu vào cho mạng actor và critic. Mạng actor có số đầu ra tương ứng với số hành động của agent và tính toán ra xác suất của mỗi hành động $\pi(a|s)$ với đầu vào trạng thái, còn mạng critic sẽ đánh giá trạng thái đầu vào bằng cách ước lượng hàm giá trị trạng thái $V(s)$.



Hình 2: Cấu trúc mạng neuron thuật toán PPO

Mô hình có tới 2 mạng neuron riêng biệt, vì vậy chúng ta cần có hàm mất mát riêng cho mỗi mạng.

Hàm mất mát cho mạng actor:

$$L^{CLIP}(\theta) = \mathbf{E_t}[min\ (r_t(\theta)A_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t))]$$

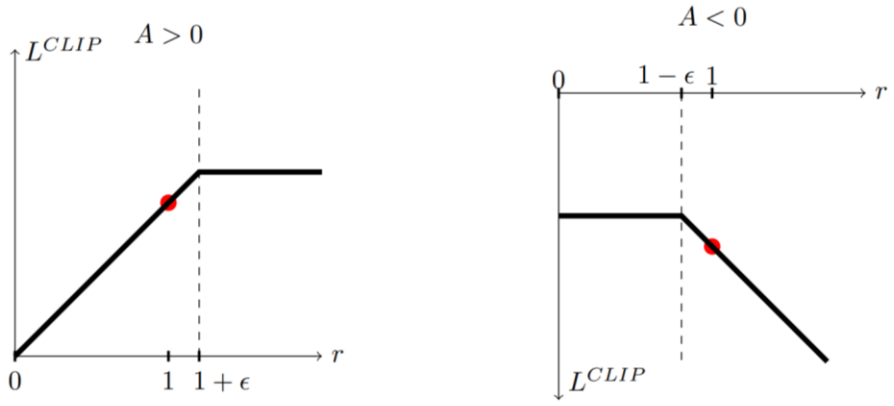
$$Với\ r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$$

Còn 1 thành phần nữa mà tôi chưa nhắc tới chính là A_t, kí hiệu cho hàm Advantage tại thời điểm t. Hàm Advantage gồm 2 phần là phần thưởng suy hao và ước tính giá trị trạng thái, nhằm đánh giá xem hành động đưa ra tốt hơn hay tệ hơn so với kì vọng. Hàm Advantage có thể được tính như sau

$$A_t = \delta_t + (\gamma\lambda)\delta_{t+1} + \cdots + \cdots + (\gamma\lambda)^{T-t+1}\delta_{T-1}$$

$$Với\ \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

Với việc giá trị hàm mất mát không được giới hạn có thể sẽ dẫn tới việc hệ thống mất ổn định. Để tránh điều này xảy ra chúng ta có thể sử dụng phương pháp "vùng tin cậy" như trong thuật toán TRPO, như đã đề cập ở trên PPO được phát triển tử TRPO tuy nhiên nhằm đơn giản hóa, thuật toán PPO sử dụng hàm clip với ϵ thường mặc định có giá trị bằng 0.2 để giới hạn giá trị của hàm mất mát, tăng tính ổn định cho hệ thống. Trong trường hợp hàm Advantage trả về giá trị dương, khi $r_t(\theta)$ trở nên lớn, nghĩa là hành động đưa ra có xác xuất lớn hơn nhiều so với trong quá khứ, giá trị hàm mất mát sẽ được trải phẳng để tránh cho mô hình phải cập nhật một giá trị quá lớn dẫn đến việc mất ổn định. Ngược lại, trong trường hợp hàm Advantage trả về giá trị âm, giá trị $r_t(\theta)$ gần với 0, giá trị hàm mất mát lần nữa được trải phẳng vì trong trường hợp này xác suất của hành động đưa ra nhỏ hơn nhiều so với trong quá khứ và tiếp tục cập nhật sẽ khiến xác suất của hành động giảm về 0.



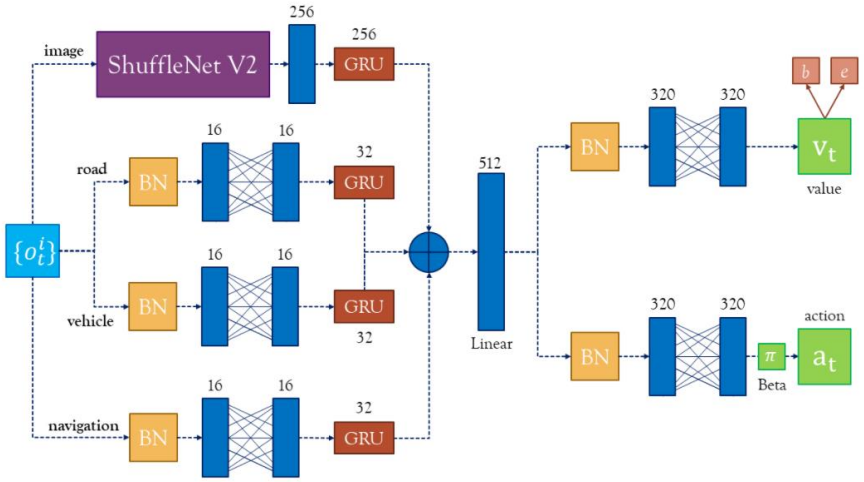
Hình 3: Hàm mất mát sau khi sử dụng hàm clip

Hàm mất mát cho mạng critic được tính tương đối đơn giản hơn:

$$L_{critic} = \sum_t (V_t - R_t)^2$$

Thuật toán PPO tương tự như các thuật toán actor-critic khác, rất phù hợp với những bài toán có không gian trạng thái liên tục. Đồng thời khắc phục được vấn đề nhạy cảm với hyperparameter của thuật toán actor-critic, có hiệu năng tốt và ổn định.

3.3. Cấu trúc của tác nhân.



Hình 4:Tác nhân sử dụng cấu trúc deep neuron network (Nguồn: Luca Anzalone -2021)

Tác nhân là một mạng nơ-ron học sâu lấy các quan sát o_t làm đầu vào và đầu ra là hành động tiếp theo a_t và giá trị kỳ vọng v_t với cả hai nhánh hành động và giá trị sử dụng một mạng nơ-ron chung, được ký hiệu là Pψ với các tham số ψ, xử lý các quan sát o thành một biểu diễn trung gian z. Vì mỗi o_t quan sát là một chồng gồm 4 khung hình, tức là $O_t = [o^1_t, \ldots, o^4_t]$, mạng Pψ được áp dụng tuần tự trên mỗi o^i_t, tạo ra bốn z^i_t được tổng hợp bởi Gated Recurrent Units (GRU) để thu được z_t. Hơn nữa, Pψ nhúng ShuffleNetV2 để xử lý dữ liệu hình ảnh. Cuối cùng, cả hành động và giá trị kỳ vọng đều là NN chuyển tiếp có hai lớp với 320 đơn vị được kích hoạt SiLU và Batch Normalization.

3.4. Học chương trình tăng cường

Việc vận hành một chiếc xe tự lái là một việc khó khăn, vì vậy chúng tôi đã cho tác nhân PPO, học tập theo cách của Học chương trình. Chia toàn bộ quá trình thành 5 giai đoạn học với mức độ khó của các chương trình sẽ tăng dần, sao cho Carla có thể học tập một hành vi phức tạp. Tất cả các chương trình học sẽ được đào tạo trên thị trấn 3 của CARLA: nơi có những cung đường phức tạp ở trên CARLA.

+ Giai đoạn 1: Học tập trên 10 vị trí khác nhau. Và không có mật độ giao thông cũng như người đi đường nào.

+ Giai đoạn 2: Học tập trên 50 vị trí khác nhau. Và có them 50 người đi đường ở những nơi ngẫu nhiên.

+ Giai đoạn 3: Thời tiết học sẽ thay đổi ví dụ như tác nhân hoạt động trong môi trường mưa hoặc ban đêm,…Và sẽ có 50 người và 50 phương tiện xuất hiện bất kỳ trong thị trấn.

Tất cả các giai đoạn đều chạy 500 tập huấn luyện trên 512 timesteps.

IV. Kết quả

4.1. Quy trình đánh giá

Chúng tôi thực hiện đánh giá xe tự lái trên 6 chỉ số, trên tất cả các thị trấn khác nhau và điều kiện thời tiết, cũng như mật độ giao thông khác nhau:

+ Các chỉ số: tỷ lệ va chạm, độ tương đồng (đo sự tương đồng của đường dự kiến và phương tiện đi), khoảng cách điểm tham chiếu, tốc độ, tổng phần thưởng và khoảng thời gian xe di chuyển (thời gian di chuyển trước khi bị va chạm vào vật thể phương tiện hay người đi đường).

+ Các thị trấn: Các bản đồ thì trấn ở trong Carla đều có sự khác biệt nhau riêng biệt. Chúng tôi đào tạo mô hình của mình trên một thị trấn 3 của Carla và đánh giá trên các thị trấn: Thị trấn 01, thị trấn 02, thị trấn 03, thị trấn 04, thị trấn 05.

+ Thời tiết: Thời tiết ở trong môi trường được đào tạo trong môi trường như nhiều sương, có mưa, vào ban đêm,… như những thời tiết thường gặp ở ngoài đời.

+ Giao thông: Chúng tôi có cài đặt hai loại môi trường giao thông là môi trường không có phương tiện tham gia là NoCrash và môi trường có mức độ giao thông vừa phải với 50 phương tiện và 50 người đi bộ.

4.2. Thảo luận

Chỉ số / Thị Trấn	Thị trấn 1	Thị trấn 2	Thị trấn 3	Thị trấn 4	Thị trấn 5
Tỷ lệ va chạm	0.86	0.78	0.88	0.51	0.49
Độ tương đồng	0.95	0.95	0.94	0.92	0.91
Tốc độ	7.78	8.46	8.13	9.05	8.55
Thời gian	296	335	347	413	406
Tổng phần thưởng	1866	2530	2157	2161	1764
Khoảng cách tham chiếu	1.55	1.44	1.75	3.75	3.74

Bảng 2:Chỉ số của tác nhân trong các môi trường khác nhau thu được trong điều kiện giao thông không có phương tiện và thời tiết ban ngày.

Từ bảng quan sát chúng tôi nhận định rằng thứ nhất xe còn chạy khá là chậm (khoảng 8 -9 km/h) và thứ 2 xe gặp khó trong việc nhận diện được vật cản. Điều này dẫn đến việc tốc độ xe chậm và hay gặp va chạm. Có thể là do thiếu khả năng khám phá mạng và những hành động chính sách khác. Nhưng sau khi thử nghiệm và áp dụng chương trình thì chúng tôi nhận định thời gian đào tạo của xe được rút ngắn hơn so với các phương pháp học tăng cường đơn giản.

V. Tài liệu trích dẫn

[1] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov. Proximal Policy Optimization Algorithms. 20, July, 2017.

[2] Luca Anzalone, Silvio Barra, Michele Nappi. Reinforced Curriculum Learning For Autonomous Driving In Carla. 2021 *IEEE International Conference on Image Processing (ICIP)*. 19-22, September, 2021.

[3] Luca96/carla-driving-rl-agent: Code for the paper "Reinforced Curriculum Learning for Autonomous Driving in CARLA" (ICIP 2021) (github.com)

[4] F. Codevilla, M. Muller, A. L ¨ opez, V. Koltun, and A. Dosovitskiy, “End-to-end driving via conditional imitation learning,” in 2018 IEEE International Conference on Robotics and Automation (ICRA), 2018

[5] Yoshua Bengio, Jer´ ome Louradour, Ronan Collobert, and Jason Weston, “Curriculum learning,” in Proceedings of the 26th annual international conference on machine learning, 2009