Aplicación de aprendizaje por refuerzo para el estacionamiento automático de un automóvil en un ambiente simulado

Marcelo José Inocente Cornejo
20160717@aloe.ulima.edu.pe
Universidad de Lima, Perú
doi: https://doi.org/10.26439/ciis2021.5634

En el presente trabajo se propone una solución de aprendizaje por refuerzo para efectuar el estacionamiento perpendicular automático en un vehículo de cuatro ruedas. Se centra en diseñar una función de recompensas que es usada para entrenar a los algoritmos *Proximal Policy Optimization* y *Soft Actor Critic*, además de la combinación de ambos en un ensamble. Finalmente, se logra obtener un éxito cercano al 99 % y una desviación final de aproximadamente un grado. Al entrenar los algoritmos con posiciones iniciales aleatorias se obtiene un desempeño pobre.

THE REINFORCEMENT LEARNING APPLICATION FOR THE AUTOMATIC PARKING OF A CAR IN A SIMULATED ENVIRONMENT

In the present work, a reinforcement learning solution is proposed to carry out automatic perpendicular parking in a 4-wheeled vehicle. It focuses on designing a reward function used to train the Proximal Policy Optimization and Soft Actor-Critic algorithms and combine both in an assembly. Training the algorithms with random starting positions gives poor performance. Finally, it is possible to obtain a success close to 99 % and an absolute deviation of approximately 1 degree.

Aplicación de aprendizaje por refuerzo para el estacionamiento automático de un automóvil en un ambiente simulado

Marcelo José Inocente Cornejo
20160717@aloe.ulima.edu.pe

Resumen

En el presente trabajo se propone una solución de aprendizaje por refuerzo para efectuar el estacionamiento perpendicular automático en un vehículo de cuatro ruedas. Se centra en diseñar una función de recompensas que es usada para entrenar a los algoritmos *Proximal Policy Optimization* y *Soft Actor Critic*, además de la combinación de ambos en un ensamble. Finalmente, se logra obtener un éxito cercano al 99 % y una desviación final de aproximadamente un grado. Al entrenar los algoritmos con posiciones iniciales aleatorias se obtiene un desempeño pobre.

Introducción

La tarea de estacionar un automóvil puede llegar a ser dificultosa, así como también provocar riesgos de accidentes asociados (Green, 2006). Este problema ya ha sido abordado por la industria automotriz. Diversos fabricantes han lanzado modelos al mercado con la funcionalidad de estacionamiento automático. Sin embargo, muchas quejas ponen de relieve el prolongado tiempo de detección de espacios o la incapacidad de estacionar al ser estos muy estrechos (Dyer, 2017). Por otro lado, estos mecanismos funcionan con base en métodos geométricos de cálculo de ruta (Ballinas *et al*., 2018; Hsu *et al*., 2008), mientras que muchos trabajos de la literatura actualmente proponen el uso de aprendizaje automático (Bojarski *et al*., 2016; P. Zhang *et al*., 2019; Zhuang *et al*., 2018), de forma que el algoritmo determine por sí mismo la mejor manera de manejar un auto.

A diferencia de otros métodos de aprendizaje como el supervisado, el aprendizaje por refuerzo no requiere de una data predefinida para realizar su entrenamiento. En lugar de ello, este tipo de algoritmos reciben como entrada un ambiente y un agente que actuará dentro de él, junto con un conjunto de "reglas de juego" que servirán como recompensas para indicar al agente qué comportamiento debe seguir. Una vez terminado el entrenamiento del algoritmo bajo estos parámetros, el agente será capaz de llevar a cabo la tarea objetivo.

La propuesta presentada en este trabajo hace uso del aprendizaje por refuerzo para entrenar a un modelo que efectúe la tarea de estacionamiento automático. Para esto se hace uso de los algoritmos *Proximal Policy Optimization* (PPO) y *Soft Actor Critic* (SAC), y su posterior combinación en un ensamble, a fin de comparar los desempeños de estos modelos en el estacionamiento perpendicular tanto desde una posición de inicio fija como desde posiciones iniciales aleatorias.

Materiales y métodos

La metodología seguida en este trabajo se detalla en la figura 1.

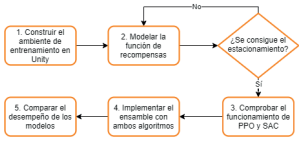

Figura 1. Diagrama de flujo de la experimentación
Elaboración propia

La etapa más crítica en este proceso es la modelación de la función de recompensas, pues consiste en definir cuáles serán las recompensas numéricas que se le darán al agente (por ejemplo, un valor de -1 cuando colisione y +1 cuando se estacione), las cuales impactarán en el posterior entrenamiento.

Materiales y métodos

Se construyó el ambiente de entrenamiento virtual haciendo uso del motor Unity mostrado en la figura 2. En este el agente, representado por un auto color verde, debía trasladarse hasta el espacio disponible entre los demás autos.

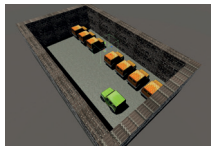

Figura 2. Escenario de estacionamiento construido usando el motor Unity
Elaboración propia

Para el entrenamiento de los modelos se hizo uso de la librería ML Agents, que permitía el uso de este escenario para entrenar implementaciones de los algoritmos a evaluar en este trabajo.

Resultados

Se entrenaron los algoritmos PPO y SAC hasta que fueron capaces de efectuar la maniobra de manera consistente. La evolución del ángulo y porcentaje de éxito a lo largo del entrenamiento puede verse en la figura 3. Posteriormente, fueron probados por 1000 episodios para poder extraer los resultados sobre su desempeño.

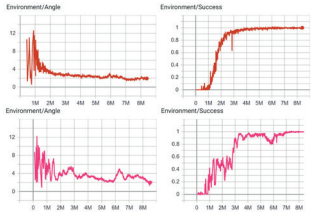

Figura 3. Evolución de la desviación (izquierda) y éxito (derecha) para PPO (arriba) y SAC (abajo)
Elaboración propia

Se logró obtener un éxito de 99 % en el escenario planteado utilizando el algoritmo PPO y uno del 99 con el algoritmo SAC. Por su parte, el ensamble logró un éxito de solo 96,8 %. En cuanto a los grados de desviación, PPO presenta un promedio de 1,86 grados, SAC tiene uno de 1,9 grados, y el ensamble registra uno de 1,71.

Para la lógica del ensamble se aplicó el cálculo del promedio de las acciones producidas por ambos algoritmos. Si las acciones diferían demasiado se asignaban pesos de 90 % para SAC y 10 % para PPO para evitar que se genere una "confusión" al recibir órdenes contradictorias.

Resultados

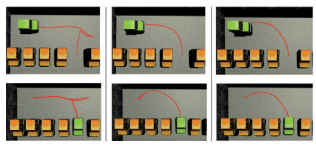

Figura 4. Trayectorias y posiciones finales del agente en el escenario perpendicular usando PPO (izquierda), SAC (centro) y ensamble (derecha)
Elaboración propia

Las trayectorias generadas por los algoritmos para estacionar el auto son mostradas en la figura 4. Se puede apreciar una clara diferencia entre las trayectorias de PPO y SAC, pues el primero se detiene para realizar una maniobra en reversa, mientras que SAC logra estacionar el auto de manera directa sin necesidad de detenerse. El ensamble exhibe una trayectoria similar a la de SAC debido a la preferencia configurada.

Al entrenar a los algoritmos en los escenarios, comenzando desde posiciones aleatorias, se obtuvieron resultados pobres, con porcentajes de éxito por debajo del 10 %.

Conclusiones

Se logró el objetivo de conseguir el estacionamiento automático perpendicular usando modelos de aprendizaje por refuerzo. La solución presentada obtiene resultados similares a los vistos en la literatura con respecto al éxito y grados de desviación, pues podemos notar que el porcentaje de éxito es alto, encontrándose cerca del 100 % que lograron Zhuang *et al*. (2018), mientras que los grados de desviación son mayores a los exhibidos por P. Zhang *et al*. (2019), quienes mostraron una desviación máxima de 0,747 grados en sus pruebas, aunque estos últimos ejecutaron su modelo con una posición inicial más favorable.

Pasando a hablar sobre la solución, la función de recompensas jugó un papel importante en este problema, pues fue modelada paso a paso para guiar el comportamiento de los modelos de forma que se logre el estacionamiento. En cuanto al comportamiento de los algoritmos, PPO mostró una mayor inclinación a la explotación de recompensas, mientras que SAC daba más prioridad a la exploración de su entorno.

La principal carencia de la solución presentada es la falta de adaptación a nuevas posiciones iniciales. Para contrarrestar esto se podría aplicar aprendizaje por currículo para un aumento gradual de la dificultad de las tareas o entrenar distintos modelos para las diferentes etapas como la detección, introducción al espacio y corrección de postura.

Referencias

Ballinas, E., Montiel, O., Castillo, O., Rubio, Y., y Aguilar, L. T. (2018). Automatic Parallel Parking Algorithm for a Car-Like Robot Using Fuzzy pd+i Control. *Engineering Letters*, *26*(4), 447-454.

Hsu, T. H., Liu, J. F., Yu, P. N., Lee, W. S., y Hsu, J. S. (2008). Development of an Automatic Parking System for Vehicle. *2008 IEEE Vehicle Power and Propulsion Conference, VPPC 2008*, 1-6.

Bojarski, M., Testa, D. del., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., y Zieba, K. (2016). *End to End Learning for Self-Driving Cars.* 1-9.

Dyer, E. (2017). *The Fallacy of the Self-Parking Car*. Popular Mechanics.

Faußer, S., y Schwenker, F. (2011). Ensemble Methods for Reinforcement Learning with Function Approximation. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 6713 LNCS*, 56-65.

Green, P. (2006). Parking Crashes and Parking Assistance System Design: Evidence from Crash Databases, the Literature, and Insurance Agent Interviews. *SAE Technical Papers*, 724.

Zhuang, Y., Gu, Q., Wang, B., Luo, J., Zhang, H., y Liu, W. (2018). Robust Auto-Parking: Reinforcement Learning based Real-time Planning Approach with Domain Template. *NIPS 2018 Workshop MLITS*.

Zhang, P., Xiong, L., Yu, Z., Fang, P., Yan, S., Yao, J., y Zhou, Y. (2019*). Reinforcement Learning-Based End-to-End Parking. Sensors.*