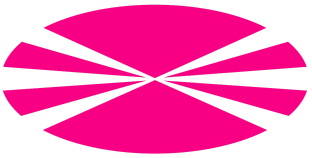

UNIVERSIDADE
DA CORUÑA

Robótica

Memoria práctica 3

Q-Learning

Óscar Alejandro Manteiga Seoane
Antonio Vila Leis

Q8 2022/2023

Introducción

En esta práctica se aborda el aprendizaje de seguir una línea coloreada de negro en un suelo blanco utilizando el algoritmo Q-Learning. El objetivo principal consiste en entrenar al robot Khepera IV para que recorra y siga la línea marcada en el suelo. Además, se incorpora un sencillo control de evitar obstáculos, el cual no forma parte del aprendizaje y que entrará en funcionamiento en casos donde el robot se encuentre a punto de chocar.

Implementación

Para implementar el comportamiento deseado al robot, dividimos el desarrollo en 2 etapas. La primera de ellas consistía en desarrollar los movimientos y el código base, mientras que la segunda consistía en crear la parte de aprendizaje. Cabe destacar que en principio pensamos en hacer una tercera en donde ampliaríamos los estados y complejidad de la práctica, pero por problemas que explicaremos más adelante, tuvimos que cancelarlo.

Etapa 1: movimiento sin aprendizaje

El desarrollo de esta etapa consistió, en primer lugar, en inicializar todos los sensores, actuadores y variables necesarias: sensores infrarrojos, motores y encoders de los mismos, variable gamma, matriz Q (tamaño 3x3 inicializada como matriz unidad) y matriz de visitas (tamaño 3x3) entre otras. Posteriormente, creamos los 3 estados posibles (S1, el robot sale por la izquierda; S2, el robot sale por la derecha; S3, resto de casos), con las 3 posibles acciones (A1, girar a la derecha; A2, girar a la izquierda; A3, seguir recto).

Una vez hecho esto, el siguiente paso fue crear el código necesario para evitar paredes (que lo obtuvimos del código en C que trae Khepera IV por defecto) y el que nos permitiera movernos por el escenario correctamente con la matriz puesta a mano. Para hacerlo, implementamos varias funciones: obtener la acción óptima de la matriz, realizar esa acción, obtener los valores de los sensores infrarrojos y obtener el estado actual del robot. Con todo esto realizado, las funciones de movimiento ajustadas y las variables auxiliares necesarias implementadas, el código funcionó acorde a lo esperado.

Etapa 2: movimiento con aprendizaje

Para incorporar aprendizaje al comportamiento del robot, cambiamos y añadimos ciertos elementos al código anterior. Añadimos las funciones de calcular el refuerzo y actualizar la matriz Q, modificamos la función de obtener la acción para añadir cierto grado de aleatoriedad (la iteración 0 será siempre aleatoria y la 1000 siempre óptima, mientras que los valores interiores se guiarán por la probabilidad).

Lo más importante en esta etapa es definir correctamente el cálculo de los valores para la matriz Q y el refuerzo que se devuelve. En nuestro caso, el refuerzo lo calculamos con la diferencia de la suma de los sensores que estaban encima del negro en el paso anterior y la suma de los sensores que se encuentran encima del negro ahora. A mayores, siempre que el robot tenga los 4 sensores en el negro, proporcionamos un 0.5 de refuerzo. Para el cálculo de la matriz Q, obtuvimos la fórmula de los apuntes dados:

$$Qn(s,a) \leftarrow (1 - \alpha n)Qn - 1(s,\ a) + \alpha n\{r + \gamma\, max(Qn - a(s',\ a'))\}$$

$$\alpha \leftarrow \frac{1}{1 + visitasn\ (s,a)}$$

Problemas en la implementación

A lo largo del desarrollo, tuvimos una serie de problemas que complicaron la implementación de la práctica y la posibilidad de realizar la mencionada etapa 3.

En un principio, no teníamos movimientos fijos, por lo que el robot con la matriz identidad funcionaba bien. Al cambiar los movimientos posteriormente no nos fijamos si lo seguía haciendo bien con dicha matriz, por lo que pasamos 2 semanas intentando incorporar aprendizaje a un robot que no podría nunca seguir la línea.

Finalmente, otro elemento que nos retrasó en gran medida el desarrollo, va en conjunto con la anterior. El error mencionado solo lo detectamos en tutoría, por lo que durante 2 semanas estuvimos cambiando el refuerzo, la detección de estados y los movimientos esperando que alguna combinación resolviese el problema.

Resultados

Con los problemas solucionados, ahora el robot sigue el comportamiento esperado. En un principio, sigue patrones aleatorios de los cuales aprender. Pasado la mitad de las iteraciones de aprendizaje empezará cada vez a hacer más elecciones óptimas y, llegado a las 1000 iteraciones, siempre optará por ellas. Algo que detectamos, es que en alguna de las ejecuciones, el robot no consigue aprender en lo referente al estado S3.

```
Iteración número:  1
Acción aleatoria
Estado S3 (resto de casos)
Matriz Q:
      A1     A2     A3
-----------------------
S1:   0.0   0.0   0.0
S2:   0.0   0.0   0.0
S3:   0.0   2.0   0.0
-----------------------
```

```
Iteración número:  500
Acción aleatoria
Estado S3 (resto de casos)
Matriz Q:
      A1     A2     A3
-----------------------
S1:  0.23 -0.38  -0.1
S2:  0.13  0.82  0.61
S3:  0.13  0.13   0.2
-----------------------
```

```
Iteración número:  1000
Realizando acción óptima
Estado S3 (resto de casos)
Matriz Q:
      A1     A2     A3
-----------------------
S1:   0.6 -0.16  0.12
S2:  0.07  0.94   0.5
S3:  0.21   0.2  0.43
-----------------------
```

Conclusión

En conclusión, el enfoque implementado utilizando el algoritmo Q-Learning permitió al robot Khepera IV aprender a seguir una línea coloreada de negro en un suelo blanco. A pesar de los desafíos encontrados durante el desarrollo, se logró obtener un comportamiento satisfactorio en el robot. Como posibles mejoras futuras, se podrían explorar estados más complejos (la mencionada etapa 3), probar diferentes algoritmos de aprendizaje por refuerzo para ampliar nuestro conocimiento y ampliar la complejidad general de la práctica.