Aprendizaje por Refuerzo en Robótica
Departamento de Ingeniería Informática
Instituto Tecnológico de Buenos Aires

Trabajo Práctico 4

Objetivo

Implementar comportamientos de evitamiento de obstáculos y seguimiento de paredes en un robot e-puck (simulado y real) mediante el algoritmo Q-Learning de aprendizaje por refuerzo.

1. Evitamiento de obstáculos

1. Crear un nuevo proyecto de Webots llamado arrtp6. Crear un ambiente con 3 ó 4 obstáculos en el subdirectorio worlds (se puede copiar un ambiente utilizado anteriormente) y crear un archivo para el controlador llamado arrtp6a.c en el subdirectorio controllers/arrtp6a.

2. Programar el controlador para que el e-puck evite obstáculos mediante el algoritmo de Q-Learning.

    1. Definir la representación de los estados (conjunto *S*) y las acciones (conjunto *A*) del agente y la función de refuerzo. El objetivo es hallar una función $Q: S \times A \rightarrow \mathbb{R}$ que maximice la recompensa total del agente con las acciones que puede tomar en cada estado.

    2. Implementar el algoritmo de Q-Learning según el siguiente pseudo-código:

> 1. Inicializar *Q(s, a)* arbitrariamente (por ejemplo, como matriz nula)
> 2. Repetir para cada episodio:
>   1. Leer el estado *s*
>   2. Repetir para cada paso del episodio, hasta que *s* sea terminal:
>   1. Elegir una acción mediante política ε-greedy teniendo en cuenta el estado actual de la función *Q*
>   2. Tomar la acción *a*
>   3. Leer el nuevo estado s' y obtener el refuerzo *r* a partir de evaluar la función de refuerzo
>   4. $Q(s, a) \leftarrow Q(s, a) + \alpha [r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$
>   5. Actualizar ε para que siga un decrecimiento lineal a través del tiempo
>   6. $s \leftarrow s'$

3. Correr una simulación para aplicar el algoritmo de Q-Learning. Una vez concluido el aprendizaje, evaluar el comportamiento del agente con la función *Q* aprendida.

4. Finalmente, subir el programa a un robot e-puck para evaluar los resultados en un ambiente real. Discutir los resultados en el informe de laboratorio.

2. Seguimiento de paredes

1. En el mismo proyecto, crear otro ambiente con varios obstáculos (incluyendo paredes que se quiebren a izquierda y derecha) en el subdirectorio worlds (se puede copiar un ambiente utilizado anteriormente) y crear un archivo para el controlador llamado arrtp6b.c en el subdirectorio controllers/arrtp6b.

2. Programar el controlador para que el e-puck siga paredes mediante el algoritmo de Q-Learning.

    1. Redefinir la representación de estados y acciones y la función de refuerzo, según corresponda.

    2. Reutilizar la implementación del algoritmo de Q-Learning, ajustando los parámetros que sean necesarios para este caso.

3. Correr una simulación para aplicar el algoritmo de Q-Learning. Una vez concluido el aprendizaje, evaluar el comportamiento del agente con la función *Q* aprendida.

4. Finalmente, subir el programa a un robot e-puck para evaluar los resultados en un ambiente real. Discutir los resultados en el informe de laboratorio.