1 Práctica 1. Sistemas Expertos

Descripción general

En esta práctica se pretende elaborar métodos para el control de un robot móvil basados en sistemas expertos. Se va a considerar un robot móvil que puede desplazarse por un plano horizontal mediante comandos de velocidad lineal y velocidad angular. El objetivo del robot será recorrer lo más precisamente posible un segmento en su entorno. El entorno está libre de obstáculos y no se tendrán en cuenta los posibles sensores del robot. Conoceremos únicamente su pose, es decir, la posición (x, y) del robot en el plano y la orientación del robot.
Se proporcionará un proyecto que realizará la simulación del movimiento del robot y la visualizará.

Python

Para el desarrollo de la práctica se utilizará Python como lenguaje de programación. Se proporcionará un proyecto Python con el código necesario para lanzar un entorno gráfico que permita visualizar la pose del robot y los datos de los objetivos que marcarán el camino a recorrer por el robot

Proyecto

El proyecto proporcionado a los alumnos contiene las siguientes 4 clases:

- **P1Launcher.py** Esta clase se encarga de crear el entorno gráfico y comunicar los distintos objetos implicados en la simulación. También contiene los métodos para pintar los elementos de la simulación en la pantalla.
- **Robot.py** Esta clase mantiene la representación del robot móvil. Internamente almacena la pose del robot y los comandos de velocidad lineal y angular que reciba. También se encarga de actualizar la pose de acuerdo a las velocidades y a su dinámica interna. Los alumnos no interactúan con objetos de esta clase directamente, aunque recibirán los datos de la pose del robot como entrada al sistema experto
- **Objetivo.py**. Esta clase representa un segmento recto entre dos puntos sobre el plano en el que se mueve el robot. Se utilizará como objetivo a recorrer por el robot. Con la información del segmento y de la pose del robot el sistema experto desarrollado por los alumnos deberá tomar decisiones de la velocidad lineal y angular que ha de tener el robot.
  Esta clase nos va a servir también para definir un triángulo en el que uno de los vértices será el punto inicio del movimiento, otro de los vértices el final del movimiento (como en el caso del segmento) y el tercer vértice define un área triangular por la que el robot no debe transitar.

- **expertSystem.py**. En esta clase se implementará el sistema experto que obtendrá los comandos de velocidad lineal y angular que debe ejecutar el robot en cada instante de tiempo para resolver el problema propuesto.

Las clases **P1Launcher.py, Robot.py y Objetivo.py (y cualquier clase adicional que se vaya añadiendo al proyecto) no podrán ser modificadas** por los alumnos. En caso de que sea necesario modificar alguno de estos ficheros para la corrección de errores o para añadir nuevas funcionalidades será el profesorado quien realice los cambios. En este caso se informará a los alumnos para que actualicen estos ficheros.
El alumno modificará únicamente los métodos incluidos en la **clase expertSystem**. Se podrán añadir a esta clase las propiedades y métodos adicionales que el alumno estime convenientes.

Aplicación

Tras ejecutar el proyecto proporcionado dará comienzo una aplicación con un interfaz de usuario (UI) como el que se muestra en la Figura 1. En la ventana simplemente se muestra el segmento que ha de recorrer el robot y la repersentación del robot como un círculo de color rojo. El segmento tiene marcado el inicio con un punto de color verde y el final con un punto de color rojo.

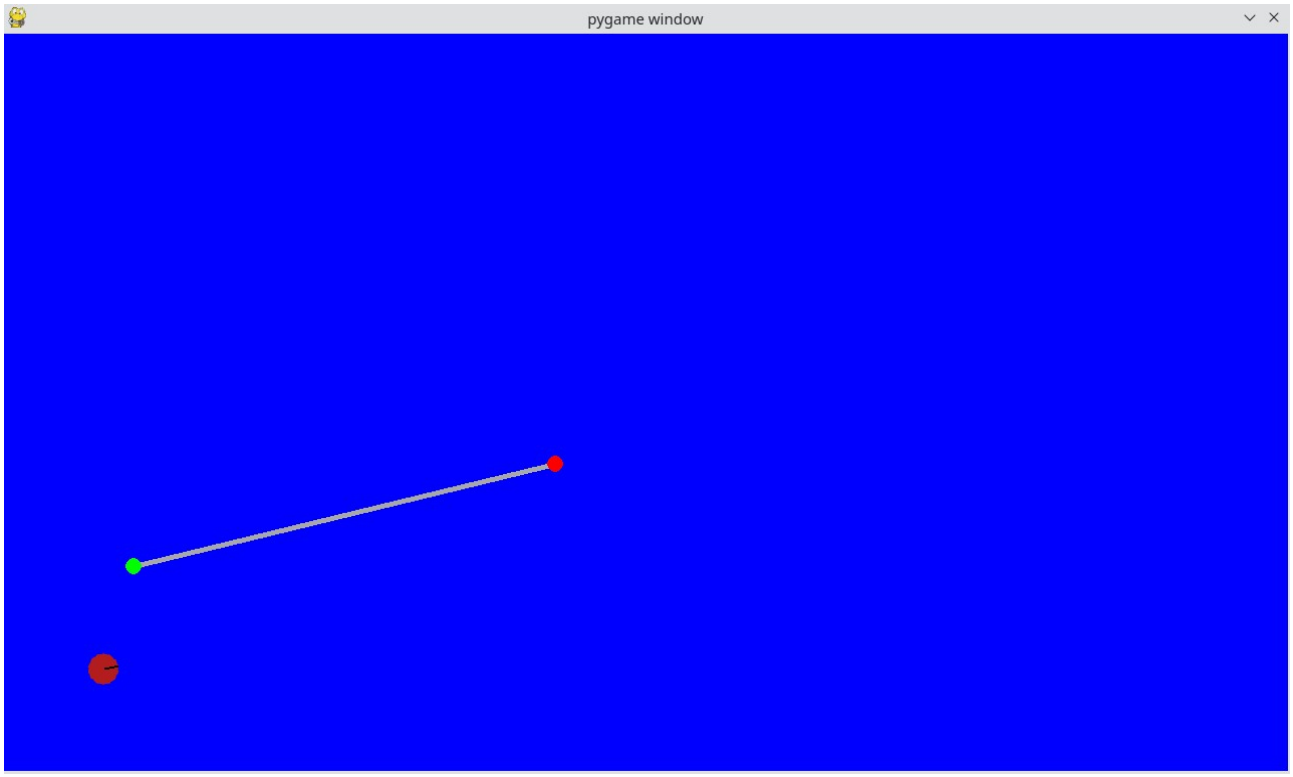

Figure 1: Ventana de la aplicación

Ejercicio 1. Recorrer un segmento

En este ejercicio se trata de implementar un sistema experto que gobierne el comportamiento de un robot móvil para trasladarse desde su extremo inicial al punto final del segmento. Se tendrán que crear las reglas de tipo if-then necesarias para que el robot se mueva desde la posición en la que se encuentre inicialmente hasta la posición final del segmento. Se valorará la cantidad de movimiento del robot que se realice por encima del segmento y el tiempo que emplee en recorrerlo.

La implementación del sistema experto se ha de realizar en el fichero proporcionado y de nombre expertSystem.py. Dentro de la clase `ExpertSystem`, el sistema experto estará implementado en la función `tomarDecision`. Esta función recibe la pose del robot y según el objetivo a recorrer debe devolver la velocidad angular y la velocidad lineal que el robot deberá tomar. El robot mantiene en la variable de clase `segmentoObjetivo` el objeto de la clase Objetivo al que debe dirigirse. También cuenta con una variable de clase llamada `objetivoAlcanzado` y con un método `esObjetivoAlcanzado()`. Al recibir un nuevo objetivo, la variable de clase `objetivoAlcanzado` pasa a valer `False`. Se tendrá que detectar cuándo se llega al punto final del objetivo y cambiar el valor a `True`. Se acepta considerar una tolerancia en la detección, es decir, no es necesario estar exactamente en la posición que marca el punto final del obstáculo.

El robot se mueve mediante un sistema dinámico diferencial utilizando dos ruedas motrices. Esto quiere decir que cuando la velocidad lineal del robot es cero, los giros se realizan alrededor del punto medio del robot. La velocidad lineal máxima que puede alcanzar el robot es de 3 m/s y la velocidad angular máxima es de 1 rad/s. Además presenta una aceleración y deceleración lineal de 1 m/s^2 y una aceleración y deceleración angular de 0.5 rad/s^2

El código debe estar preparado para que funcione independientemente de las posiciones del robot y de los puntos de inicio y final del segmento.

La aplicación puede que tenga una lista de objetivos y que vaya asignando nuevos objetivos al sistema experto según éste comunique que ha llegado al punto final del último objetivo asignado.

Además, se **debe** incluir una función que se llame `hayParteOptativa()` que devuelva *True* en caso de que la parte optativa haya sido implementada y *False* en caso contrario.

Ejercicio 1. Parte optativa.

En la parte optativa, el robot debe estar preparado para moverse de un punto inicial (A) a un punto final (B) sin entrar en el triángulo formado por un tercer punto (C). El objetivo por lo tanto será pasar el punto A al punto B sin entrar en el triángulo.

Ejercicio 1. Documentación

Es obligatorio documentar todo el trabajo realizado. Si alguna parte del trabajo realizado no se documenta se considerará como que no se ha hecho, lo que puede llevar a suspender la práctica. La documentación debe incluir una explicación detallada de todas las reglas implementadas y su justificación, así como de las pruebas que se hayan realizado para comprobar su correcto funcionamiento.

Ejercicio 1. Evaluación

El ejercicio 1 se puntuará de 0 a 10 puntos. Si la solución aportada consigue que el robot alcance todos los objetivos en cualquier situación, la nota no podrá ser menor de 5. Si la solución aportada es la que consigue la mayor puntuación de recorrido de entre todas las soluciones presentadas, su nota será de 10 puntos. Si la solución aportada es la que consigue la menor puntuación de recorrido de entre todas las soluciones presentadas, su nota será de 5 puntos. Cualquier solución con una puntuación de recorrido intermedia obtendrá una nota interpolada entre la mejor y la peor solución.

- ¡¡¡IMPORTANTE!!! Recordad que las prácticas son individuales y NO se pueden hacer en parejas o grupos. Cualquier código copiado supondrá un suspenso de la práctica para todas las personas implicadas en la copia. Se utilizarán herramientas para la detección de copia. Estos hechos serán comunicados a la Escuela Politécnica para que se tomen las medidas disciplinarias oportunas contra los infractores.