Universidad de Alcalá

Miniproyecto

Diseño de Controladores Borrosos y Neuroborrosos para un Robot Móvil

Sistemas de Control Inteligente

Grado en Ingeniería Computadores
Grado en Ingeniería Informática
Grado en Sistemas de Información
Universidad de Alcalá

1. Introducción

El objetivo de la práctica es el diseño del control de velocidad (lineal y angular) de un robot móvil para que éste recorra un circuito cerrado (delimitado por paredes), en el que pueden aparecer obstáculos estáticos, en el menor tiempo posible.

El circuito que se debe recorrer se muestra en la Figura 1. Es de tipo circular con un diámetro interno de 23 m con una distancia entre paredes de 3 m. Los obstáculos del entorno consisten en cilindros con un diámetro máximo de 1,5 m y pueden aparecer indistintamente en la pared interior o exterior del circuito. Se garantiza, no obstante, que la distancia entre dos obstáculos es siempre superior a 2 veces la distancia entre las paredes. A priori, se desconoce la posición de los obstáculos y su detección se deberá basar en las medidas utilizadas por los sensores a bordo del robot.

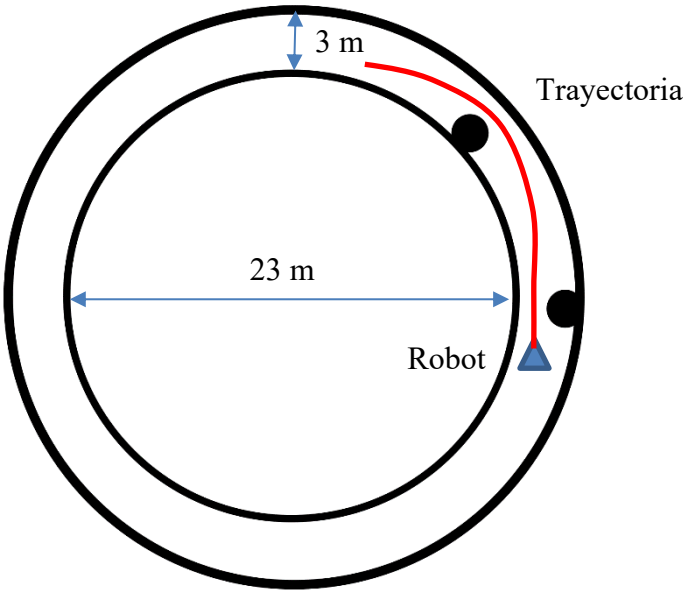

Figura 1. Problema de control propuesto.

2. Plataforma robótica

El robot móvil sobre el que se probarán los controladores tiene las siguientes características:

Longitud	33 cm	
Anchura	28 cm	
Altura	15 cm	
Velocidad lineal máx.	1 m/s	
Velocidad angular máx.	1 rad/s	

El robot dispone de sensores de posicionamiento absoluto con respecto al centro geométrico del círculo que forma el circuito y un anillo de 8 sensores de ultrasonidos dispuestos como se muestra en la

Figura 2. Los sensores de ultrasonidos tienen un rango de medida de distancia de [0,5] m.

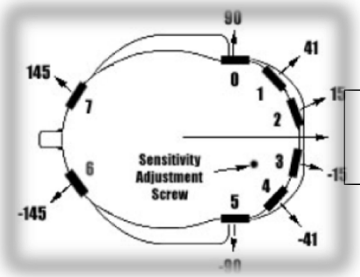

Figura 2. Distribución de sensores de ultrasonidos en el robot.

3. Entorno de simulación Matlab/ROS

El comportamiento del robot y su interacción con el entorno se simula en la plataforma ROS-STDR. Todos los detalles relativos a ROS y al simulador STDR, tanto su instalación como su configuración, desarrollo, ejecución, etc. se detallan en el documento "*Guía Rápida introducción ROS 1920*", disponible en la página web de la asignatura. Si no se tienen conocimientos de ROS, el estudio de dicho documento es indispensable para poder entender el contenido de esta práctica.

En la práctica se proporcionan los siguientes archivos para realizar la simulación:

- Máquina virtual con Linux y ROS.
- Archivo de configuración del mapa: `EntornoPracticaFinal.yaml`
- Archivo *launch* que lanza el STDR, el mapa del entorno y el robot: `PracticaFinal.launch`
- Imágenes utilizadas para crear el mapa:
  - `EntornoSinObstaculos.png`
  - `EntornoConObstaculos.png`
  - `EntornoConObstaculos2.png`
- Archivo de descripción del mapa: `EntornoPracticaFinal.yaml`
- Archivo de simulación del robot: `amigobot.xml`

Una vez guardados los archivos en los directorios correspondientes de la distribución de ROS-STDR, el simulador se inicia mediante el siguiente comando en la consola de Linux de la máquina virtual.

```
roslaunch stdr_launchers PracticaFinal.launch
```

La ventana de simulación del robot en el entorno sin obstáculos se muestra en la Figura 3.

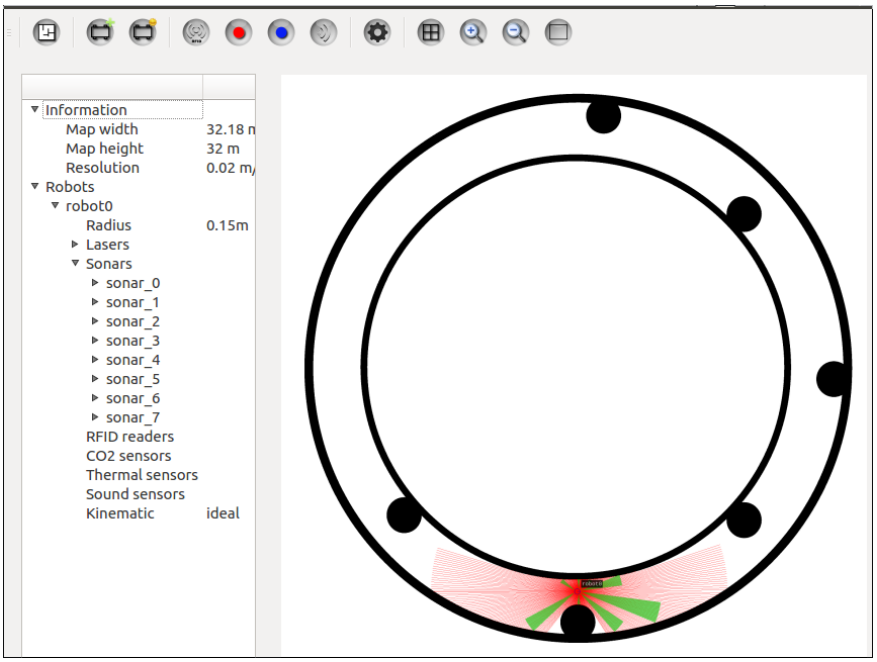

Figura 3. Entorno de simulación de la práctica.

La conexión con el robot se realiza a partir de las librerías de Matlab/ROS. Se incluye en la práctica el archivo *RobotROS.slx* que consiste en un bloque Simulink de conexión a ROS (véase Figura 4) con el que se probarán los controladores. Este bloque entrega en sus salidas la posición (salidas x e y), orientación (salida theta) y las lecturas de los 8 sensores de ultrasonidos del robot (salida sonar).

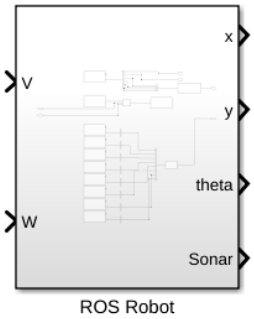

Figura 4. Bloque Simulink de acceso al robot y conexión a ROS.

Es necesario configurar correctamente la IP de la máquina virtual dentro del bloque del robot, haciendo doble click en cualquiera de los bloques de subscripción a ROS y haciendo click en “Configure network addresses” (véase Figura 5).

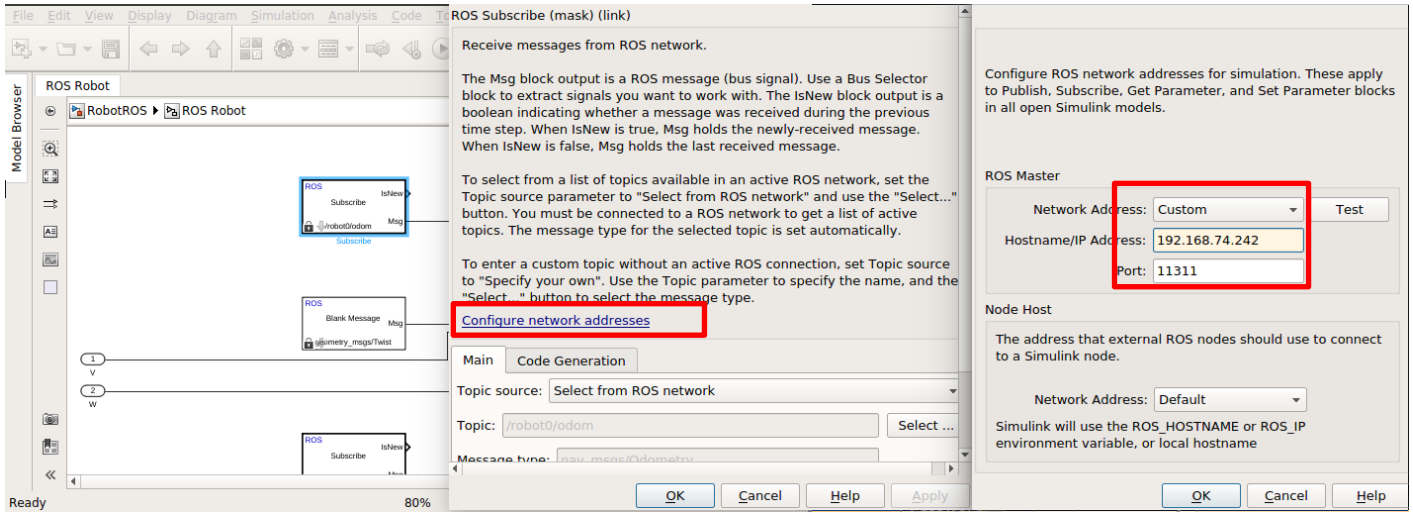

Figura 5. Configuración de la IP de la máquina virtual en el bloque ROS Robot.

4. Desarrollo del trabajo

El desarrollo de la práctica consta de dos partes que se describen a continuación.

Parte 1. Diseño manual de un control borroso de tipo MAMDANI.

En esta parte de la práctica se diseña un controlador de la velocidad angular y lineal del robot móvil en el entorno con obstáculos para recorrer el mismo en el menor tiempo posible a partir de la información obtenida de los sensores de ultrasonidos del robot y la posición y orientación con respecto al entorno. En esta parte, se dispone de libertad para elegir el número de entradas del controlador, los sensores utilizados, rangos de entrada y salida del controlador, funciones de pertenencia, y reglas del controlador, de manera que el robot sea capaz de cumplir con la tarea planteada.

Para la realización de esta práctica se hará uso de la herramienta **fuzzy** de Matlab utilizada en prácticas anteriores. Se recomienda al alumno el diseño de controladores que hagan uso del mínimo número de sensores posible para realizar la tarea encomendada. Una vez diseñado el controlador, se creará un bloque de control en Simulink donde se indique el archivo .fis que contiene el controlador y se conectará al bloque del robot. En la Figura 6 se muestra un ejemplo de controlador borroso que usa los sensores sonar_0, sonar_2, sonar_3 y sonar_5 para realizar el control del robot.

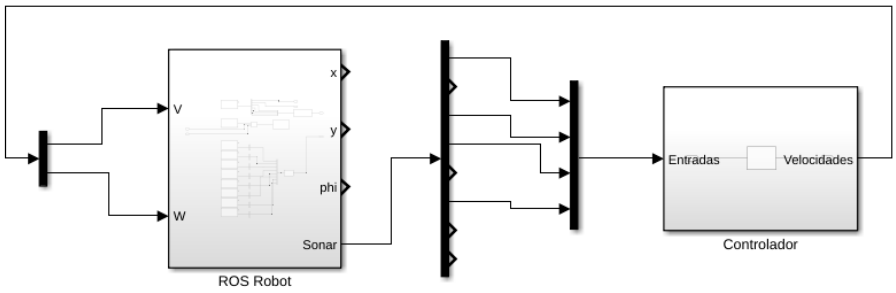

Figura 6. Ejemplo de control del robot

Parte 2. Diseño automático de un controlador neuroborroso de tipo SUGENO

Para la realización de esta parte se utilizará la herramienta **Anfisedit** y se dispone de una interfaz a través del teclado (script de Matlab `ControlManualRobot.m` incluido en los archivos de la práctica) con la que el alumno puede controlar manualmente el robot para que éste recorra el circuito evitando obstáculos y en el que se registrará la velocidad angular, lineal, posición, orientación y lecturas de los sensores. Para la ejecución del control manual se deberán configurar correctamente las direcciones IP de la máquina virtual (ROS_MASTER_IP) y la IP de la máquina (ROS_IP) en la primera línea de código de `ControlManualRobot.m`.

```
rosinit(['http://',ROS_MASTER_IP,':11311'],'NodeHost',ROS_IP)
```

Una vez iniciado el simulador STDR, se ejecuta el script `ControlManualRobot.m` desde la consola de Matlab. Para el control manual de las velocidades del robot se utilizan las teclas de dirección DERECHA e IZQUIERDA, para incrementar y decrementar respectivamente la velocidad angular del robot, con incrementos determinados por la variable incAngular=0.1 rad/s. Asímismo, se utilizan las teclas de dirección ARRIBA, ABAJO para incrementar y decrementar respectivamente la velocidad lineal del robot en un valor determinado por la variable incLineal = 0.1 m/s. Se deberá tener el foco de la ventana de la figura de Matlab que aparece al ejecutar el script activado para poder telecontrolar el robot. Mediante la pulsación de la barra espaciadora se dará por finalizado el control manual. Se grabará cada 0.1 segundos la siguiente información en las filas de la matriz `training`:

```
training(i,:)=[sonar_0, sonar_1, … , sonar_7, x, y , theta, vel_angular, vel_lineal]
```

donde,

- *sonar_0, …, sonar_7* son las medidas instantáneas de los sensores de ultrasonidos
- *x, y* es la posición instantánea del robot
- *theta* su orientación
- *vel_angular* y *vel_lineal* son los valores de velocidad angular y lineal del robot.
- El número de filas de `training` dependerá del tiempo empleado en el control manual del robot.

La variable `training` se utiliza para crear los controladores de velocidad lineal y angular del robot mediante la herramienta **Anfisedit** (véase Figura 7). En este caso se crearán controladores independientes para la velocidad lineal y angular del robot.
También se recomienda reducir el número de filas de la matriz para que contenga un máximo aproximado de 1500 filas.

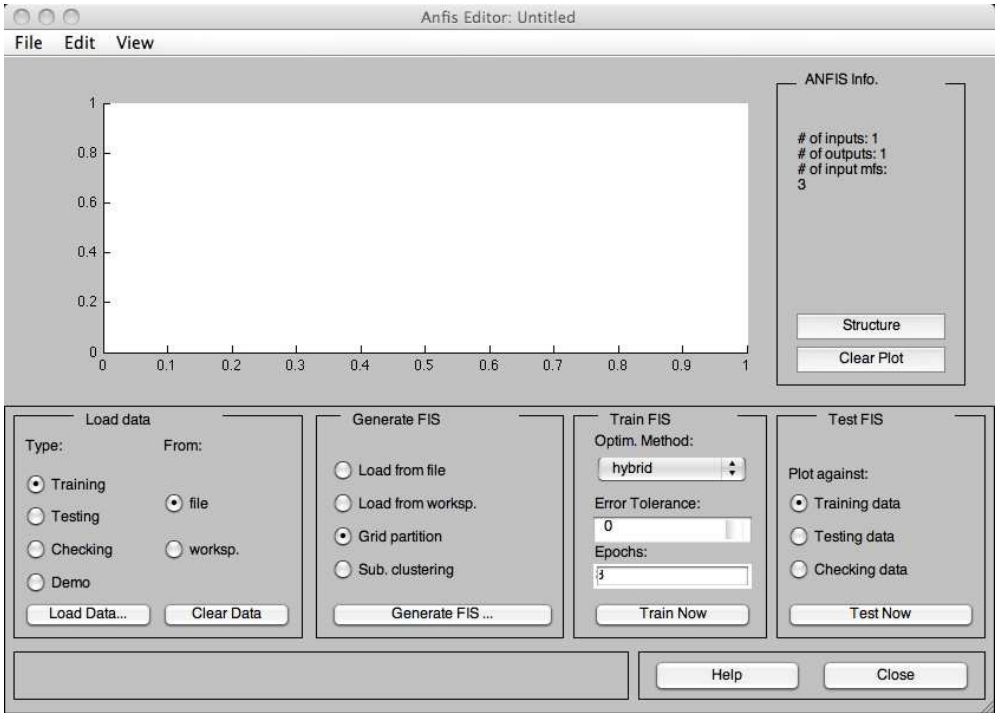

Figura 7. Herramienta Anfisedit

El proceso de generación de los controladores neuroborrosos pasa por las siguientes etapas:

1. Cargar los datos de entrenamiento/training, bien desde un archivo o bien desde el Workspace. Estos son los datos generados previamente mediante el script `ControlManualRobot.m` explicado anteriormente. Para diseñar cada uno de los controladores, será necesario crear una variable a partir de `training` con los datos que se necesiten. Por ejemplo, si se quiere diseñar el controlador que genere la velocidad angular a partir de los sensores de ultrasonidos 0 y 5, habrá que crear una variable nueva que tome las columnas 1 (sonar _0) y 6 (sonar_5) y 12 (vel_angular) de la variable `training`

```
train_angular = training(:,[1,6,12])
indices = round(linspace(1,size(traning,1),1500))
train_angular = train_angular(indices,:)
train_angular(isinf(train_angular)) = 5.0
train_angular = double(train_angular)
```

Para cargar estos datos, se pulsa el botón “Load Data” con la configuración mostrada en la Figura 8 y se escribe el nombre de la nueva variable creada en el cuadro que se abre a tal efecto (ANFISEDIT asume que la última columna de datos de entrada es la salida del sistema).

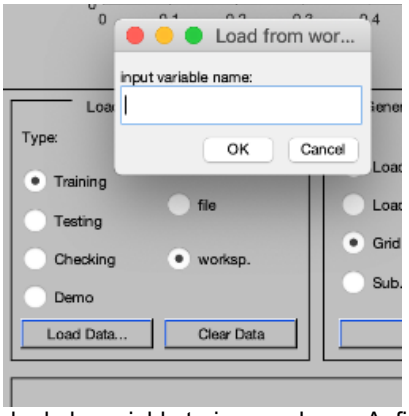

Figura 8. Entrada de la variable train_angular en Anfisedit.

2. Generar el FIS pulsando en el botón “Genarate FIS”, seleccionando alguno de los métodos disponibles (“Grid partition” o “Substractive Clustering”, por ejemplo). Al presionar este botón, aparece una ventana que permite configurar el número y tipo de funciones miembro de entrada y tipo de funciones miembro de salida, tal y como se muestra en la Figura 9.

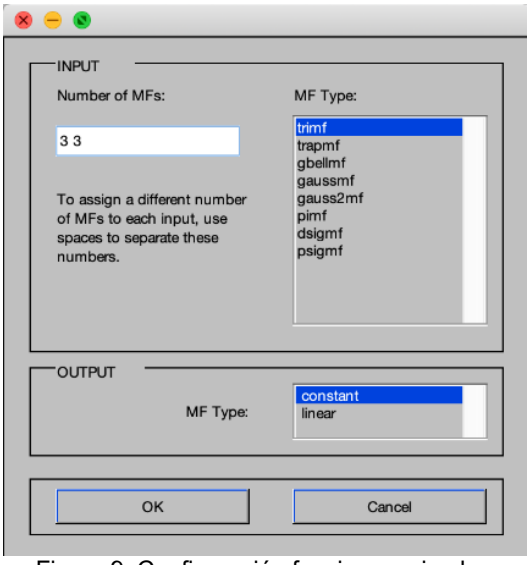

Figura 9. Configuración funciones miembro

3. Entrenar el sistema pulsando el botón “Train Now”, seleccionando el tipo de optimización tipo “hybrid” o “back_propagation” y configurando un número adecuado de “epochs”.
4. Comprobar el controlador generado contra los propios datos de entrenamiento pulsando el botón “Test Now”. El resultado será similar al mostrado en la Figura 10.

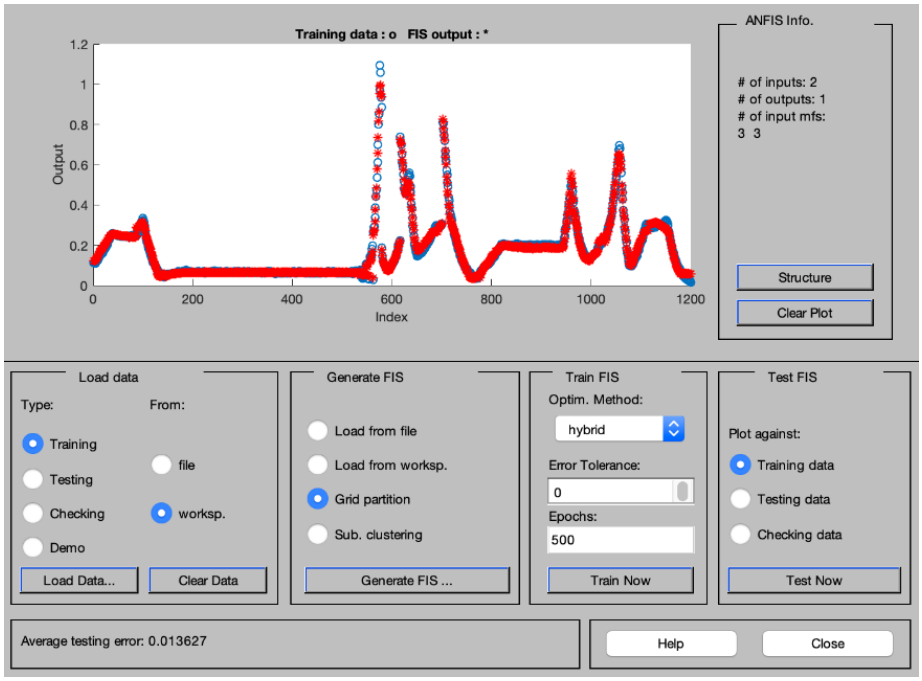

Figura 10. Resultado entrenamiento en el conjunto de entrenamiento

5. Una vez diseñado el controlador, es necesario guardarlo (File/Export/To File) con el nombre que se vaya a utilizar posteriormente en el script de Matlab (anfisV.fis o anfis.W en el ejemplo propuesto o cualquier otro que se quiera emplear).

Para probar y validar los controladores diseñados en este apartado, deberán seguirse los mismos pasos descritos para el controlador de la primera parte.

5. Evaluación de la práctica

Se indican a continuación los porcentajes máximos de la nota final para cada apartado de la práctica. Para obtener la máxima nota en cada apartado se evalúa la calidad objetiva de los diseños realizados, así como la presentación de los resultados.

- Desarrollo práctico de la práctica (75%)
  - Parte 1. (55%)
    - Diseño de un control borroso capaz de recorrer el entorno sin obstáculos (25%).
    - Diseño de un control borroso capaz de recorrer el entorno con obstáculos (30%)
  - Parte 2. (20%)
    - Diseño de un control neuroborroso para recorrer el entorno sin obstáculos (10%)
    - Diseño de un control neuroborroso para recorrer el entorno con obstáculos (10%)
- Presentación de la práctica final (25%).
  - Correcta exposición de los algoritmos utilizados y los resultados obtenidos.
- Competición (10% extra de la nota)
  - Cada grupo propondrá un controlador de los diseñados en la práctica final para recorrer el circuito con obstáculos en posiciones desconocidas “a priori” y se medirá el tiempo empleado por el controlador en recorrer el circuito. El ganador de esta competición tendrá un 10% extra en la nota.