Universidad de Alcalá

Práctica 1. Identificación y control neuronal (III)

Sistemas de Control Inteligente

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

Ejercicio 1. Identificación de un sistema utilizando una red recursiva de tipo NARX

Se dispone de un sistema dinámico de tiempo discreto (T_s = 0.1s) con una entrada y una salida y cuyas ecuaciones son desconocidas. Es posible registrar su comportamiento (salida) ante cualquier tipo de entrada mediante un sistema de prueba o "test-bench". Se desea entrenar una red neuronal de tipo NARX que emule con exactitud la respuesta del sistema. El sistema tipo caja negra se proporciona en el archivo modelo_identificacion.slx.

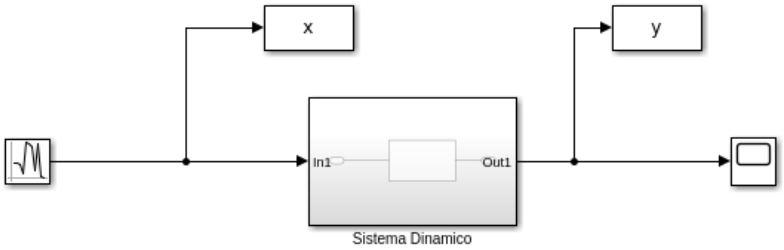


Figura 1. Sistema dinámico por identificar.

Se pide diseñar un archivo de simulación con nombre test_bench.slx (mostrado en la *Figura 1*) y en el que se simule el comportamiento del sistema ante una entrada aleatoria y donde se almacenen la variable de entrada al sistema "x" y la variable de salida "y" en estructuras de datos accesibles desde el workspace de Matlab ("*Structure with time*"). La simulación deberá tener una duración de 30s y deberá utilizar un "*solver*" de tiempo discreto con tiempo de muestreo T_s=0.1s.

La entrada aleatoria se generará con el bloque "*Sources/Random Number*" de la librería de Simulink. Configure dicho bloque con los parámetros mostrados en la y donde Ts=0.1 seg.

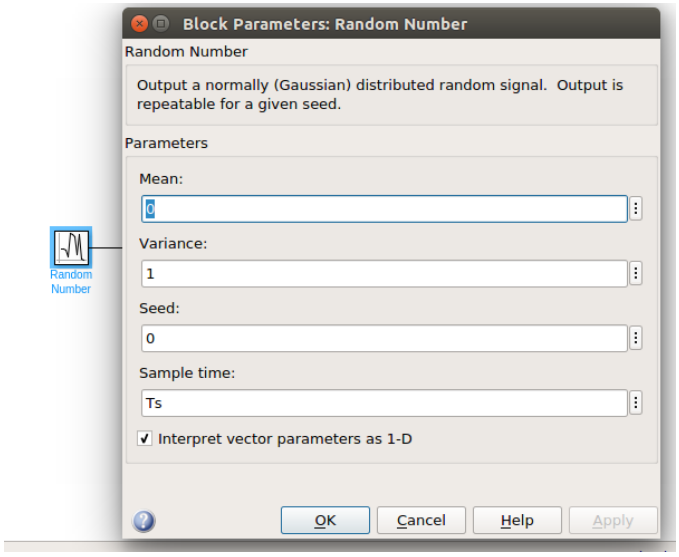


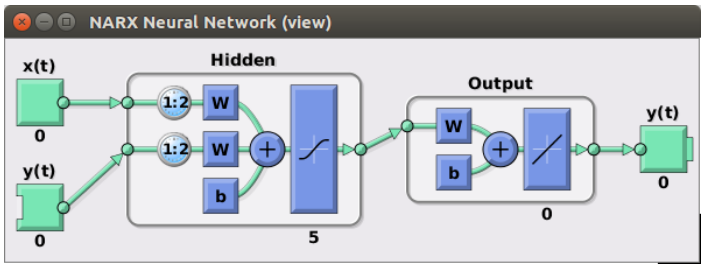
Figura 2. Configuración de una entrada aleatoria.

Mediante el siguiente código ejecute la simulación test-bench y genere los arrays de *inputs* y *outputs* con las entradas y salidas del sistema a lo largo de la simulación.

```
%% Generación de datos de simulación
Ts = 0.1;
sim('test-bench.slx')
inputs=x.signals.values';
outputs=y.signals.values';
```

Genere una red NARX con N=5 neuronas en la capa oculta y dos retardos en la entrada y en la salida realimentada. Utilice el siguiente código para la generación de la red y la visualización de su estructura.

```
%% Definición del modelo NARX
N=5;
net = narxnet(1:2,1:2,[N]);

view(net)
```



Prepare los arrays *inputs* y *outputs* para entrenar la red NARX. Para eso es necesario transformarlos en arrays de tipo cell que se guardan en las variables “inputsc” y “outputsc”.

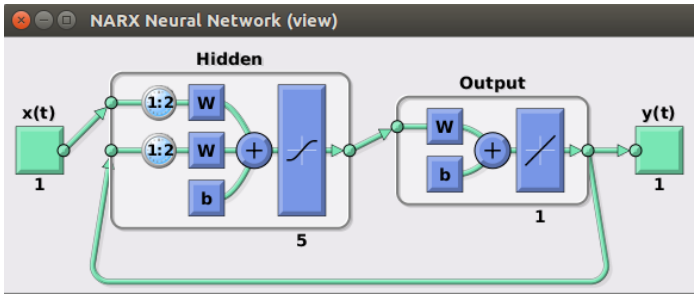
```
nT=size(inputs,2);

inputsc=mat2cell(inputs,1,ones(nT,1));

outputsc=mat2cell(outputs,1,ones(nT,1));
```

El entrenamiento de la red NARX se realiza asumiendo que la red es de tipo feedforward, lo que requiere transformar las entradas y salidas. Esto se realiza mediante la función de Matlab “preparets”.

```
[x,xi,ai,t] = preparets(net,inputsc,{},outputsc);
```

El entrenamiento de la red y su posterior conversión a una red recursiva se realiza mediante los siguientes comandos de Matlab.

```
net = train(net,x,t,xi,ai);
net = closeloop(net);
view(net)
```



Compruebe el error de entrenamiento de la red en los conjuntos de train, validation y test.

Por último, genere un modelo de Simulink de la red entrenada mediante el comando gensim y compare el desempeño de la red y el modelo, generando el s esquema de simulación test_bench2.xls mostrado en la Figura 3. Visualice el error entre la red neuronal y el sistema mediante el “scope” de dos canales.

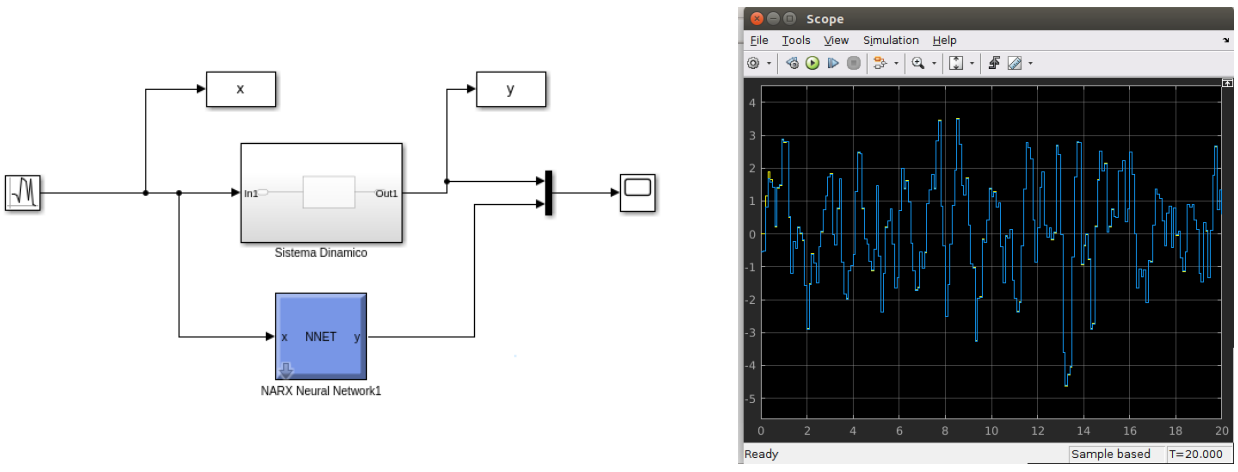


Figura 3. Comparación del funcionamiento de la red.

Ejercicio 2. Control para seguimiento de trayectorias mediante redes recurrentes.

1. Objetivo y descripción del sistema.

El objetivo de este ejercicio es diseñar un controlador que permita al robot, ya utilizado en sesiones anteriores de esta práctica, seguir una trayectoria preconfigurada. El esquema de control, mostrado en la Figura *4*, es idéntico al utilizado en la sesión II de la práctica1, sustituyendo las referencias de posición (bloques refx y refy) por un generador de trayectorias y sustituyendo el controlador por un nuevo controlador de tipo "caja negra" diseñado especialmente para el seguimiento de trayectorias ("controlblackboxTrajectory.slx").

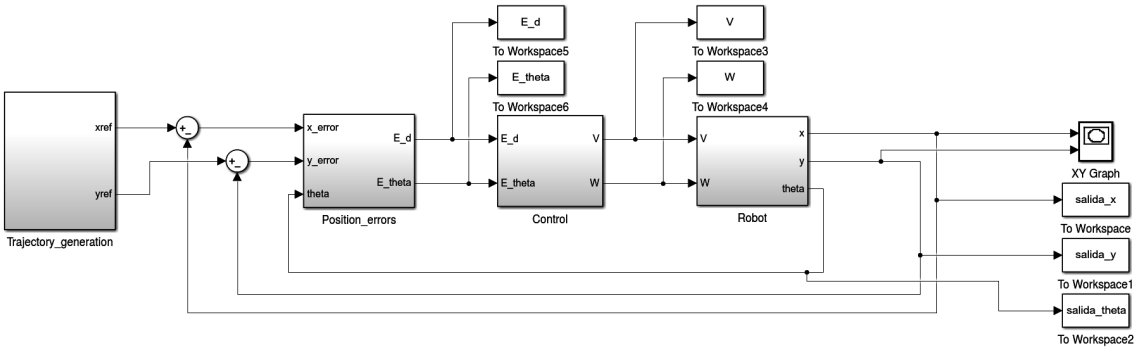


Figura 4 Diagrama de control para seguimiento de trayectorias.

La Figura 5 muestra la arquitectura del subsistema "Trajectory_generation", que generará la trayectoria que debe seguir el robot móvil. Este subsistema se proporciona dentro de la documentación de la práctica en el fichero "Trajectory_generator.slx". Para utilizar este bloque es necesario inicializar los parámetros iniciales de la trayectoria (x_0, y_0, th_0) y el periodo de muestro (Ts).

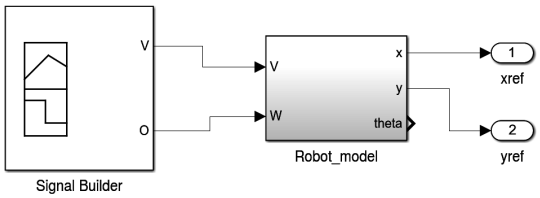


Figura 5. Subsistema generador de trayectorias.

El bloque "Signal Builder" se utiliza para generar las señales de velocidad lineal y angular. El bloque "Robot_model" es idéntico al utilizado para modelar el robot, añadiendo variables a los integradores para cambiar la posición (variables x0 e y0) y orientación (theta0) iniciales de la trayectoria. Este bloque y la trayectoria generada se muestran a continuación:

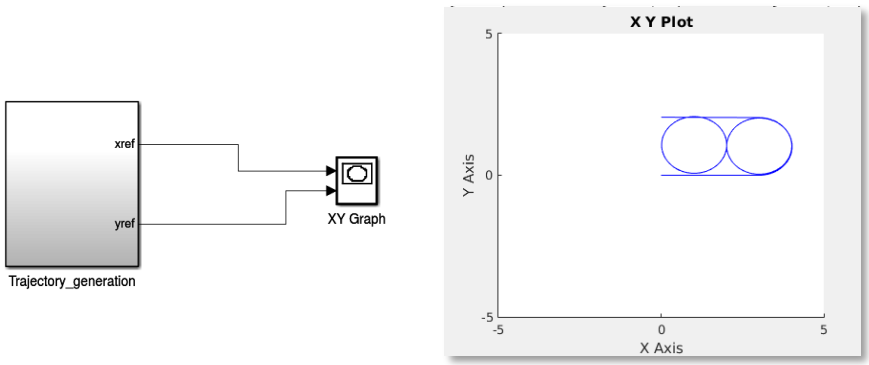


Figura 6. Trayectoria generada por el subsistema "Trajectory_generation".

2. Desarrollo del ejercicio.

Como se ha indicado anteriormente, el objetivo este ejercicio es diseñar un controlador que permita al robot seguir una trayectoria preconfigurada mediante el uso de redes recurrentes. Para ello se piden los siguientes apartados:

a) Implemente el esquema de la Figura 4. Utilice como controlador el proporcionado en el archivo "controlblackboxTrajectory.slx". Mantenga la configuración de los parámetros de la simulación (menú "Simulation/Model Simulation Parameters") de sesiones anteriores. Guarde el esquema de Simulink con un nombre reconocible.
b) Genere un script de Matlab, "RunTrajectoryControl.m" donde se muestre el seguimiento de la trayectoria seguida por el robot mediante el controlador proporcionado. Visualice mediante el comando plot de Matlab la trayectoria seguida por el robot y compárela con la trayectoria generada. Para ello, añada bloques de exportación al entorno de Matlab para las referencias de posición.
c) Siguendo los pasos descritos en el ejercicio 1, diseñe y entrene una red neuronal recursiva de tipo "narx" para emular el comportamiento del controlador proporcionado. El entrenamiento se realizará a partir de los datos generados en el apartado b). La red debe tener una capa oculta, dos retardos en la entrada y un retardo en la realimentación de la salida. Justifique la elección del número de neuronas de la capa oculta mediante experimentación.
d) Utilizando el comando gensim de Matlab, genere los bloques de Simulink correspondientes a las redes neuronales diseñadas en los apartados c) y d). Simule el comportamiento de dichos bloques para diferentes trayectorias generadas variando los valores de x0 e y0 y compare los resultados con el comportamiento del controlador original.