Miniproyecto: Diseño de controladoers borrosos y neuroborrosos para un robot móvil

Pablo Acereda García Laura Pérez Medeiro

Enero 2020

1 Introducción

El presente documento tiene la finalidad de dar una mayor explicación de las soluciones dadas para cada uno de los apartados propuestos en el enunciado de la práctica.

2 Diseño manual de un control borroso de tipo MAMDANI

En esta primera parte del proyecto el objetivo es completar un circuito (generado a partir de dos círculos concéntricos), con obstáculos en algunos puntos del mismo, y en el menor tiempo posible. Esto debe realizarse a partir de la información que se obtiene a partir de los sensores del robot (los cuales incluyen: sensores láser y sónares) que indican la distancia; así como de la orientación del mismo dentro del circuito.

Dentro de la carpeta 'Part1_Fuzzy' encontramos dos controladores borrosos (Cont_conObs.fis y Cont_sinObs.fis) con las reglas, funciones de pertenencia y rangos de las mismas necesarios para completar el circuito sin que el robot colisione contra ningún obstáculo o contra las paredes del mismo; así como el archivo de Simulink test_controller_generic.slx donde se encuentra el robot conectado al controlador que utilizará (el cual ha sido proporcionado por los profesores de la asignatura y del que solo se ha modificado el controlador a emplear y las conexiones entre demultiplexor y multiplexor para el uso de los sónares).

Cont_sinObs.fis es un controlador borroso pensado para recorrer un circuito de carácter circular sin la presencia de obstáculos Para ello usaremos la información de dos de los sensores, el sonar_0 y el sonar_2 (situados en la parte izquierda y la parte superior izquierda del robot). Las reglas que utilizaremos dentro del controlador serán reglas simples donde se tienen los valores CERC (cercano), MED (mediano) y LEJ (lejano) de los sonares, para las velocidades lineales se tendrán los valores LENT, MED, RAP (lento, medio y rápido) y para

las velocidades angulares los valores NEG, CERO y POS (negativo o disminuir, cero o mantener y positivo o aumentar).

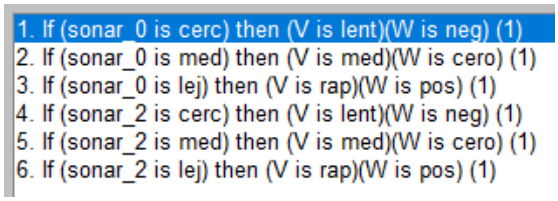


Figure 1: Reglas usadas para el controlador sin obstáculos

Para el controlador borroso Cont_conObs.fis necesitaremos información de un mayor número de sensores (sonar_0, sonar_2, sonar_3 y sonar_5 - siendo los dos nuevos incluidos los opuestos a los anteriores), así como una mayor cantidad de reglas, donde la mayoría de ellas son reglas simples pero también necesitaremos incorporar reglas compuestas - encargadas de resolver posibles conflictos o situaciones con el resto de reglas descritas anteriormente - así como asignarles un peso - para darle mayor importancia a las reglas que se deban ejecutar con mayor prioridad.

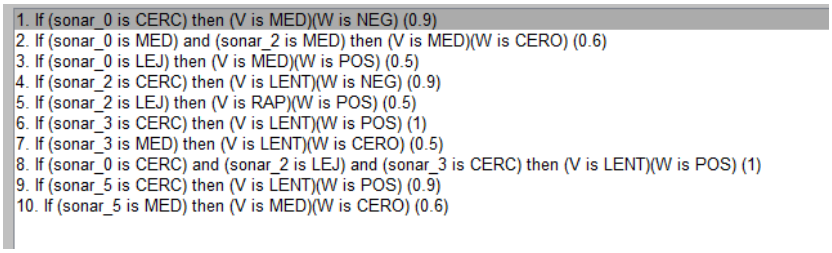


Figure 2: Reglas usadas para el controlador con obstáculos

Es decir, hay reglas que tendrán mayor importancia que otras como puede ser: Si el sonar_0 está cerca, el sonar_2 está lejos y el sonar_3 está cerca entonces la velocidad lineal deberá ser lenta y la velocidad angular deberá ser positiva. Es decir, si nos encontramos cerca de la pared del circuito pero también tenemos cerca un obstáculo deberemos reducir nuestra velocidad y tratar de girar hacia la cara interna del ciurcuito.

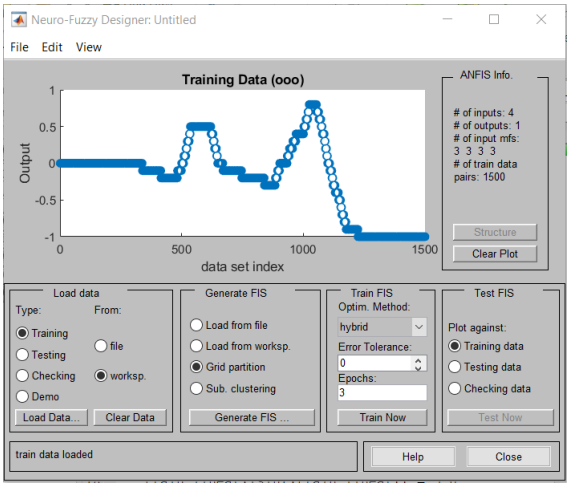
Se decidió darle pesos a las reglas porque se ha considerado que hay situaciones más críticas que otras. Se eligieron los sensores 0 y 5 para poder tener un mayor control sobre la proximidad de los límites del circuito y los sensores 3 y 5 para la detección de los obstáculos.

3 Diseño automático de un controlador neuroborroso de tipo SUGENIO

Para este apartado, abriremos el archivo 'ControlManualRobot.m' dado junto con otros archivos de la práctica y cambiaremos los valores de ROS_MASTER_IP por la IP de la máquina virtual o donde se vaya a ejecutar el nodo de ROS y ROS_IP por la IP de nuestra máquina o desde donde se vaya a ejecutar MATLAB. Después iniciaremos el simulador STDR y ejecutaremos el script de MATLAB, llamado 'ParseTrainingData.m', para recuperar los datos generados por el archivo de control manual y que se usarán usarán para el entrenamiento de la red neuronal.

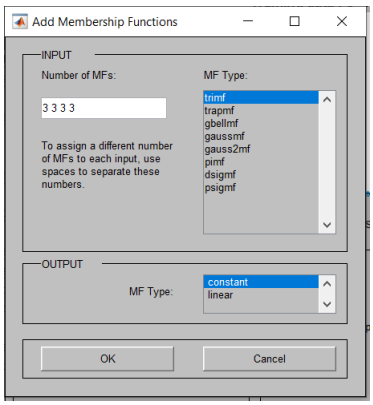
Para generar el controlador neuroborroso tendremos que seguir los siguientes pasos:

1. Cargar los datos de entrenamiento generados previamente con el script de control manual. Para ello usaremos el script llamado 'ParseTrainingData.m' desde donde cargamos la información del fichero guardado como 'datos_entrenamiento.mat' que contiene los datos del entrenamiento realizado, después indicamos los sensores que necesitaremos para controlar la velocidad angular y lineal (que serán el 0, 2, 3 y 5) además de limitar el número de líneas a utilizar a 1500. Tras la ejecución del script 'ParseTrainingData.m' tendremos las variables train_angular y train_linear cargadas dentro del workspace, que posteriormente serán llevadas al entorno de anfisedit.

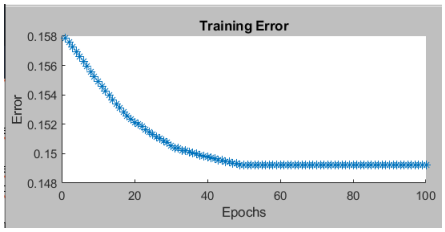


2. Para la generación del FIS seleccionamos el modo "Grid partition" (el cual genera funciones de membresía al particionar de manera uniforme los rangos de variables de entrada, creando un sistema difuso Sugeno de salida única y cuya base de reglas difusas contiene una regla para cada

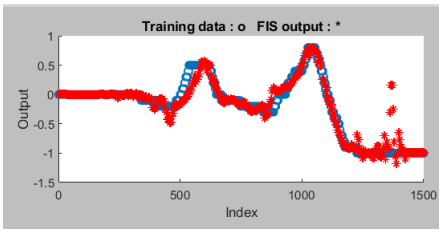
combinación de funciones de membresía de entrada). Se decide usar este método porque se observa que con él se obtienen mejores resultados aunque a suponga de necesitar una mayor cantidad de tiempo para generar el FIS.



3. Entrenar el sistema con el tipo de optimizacion “hybrid” y configurando el número de epochs (épocas) a 100, ya que así logramos buenos resultados haciendo que la solución sea más genérica - cuanto mayor es el número de epochs más específica estamos haciendo nuestra solución y por tanto menos susceptible es a los cambios (memoriza, no aprende).



4. Por último se comprueban los datos generados contra el propio sistema usando ”Test Now” y se guardan los controladores con los nombres anfisV.fis y anfisW.fis.



Para poder ejecutar el robot haciendo uso del controlador neuroborroso, encontramos el archivo ’test_controller_neuralfuzzy.slx’ dentro de la carpeta ’Part2_NeuralFuzzy’ donde encontramos el controlador_V que usa

anfisV.fis y el controlador_W que usa anfisW.fis. La decisión de separar el controlador inicial por dos se debe a que solo se pueden generar controladores neuro-borrosos individuales (uno para velocidad lineal y el otro para la angular).

4 Conclusión

Con los experimentos realizados se puede llegar a la conclusión de que el controlador neuro-borroso ofrece una solución mejor al controlador borroso desarrollado manualmente. Esto se puede deber a que el primero consta de datos provenientes de un experimento a partir del cual se obtiene el controlador; y el segundo obtiene sus valores a partir de la experimentación - además de tener reglas dependientes de la imaginación/ocurrencia de los creadores del controlador.