Práctica de lógica borrosa

Sandra Gómez Gálvez

Conducción cooperativa, conectada y autónoma
Universidad Politécnica de Madrid
2020/21

26 de diciembre de 2020

Índice general

Capítulo 1

Introducción

En esta práctica de lógica borrosa correspondiente al *Tema 2, Arquitecturas de control de vehículos autónomos*, de la asignatura *Conducción cooperativa, conectada y autónoma*, se implementará un controlador borroso para controlar la velocidad de un vehículo. A lo largo de este trabajo se detallará la implementación del sistema borroso en *MATLAB*, se mostrarán los resultados de la ejecución, y se analizarán los resultados.

Capítulo 2

Implementación del sistema borroso

Para implementar el sistema borroso se ha utilizado el software *MATLAB*, el cual proporciona un motor de inferencia de lógica borrosa.

Para comenzar, se ha abierto la aplicación *MATLAB* y se ha introducido el comando *fuzzy*, como se puede ver en la imagen 2.1, para abrir el motor de inferencia, también llamado *Fuzzy Logic Designer* (imagen 2.2).

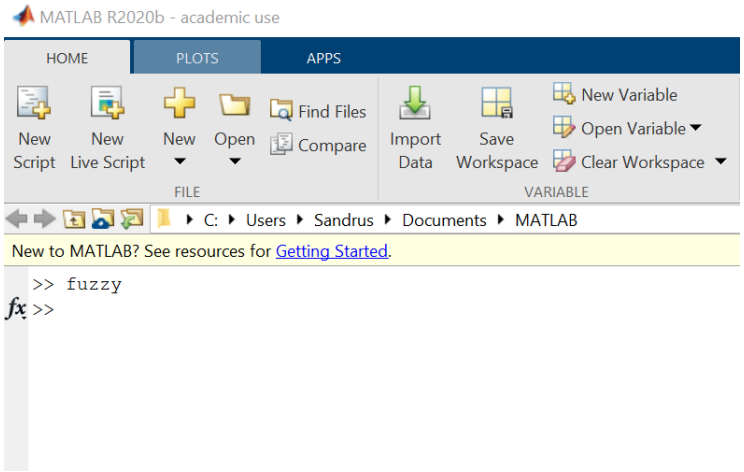

Figura 2.1: Comando *fuzzy*

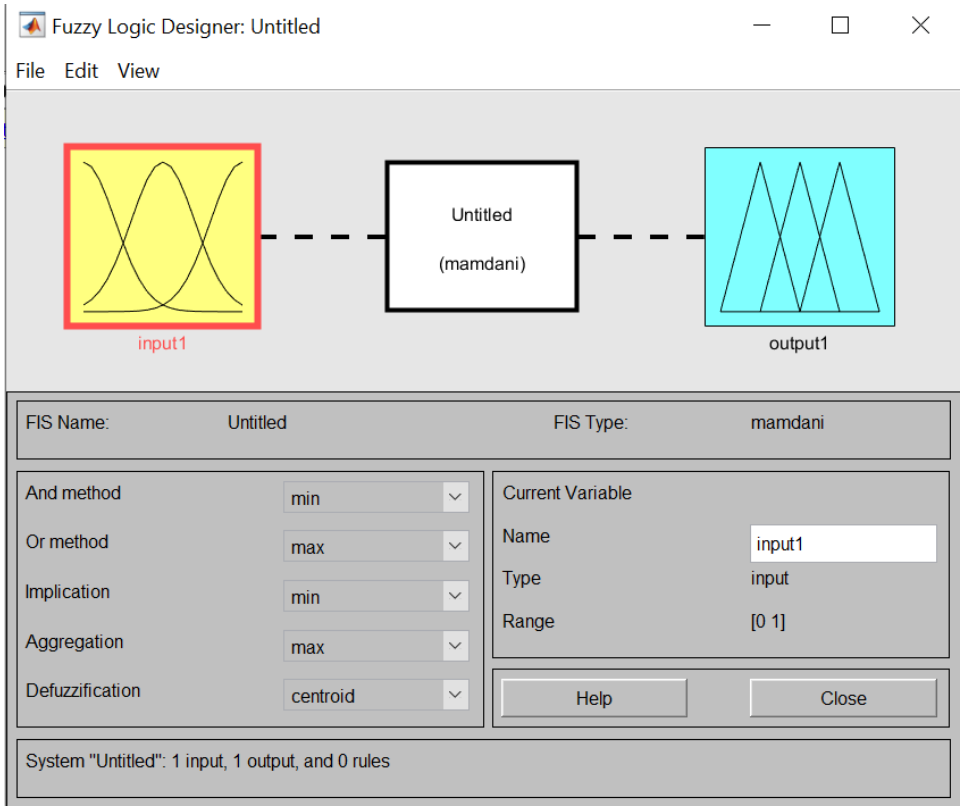

Figura 2.2: Fuzzy Logic Designer

Ya abierto el *Fuzzy Logic Designer* se ha seleccionado un sistema borroso de tipo Mamdani, como se puede ver en la imagen 2.3. En estos sistemas el consecuente de las reglas es un conjunto borroso representado por una función de pertenencia. En este caso, en el enunciado se exige que la función de pertenencia sea la trapezoidal.

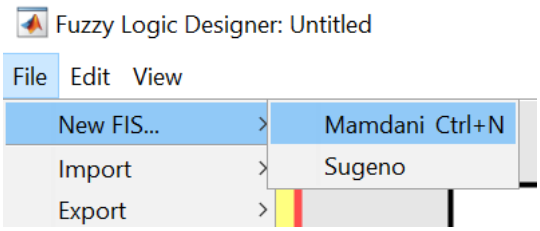

Figura 2.3: Selección del tipo de sistema borroso Mamdani

Después, se han introducido las variables, funciones de pertenencia y las reglas correspondientes.

Antes de explicar este proceso se muestra la definición del sistema:

- **Definición de los conjuntos borrosos de entrada:**
  - **Error_velocidad**. Funciones de pertenencia trapezoidales (trapmf):
    Error_vel { negativo -200 -200 -7 5 positivo -5 7 200 200 }
  - **Aceleración**. Funciones de pertenencia trapezoidales (trapmf):
    Aceleracion { negativa -100 -100 -50 50 positiva -50 50 100 100 }

- **Definición de los conjuntos borrosos de salida:**
  - **Acelerador.** Funciones de pertenencia trapezoidales (trapmf):
    Acelerador { levanta -1 -1 -0.5 0.5 pisa -0.5 0.5 1 1 }
- **Reglas:**
  - SI Error_vel positivo ENTONCES Acelerador levanta
  - SI Error_vel negativo ENTONCES Acelerador pisa
  - SI Aceleracion positiva ENTONCES Acelerador levanta
  - SI Aceleracion negativa ENTONCES Acelerador pisa

Una vez definido el sistema se detalla el proceso de introducción de variables, funciones de pertenencia y reglas.

Primero, se ha introducido dentro del *Fuzzy Logic Designer* los datos de los conjuntos borrosos de entrada. Para ello, se ha clicado en el cuadrado de *input1* (imagen 2.4).

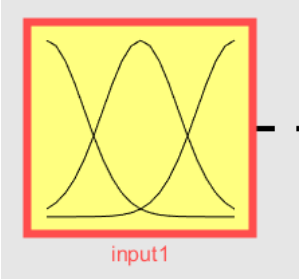

Figura 2.4: Acceso al input

Se ha comenzado introduciendo el conjunto borroso **Error_velocidad**, que está compuesto de dos funciones de pertenencia: la función *negativo* y la función *positivo*. Primero, se ha instanciado el rango de la entrada entre [-200,200] como se puede ver en 2.5. Después, se ha creado la función de pertenencia *negativo* y se han insertado sus parámetros como se muestra en 2.6. Seguidamente, de manera similar se ha hecho con la función de pertenencia *positivo* como se puede ver en 2.7. Finalmente, se pueden ver las dos funciones de pertenencia, mostradas en la figura 2.8.

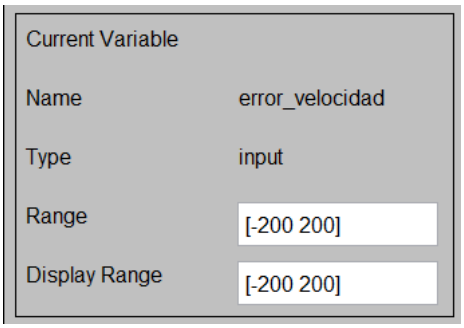

Figura 2.5: Rango instanciado para la entrada Error_velocidad

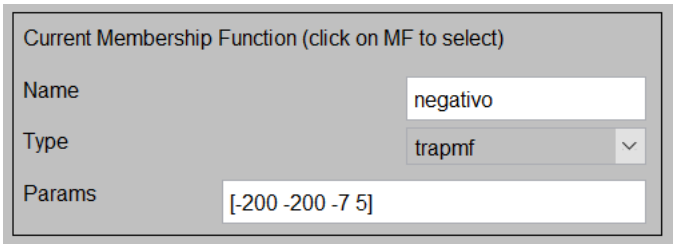

Figura 2.6: Parámetros función de pertenencia *negativo* - Error_velocidad

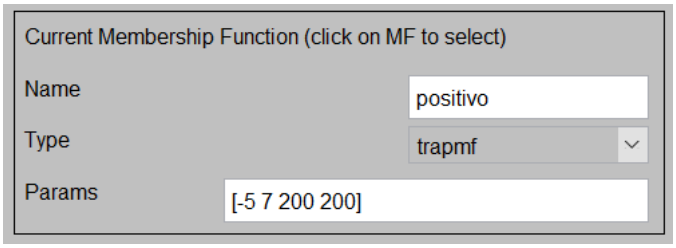

Figura 2.7: Parámetros función de pertenencia *positivo* - Error_velocidad

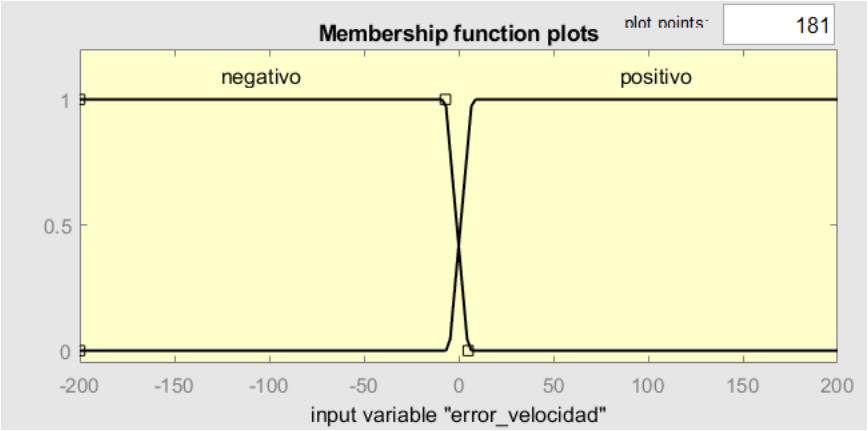

Figura 2.8: Conjunto borroso Error_velocidad - Funciones de pertenencia

De manera similar, se ha introducido el conjunto borroso de entrada **Aceleración**. Este conjunto también está compuesto de dos funciones de pertenencia *negativa* y *positiva*. Se ha instanciado el rango de entrada esta vez entre [-100,100] (imagen 2.9). Después se ha creado la función de pertenencia *negativa* y se han introducido sus valores como se puede ver en la imagen 2.10. Lo mismo se ha realizado con la función de pertenencia *positiva* como se puede visualizar en la imagen 2.11. Y, finalmente, ambas funciones de pertenencia se pueden ver en la gráfica 2.12.

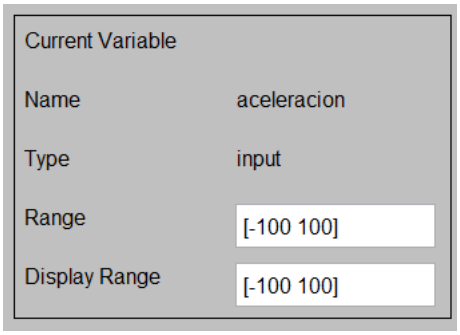

Figura 2.9: Rango instanciado para la entrada Aceleración

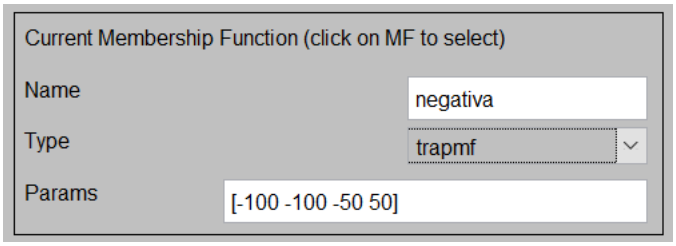

Figura 2.10: Parámetros función de pertenencia *negativo* - Aceleración

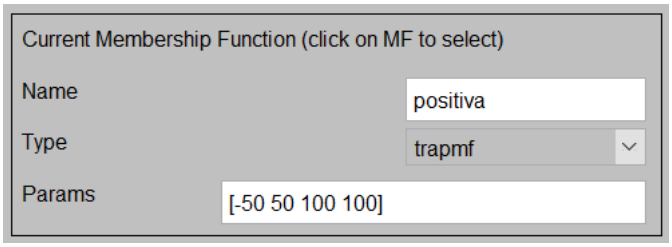

Figura 2.11: Parámetros función de pertenencia *positivo* - Aceleración

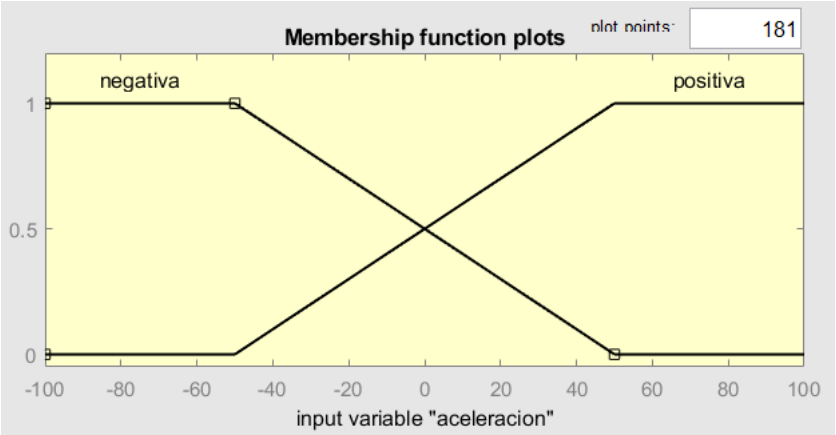

Figura 2.12: Conjunto borroso Aceleración - Funciones de pertenencia

Ahora, es el turno del conjunto borroso de salida **Acelerador**. Para introducir sus variables dentro del *Fuzzy Logic Designer* se ha clicado en el cuadrado de output (figura 2.13).

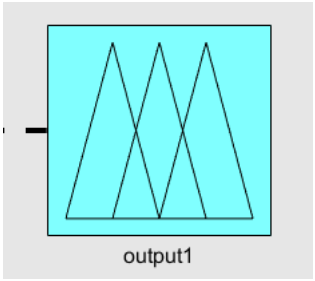

Figura 2.13: Acceso al output

El mismo proceso que con los conjuntos borrosos de entrada se ha seguido para crear el conjunto borroso de salida **Acelerador**. Este conjunto está compuesto por dos funciones de pertenencia: la función *levanta* y la función *pisa*. Primero, se ha instanciado el rango del conjunto borroso de salida entre [-1,1] (figura 2.14). Después se ha creado la función de pertenencia *levanta* como se puede ver en la figura 2.15, y, del mismo modo, se ha creado la función de pertenencia *pisa*, figura 2.16. Finalmente, ambas funciones de pertenencia se pueden ver en una gráfica, figura 2.17.

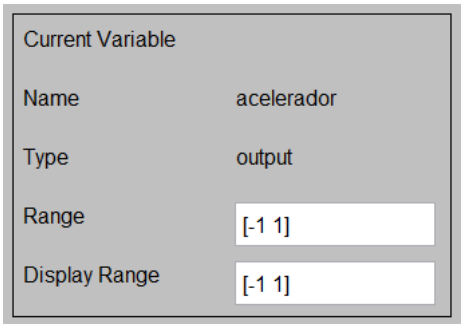

Figura 2.14: Rango instanciado para la salida Acelerador

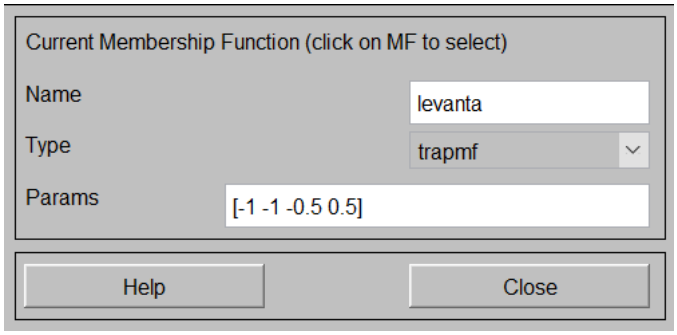

Figura 2.15: Parámetros función de pertenencia *levanta* - Acelerador

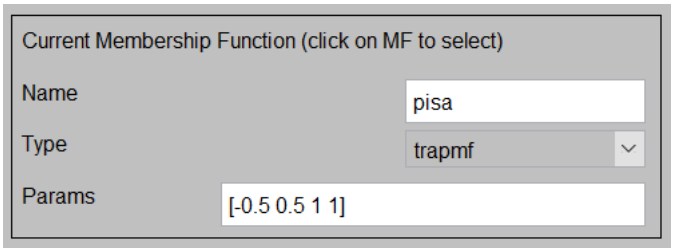

Figura 2.16: Parámetros función de pertenencia *pisa* - Acelerador

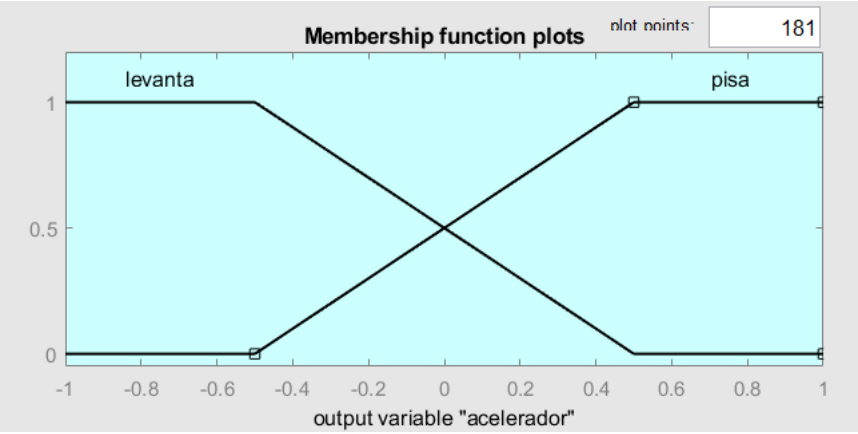

Figura 2.17: Conjunto borroso Acelerador - Funciones de pertenencia

Ahora, es el turno de las reglas. Para ello, dentro del *Fuzzy Logic Designer* se ha clicado en el cuadrado central (figura 2.18).

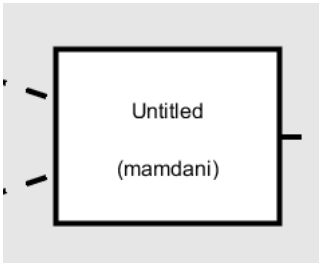

Figura 2.18: Acceso al definidor de reglas

Se han elegido las opciones determinadas para cada regla mediante el panel de definición de regla (figura 2.19).

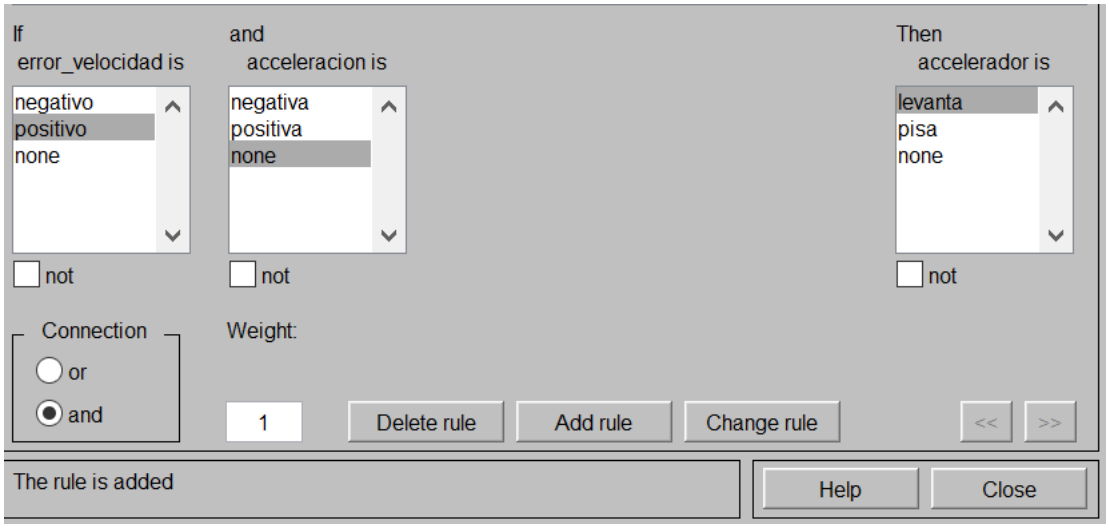

Figura 2.19: Panel de definición de regla

Así las cuatro reglas se han instanciado. Estas se pueden ver en la figura 2.20.

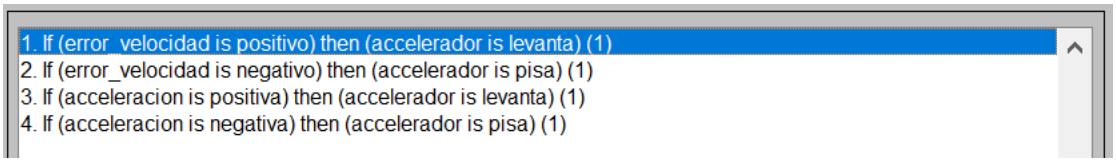

Figura 2.20: Reglas instanciadas

En penúltimo lugar, se han configurado los métodos, implicadores, agregadores y la *defuzzificación*, como se puede ver en la figura 2.21.

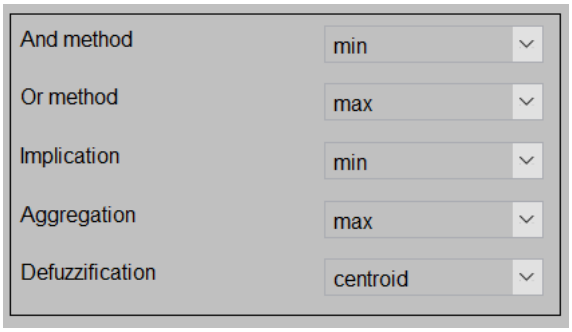

Figura 2.21: Configuraciones finales

Para finalizar el proceso de implementación del sistema borroso, simplemente se ha guardado en el disco, en un fichero .fis. Para hacer esto, se han seguido los pasos que se pueden ver en las figuras 2.22 y 2.23.

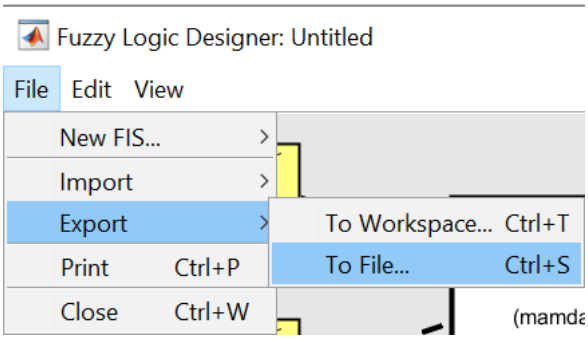

Figura 2.22: Guardado en disco - Paso 1

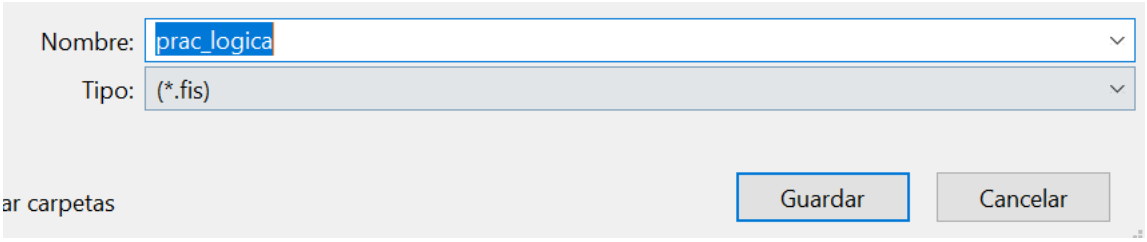

Figura 2.23: Guardado en disco - Paso 2

El sistema borroso finalmente se muestra como se puede ver en la figura 2.24.

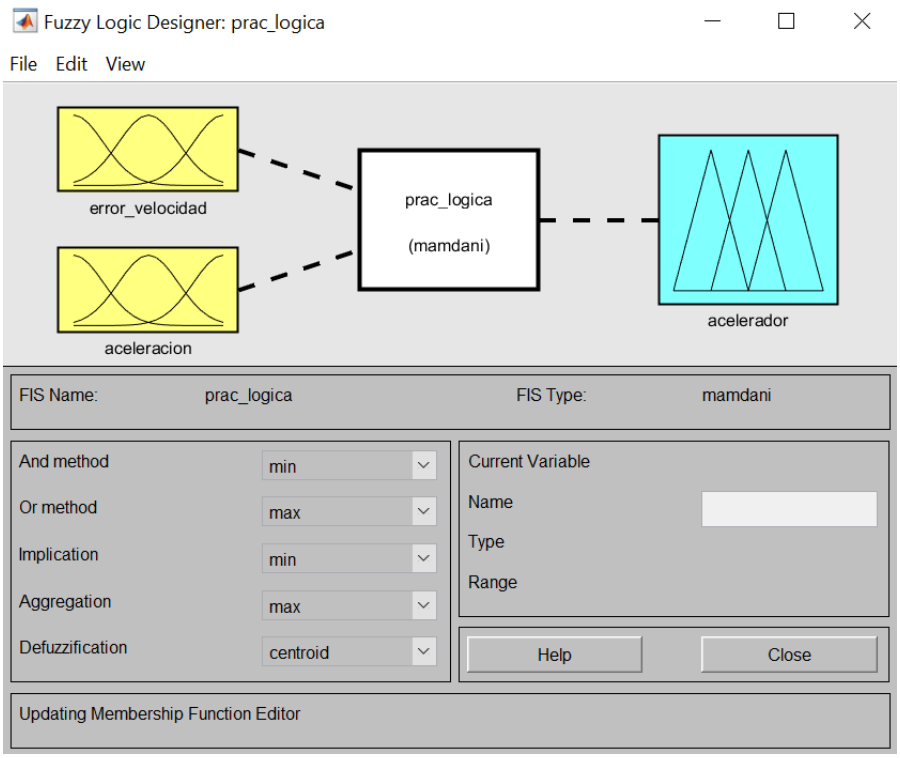

Figura 2.24: Sistema borroso

Capítulo 3

Ejecución

El motor de inferencia se ha ejecutado con un error de velocidad de -10km/h y una aceleración de 1km/h/s.

Para realizar la ejecución, primero, se ha pulsado en la opción View -> Rules (figura 3.1).

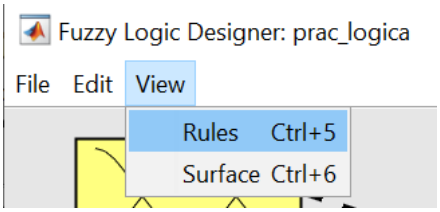

Figura 3.1: Ejecutar - Paso 1

Y seguidamente, se han introducido los valores de entrada error_velocidad -10km/h y aceleración 1km/h/s. Es decir: [-10;1]. Como se puede ver en la figura 3.2.

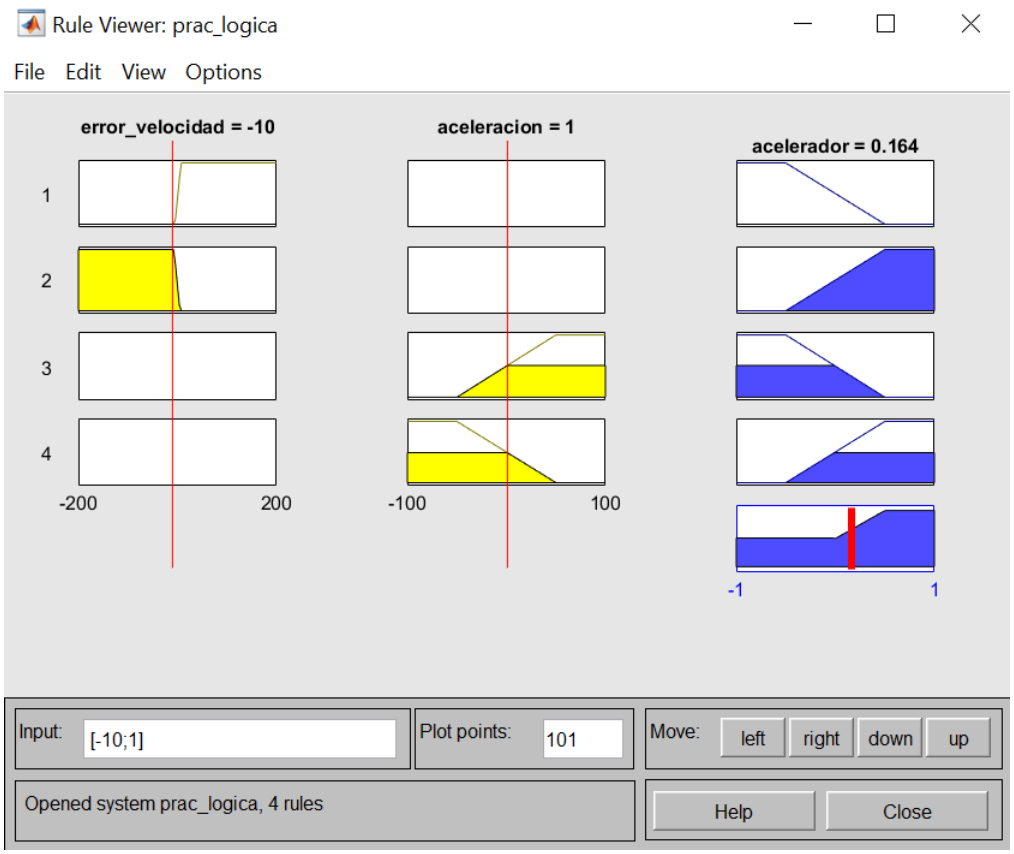

Figura 3.2: Ejecutar - Paso 2

En el capítulo siguiente se analizarán los resultados.

Capítulo 4

Análisis de resultados

Al ejecutar el motor de inferencia con un error de velocidad -10km/h y una aceleración de 1km/h/s se ha obtenido una salida de acelerador de 0.164km/h/s. Esto se puede ver en la figura 4.1.

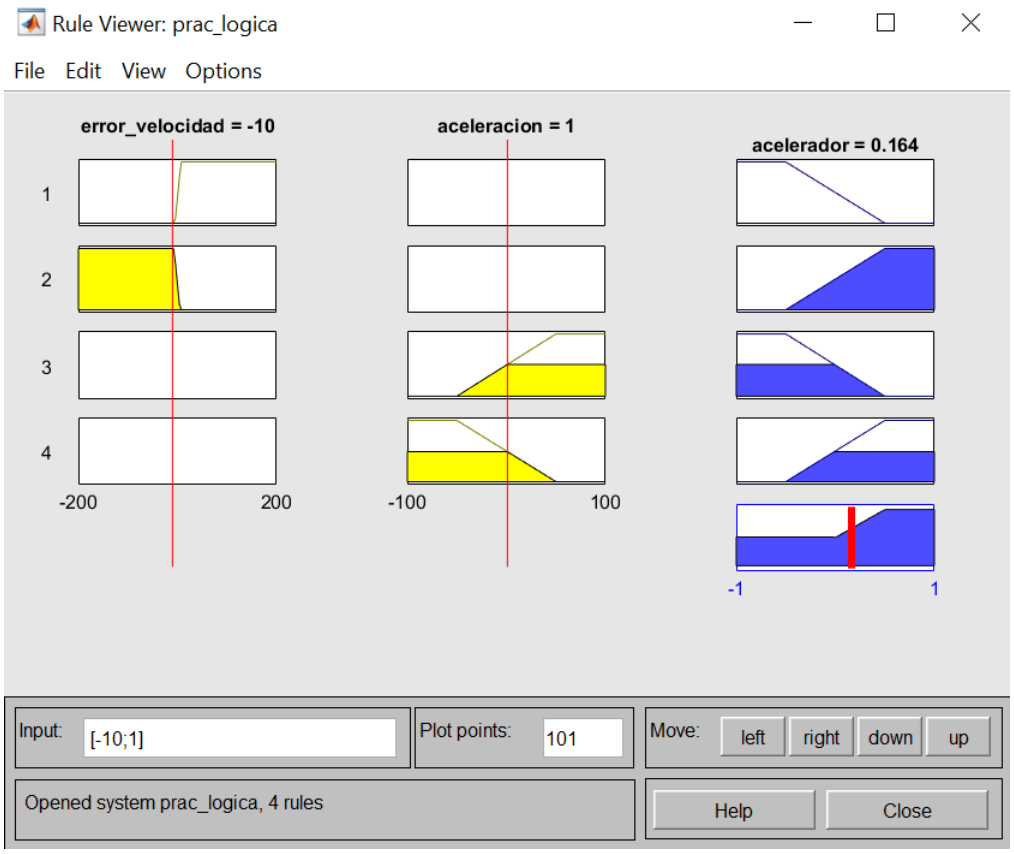

Figura 4.1: Ejecucion 1 - Reglas

Para analizar, es destacable recordar que el error de velocidad se calcula como sigue:
$Error\_vel = vel\_real - consigna\_vel$
Entonces, un valor de 0.164 tiene sentido porque al ser el error de velocidad negativo entonces el acelerador pisa, y al ser la aceleración positiva el acelerador levanta. Esto se da por las dos reglas:

- SI Error_vel negativo ENTONCES Acelerador pisa
- SI Aceleracion positiva ENTONCES Acelerador levanta

Como el valor de error de velocidad incidirá más entonces pisará más que levantará. Por tanto, un valor de 0.164 de acelerador tiene sentido ya que así pisará, pero no mucho. Esto último se puede ver en la gráfica 2.12 de las funciones de pertenencia del conjunto borroso de salida Acelerador.

Al ejecutar, también podemos obtener la superficie pulsando View -> Surface. Esta se puede observar en la figura 4.2. En esta superficie podemos ver una correlación entre el acelerador, la aceleración y el error de velocidad.

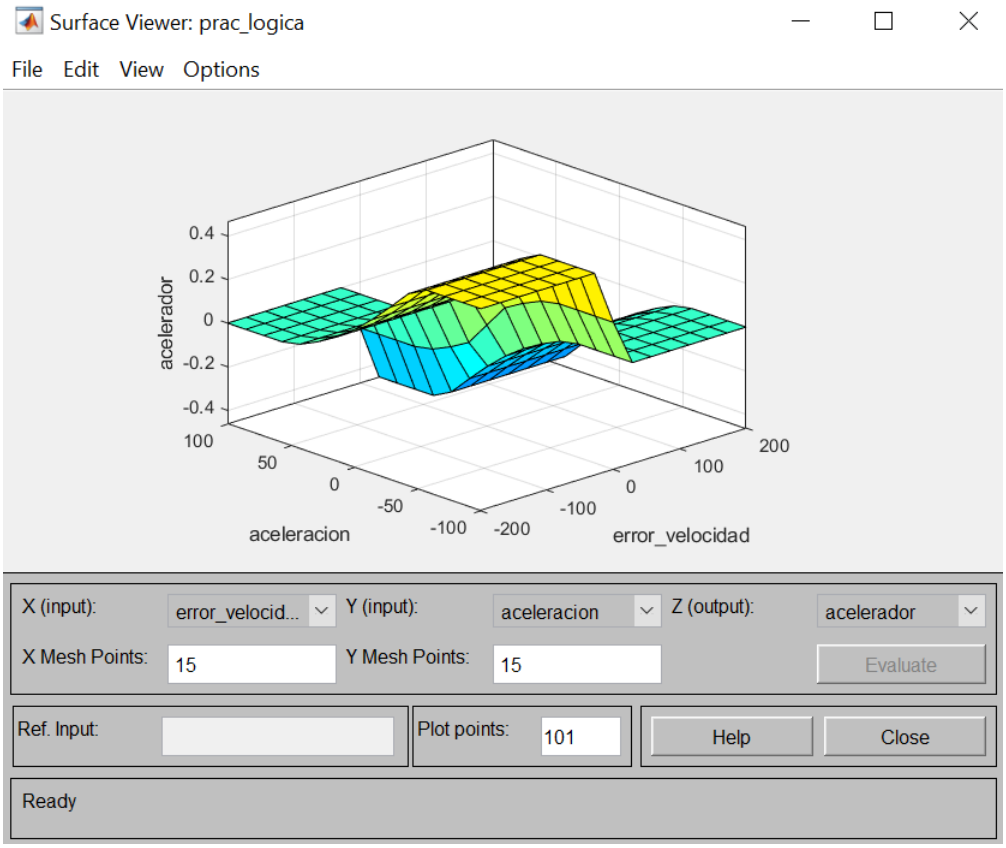

Figura 4.2: Ejecucion 1 - Superficie