4.9 Fuzzy control

The 9S12 microcontroller has an interesting feature: It supports evaluation of simple fuzzy inference systems (FIS). As this has been implemented as a hardware feature of the microcontroller core, the resulting fuzzy controller is generally very fast.

The toolbox provides a fuzzy control block which can be configured as a 1 to 3 input, 1 output Mamdani FIS. In our undergraduate laboratories we use the free MATLAB fuzzy toolbox *FlouLib*, (www.listic.univ-savoie.fr/modules.php?name=Content&pa=show&acronym=FlouLib) developed at the Université de Savoie, Chambéry Annecy (France) – Laboratoire d'Informatique, Systèmes, Traitement de l'Information et de la Connaissance (LISTIC). This toolbox provides a number of Simulink S-Function blocks for fuzzy logic systems using Mamdani FIS as well as Takagi-Sugeno FIS. The membership functions used during fuzzification and defuzzification can be defined using simple text files. The same holds for the rule base. Figure 4-34 shows an example of a simple Proportional-Derivative (PD) fuzzy controller for an inverted pendulum experiment. As this controller sits in the feedback branch of the control loop, the inputs have been placed on the right-hand side, whereas the outputs are on the left.

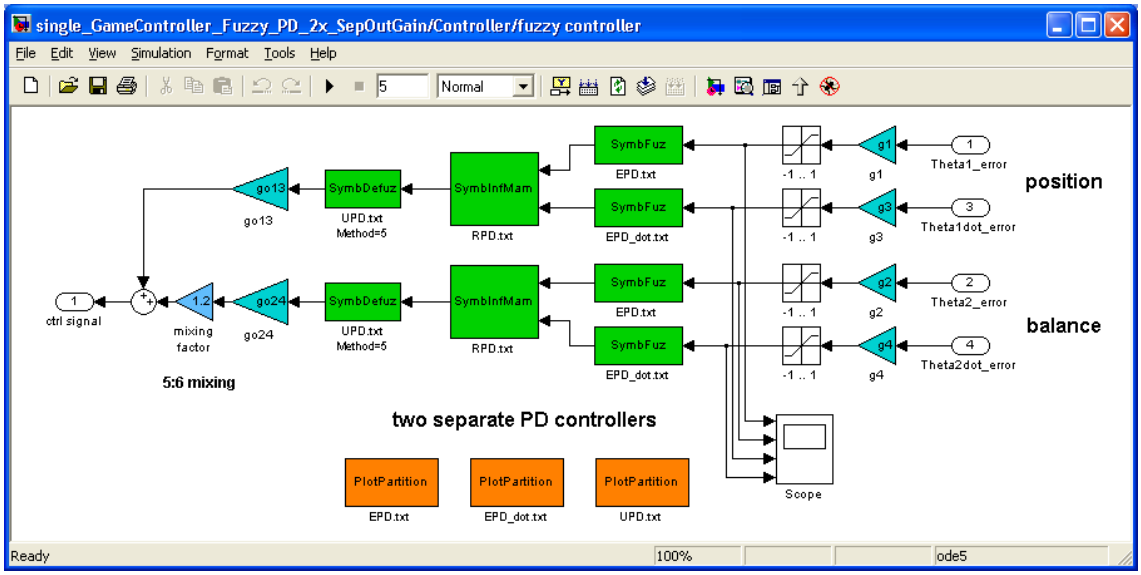

Figure 4-34 Inverted pendulum control using FlouLib blocks

Fuzzification of the inputs (position error $\theta_{1,err}$ and rate of change of the position error $d\theta_{1,err}/dt$) can be achieved using a simple text file, which is as call-up parameter in the *SymbFuz* block. In the case of the inverted pendulum example, three linguistic terms are used: N_1 (negative position errors), Z (zero-ish position errors) and P_1 (positive position errors). The input membership functions (MSF) can be defined as follows:

```
3
N1  -1  -1  -1  0
Z   -1   0   0  1
P1   0   1   1  1
```

Note that the input scaling gains g_1 – g_4 have been chosen to arrive at an effective universe of discourse of ±1. The output membership functions are defined in a similar way. In the inverted pendulum example we have chosen a total of 5 output MSFs (N_2,

N_1, Z, P_1 and P_2). The effective universe of discourse is mapped onto the normalized range from -1 to 1. An output scaling gain (g_o) ensures that the required signal levels are achieved.

```
5
N2 -1.5   -1   -1 -0.5
N1   -1 -0.5 -0.5    0
Z  -0.5    0    0  0.5
P1    0  0.5  0.5    1
P2  0.5    1    1  1.5
```

Finally, the rule base is specified in form of another text file. With 2 inputs (PD) and 5 output membership functions, the following rule base can be specified:

```
2 5
N1  N1  N2
N1   Z  N1
N1  P1   Z
 Z  N1  N1
 Z   Z   Z
 Z  P1  P1
P1  N1   Z
P1   Z  P1
P1  P1  P2
```

The shown example produces a linear control surface. The fuzzy controller therefore emulates a linear PD controller.

Once a FlouLib simulation works satisfactorily, the fuzzy controller files (definition of the input and output MSFs, rule base) can directly be used with the fuzzy block of *rtmc9s12-Target*. Simply enter the filenames in the corresponding edit boxes of the block parameter window (Figure 4-35).

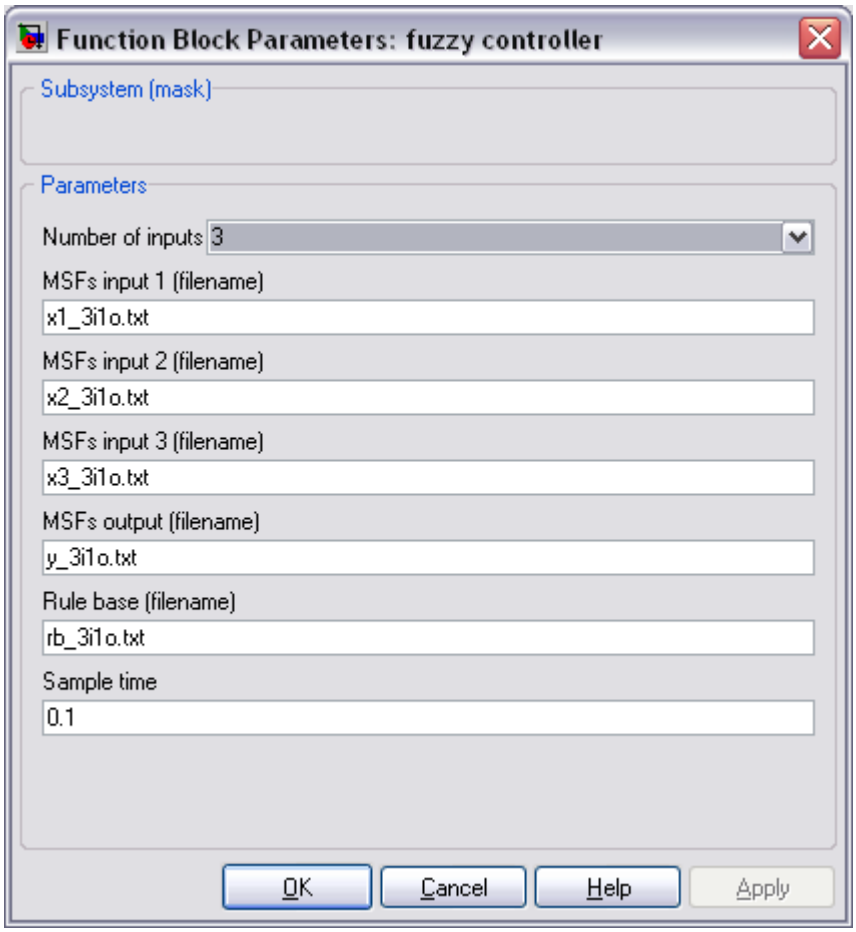

Figure 4-35 Using FlouLib definition files with the *rtmc9s12-Target* fuzzy block

The toolbox uses the settings found in the input, output and rule base files to generate a customized assembler code fuzzy engine. This source code file is then added to the list of files to be compiled into the target. To the user, all of this is fully transparent. Simply specify the required definition files and compile the model. Figure 4-36 shows a sample model which can be used to validate the correct operation of a fuzzy controller.

The fuzzy block allows for interesting lab exercises. In our laboratory we ran a small mobile robot exercise in which the students were asked to design a fuzzy controlled automatic parking system for a small mobile robot. Using a simple MATLAB based robot simulator, the students developed code to process frames from a camera / simulated stream of camera images to detect position and orientation of the robot. This information was then sent to the robots using the wireless RF links. On the robot, the current distance to the target was worked out, as well as the misalignment of the current orientation vector and a vector from the current location of the robot to the target. These two inputs were then fed to a fuzzy controller in charge of steering the robot around the workspace. The latter had been tuned using the simulator. As the simulator uses the very same algorithms and controller settings as the real application, the commonly big step from simulation to application was rather straight forward. Figure 4-37 shows the simulator used in conjunction with this exercise.

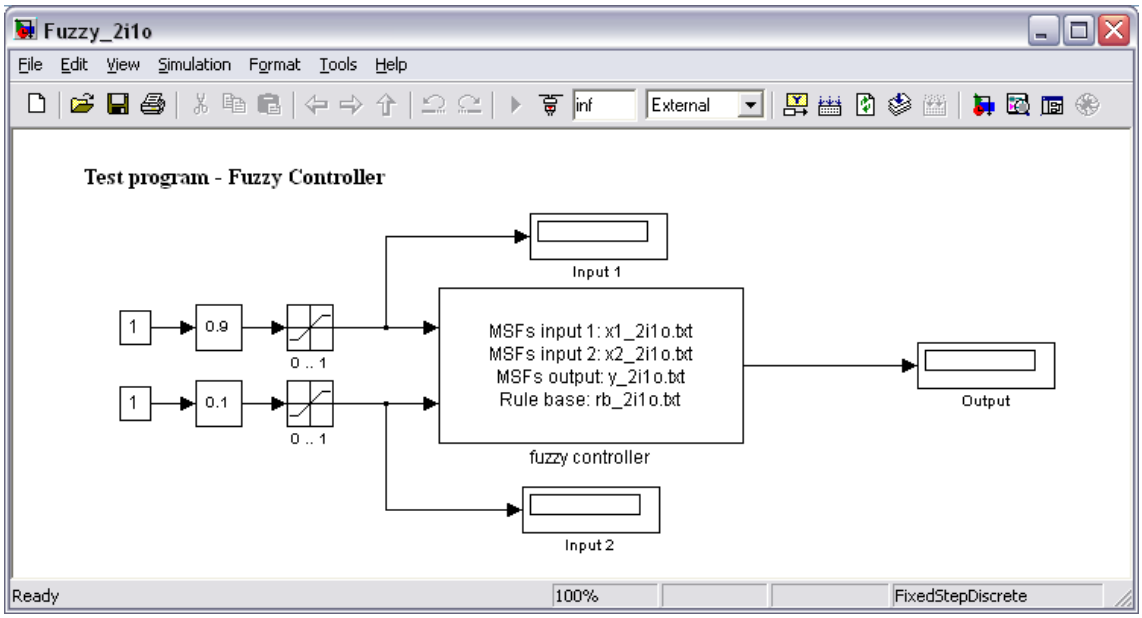

Figure 4-36 Testing a fuzzy controller on the target

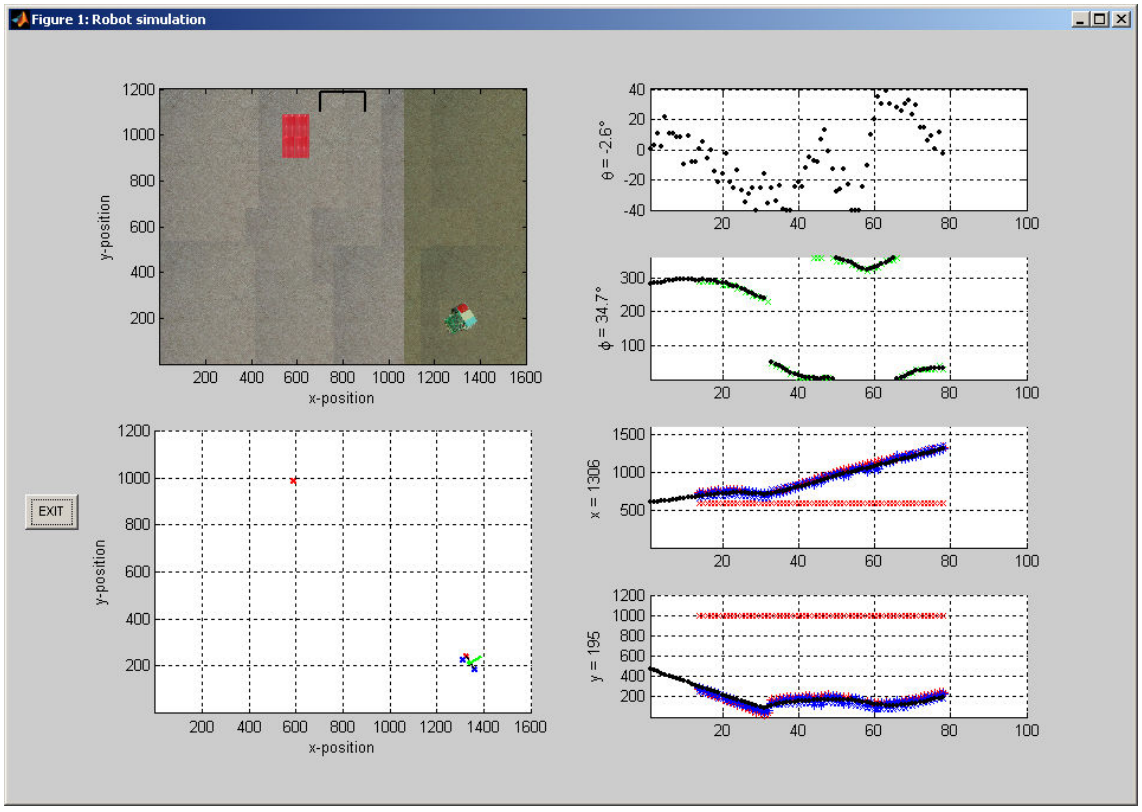

Figure 4-37 Robot control exercise using an *rtmc9s12-Target* fuzzy controller