Practical Course MATLAB/Simulink

STATEFLOW

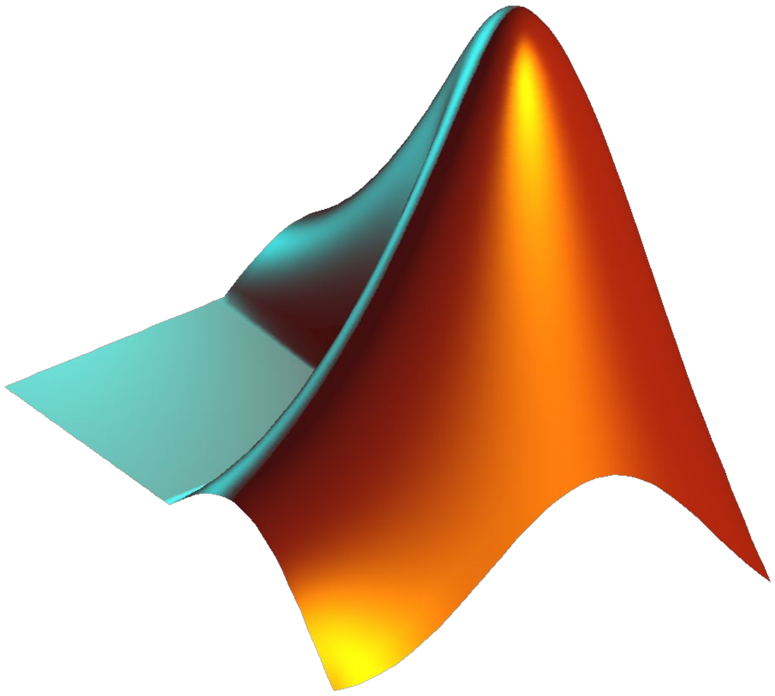
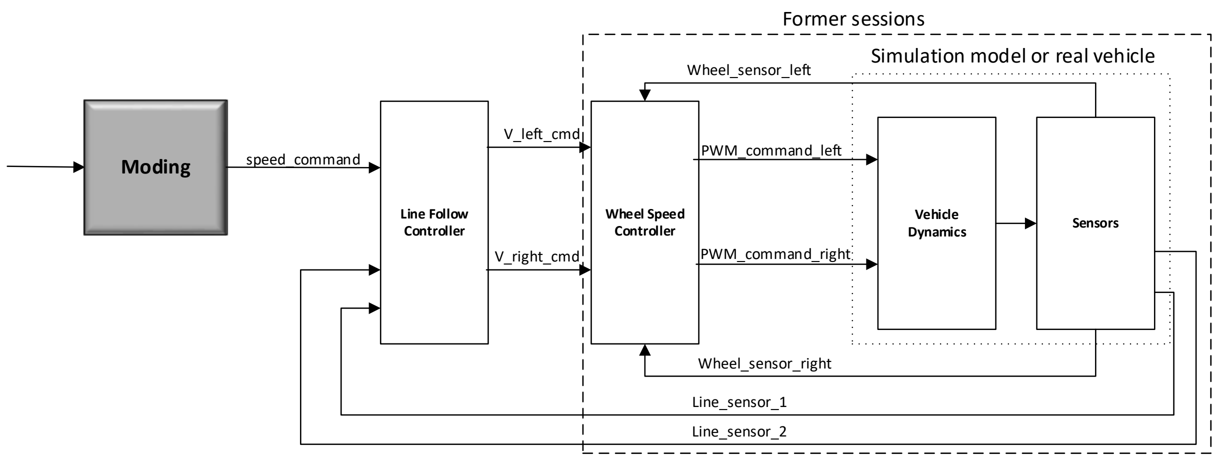


Table of Contents

0 General Information and Advice

The following tutorial contains exercises, which will lead you to a moding control, performed by a Stateflow model. Figure 1 shows a structural breakdown of the whole system, as developed in the preceding sessions.



1. In the former sessions, you implemented a model of the robot and an inner loop controller, which is able to command PWM's to the model.

2. For the complete system, there is also a line follow controller, which is given and which is able to separate a given speed command into commands for each wheel. The speed_command shall be implemented as a uint8 [0 stop, 127 slow, 255 fast]

3. Within the present session, a moding for the controller shall be implemented. The moding shall comply the following functionalities of the robot:
    - The robot shall be capable of the modes "fast", "slow" and "stop".
    - The operator shall be able to switch between "fast", "slow" and "stop"
    - The moding shall be implemented using the "Mealy" structure and the "C" action language.

4. Your upload for this exercise should include the following files
    - One Tutorial_08.pdf where the answers to the questions in this file are included.
    - A 'Robot_Moding.slx' file with your statechart implementation.
    - A 'testbench_stateflow_<XY>.slx' file with your implementation for the testing framework. Make sure to replace <XY> with the Matlab version you used.

1 Requirements and Preliminaries

In the following exercise, you will design an adequate stateflow moding structure to comply with the requirements. Multiple solutions exist for this task.

Exercise (23 Points)

(1) Which States will be necessary for such a moding?

stop
slow
fast

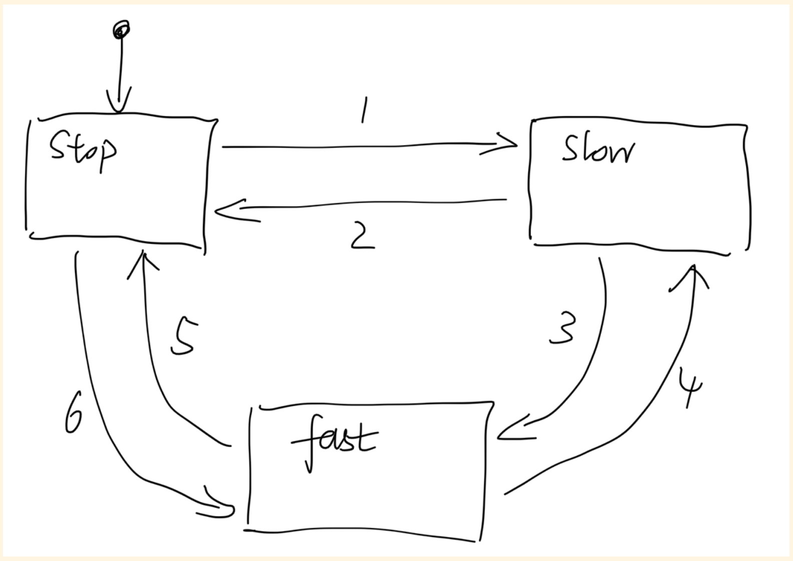
(2) What Inputs does the Stateflow chart need? Each command shall be implemented as a separate flag.

Signal Name	Signal Description
speed_off_flg (bool)	to activate the mode "stop"
speed_low_flg (bool)	to activate the mode "slow"
speed_high_flg (bool)	to activate the mode "fast"

(3) Which output is necessary for the Stateflow chart to control the robot?

Signal Name	Signal Description
speed_command	to represent the mode as an uint number

(4) Now think about the connections between the states: Which states do need a transition. Use the figure on the next page to draw it. Also use a numbering for each transition (for task 5). (10P)



(5) Write down in bullet points the conditions for each transition, you found.

Transition Number	Condition
1	[speed_low_flg == true]
2	[speed_off_flg == true]
3	[speed_high_flg == true]
4	[speed_low_flg == true]
5	[speed_off_flg == true]
6	[speed_high_flg == true]

2 Construction of the Stateflow model

In this exercise you will implement the stateflow model based on the preliminaries of chapter 1. While chapter 1 focussed on the moding content, this chapter will now deal with the implementation aspects of the Stateflow model.

Exercise (18P)

(1) Open a new simulink model, add a stateflow chart to it and save it under the name.

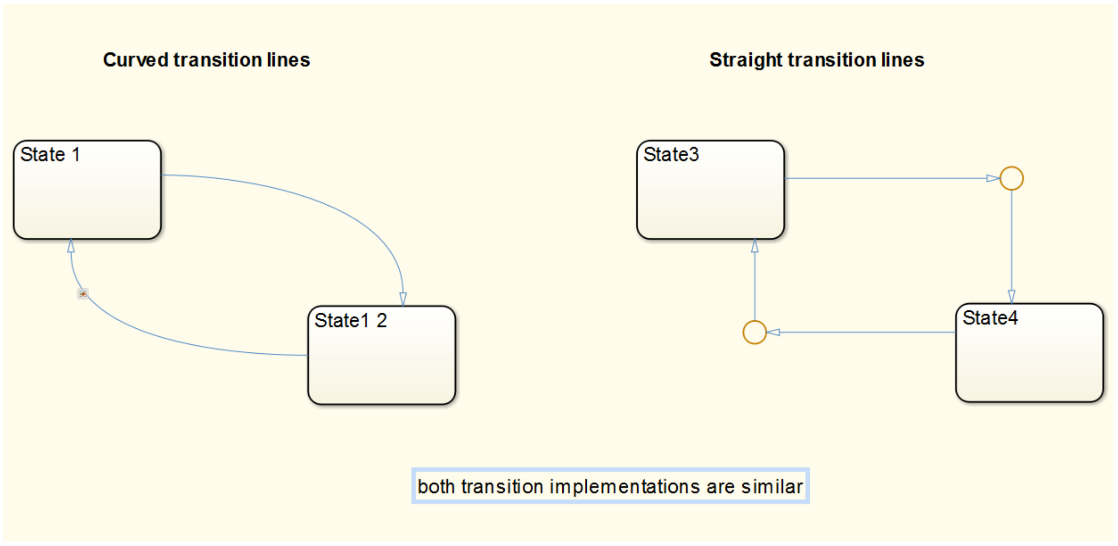
Robot_moding.slx

(2) Since we want a "Mealy" chart with "C" as action language, set those options correctly within the stateflow preferences.

(3) Add the inputs and the outputs to the stateflow charts, as specified in chapter 1. Also consider the datatype here.

(4) Now add the States to the Stateflow model, which you have specified in Chapter 1. Name all the states. Also add transition lines between the states structure for the states. Think about, which states, as specified in Chapter 1. Remind: Your model should be readable, even after you also inserted your transitions.

Hint *You can use junction blocks also to avoid curved transition lines.*


Curved transition lines
Straight transition lines
State 1
State1 2
State3
State4
both transition implementations are similar

(5) Implement now the conditions and actions for each transition.

(6) Save the model.

3 Model Test

In this exercise, you will design a testplan for the stateflow model and you will test the statechart, you have implemented in exercise 2.

Exercise (11P)

(1) Open the model **testbench_stateflow.slx**

You see a block, which can be used to stimulate your statechart. The constant on the left side can be changed during simulation and represents the command interface.

(2) The table below shows the necessary testcases for each transition within the statechart. Fill the coloumn "Expected Output" for each testcase. Note the expected output for each output signal.

Description	Required Input	Expected Output	Verified?
Stop -> slow	*Command_Input = 1*	speed_command = 127	yes
Slow -> fast	*Command_Input = 2*	speed_command = 255	yes
Fast -> slow	*Command_Input = 1*	speed_command = 127	yes
Slow -> stop	*Command_Input = 0*	speed_command = 0	yes
Stop -> fast	*Command_Input = 2*	speed_command = 255	yes
Fast -> stop	*Command_Input = 0*	speed_command = 0	yes

(3) Add a model reference to the Simulink model "testbench_stateflow.slx" and set the reference to the statechart model. (Simulink / Ports & Subsystems / Model)

(4) Connect the Test-stimulation block to the model reference. If you use another naming than the stimulation block, match the naming of the two models.

(5) Run the test plan as specified in the table. Note the test result in the third column of the table.

(6) If a check fails, find the reason and fix your model.

(7) When all tests are verified, save the model.

Congratulations. You finished the session "Stateflow"