Assignment 2b: Intro Stateflow

1DT059: Model-based Design of Embedded Software
Uppsala University

September 22, 2020

In this assignment you produce Stateflow (and Simulink) models. Each exercise should be implemented in a single .slx-file that is named after the exercise (the solution to assignment 2, exercise 2 should be in file `a2e2.slx`). Also include a file `a2.pdf` with a short high-level description of the ideas of your solutions (you do not need to include obvious things). Include your name at the top of all submitted files.

Assignments are to be solved by students individually. You can discuss ideas and concepts with fellow students, but it is absolutely forbidden to share or copy (even parts of) solutions, lines of code, or similar.

Problem 1 Billiard Ball*:*
(**15p**) Consider a Billiard table, which is just a rectangular surface with walls, with length $X = 2m$ and width $Y = 1m$. A billiard ball moves straight on the table after having been pushed (as normal), and bounces perfectly against walls.

a)(10p) Ignore holes, into which the ball may fall. Model the movement of a billiard ball in Simulink/Stateflow. Ignore friction (i.e., the ball will never stop, assuming perfect bounces). Assume balls are just points.

b)(5p) Let us add holes. So assume that corners have holes: the ball falls into a hole if it bounces within a small distance d from some corner. Let us say that $d = 5cm$. Add the feature that the ball "stops" when it falls in a hole.

c)(5p) Make it possible to see the movement of the ball (e.g., by a graphical plotting block or some animation which is shown during a paced simulation).

Problem 2 Elevator Controller*:*
(**45p**) In this problem, you will construct a Stateflow model of a controller for a simple elevator. You will later (after some modification) be able to use this model in the project on implementing an elevator controller. The elevator serves four floors (numbered 1, 2, 3, 4). There are 4 buttons, one for each floor, that can be pressed by users (for simplicity, we do not distinguish between buttons at floors and buttons inside the elevator, there is just *one* button for each floor). Whenever a button for floor n has been pressed, a light connected to that button is lit, and the elevator eventually (possibly after passing and stopping at some other floors) arrives to floor n. When this happens, the button is "unpressed" (e.g., its light goes off) and the doors open. The state of doors should be modeled by another light, which goes on when doors are open, and which goes off when they are closed. The elevator then stays at this floor until the control unit orders it to move to some other floor. Once the doors are open, they remain open for at least 5 seconds. The elevator must close the doors before leaving a floor.

The control unit controls the elevator itself via an output signal *speed*, which is the speed of the elevator cart. For simplicity, we can use the unit $floors/s$, where 0 speed means "stopped", positive speed means "going up", and negative speed means "going down". The maximum absolute speed of the elevator is $0.2 floors/s$ (i.e., it takes at least 5 seconds to go from one floor to the next). The elevator informs the control unit about its position via the signal *position* (an input to the controller), whose value is a real number between 1 and 4 (i.e., the unit is *floors*).

Apart from the input signal *position*, the controller has input events generated by pushing buttons. These are the above mentioned buttons for each floor (*floorn* for $n = 1, 2, 3, 4$). There is also an emergency button: when it is pushed, the elevator should immediately go to floor 1 and remain there (you can also have a *start* button to start and re-start the elevator after an emergency).

Output signals from the controller are (in addition to *speed*), one lamp (either on or off) for each floor button, as described above, one door lamp, indicating that doors are open, and an emergency lamp, indicating that the emergency button has been pressed.

a) (30p) Make a Stateflow model of the controller, which interacts with buttons, lamps, and a simple Simulink model of the elevator (with input *speed* and output *position*. The requirements include typical "elevator-behavior" requirements, including that: whenever the button for floor n has been pressed, the elevator must eventually arrive to floor n within some reasonable time. When this happens, the "door" lamp is lighted, signalling that the doors are open. Once the doors are open, they remain open for at least 5 seconds. The doors must

close (i.e., the lamp go off) before leaving any floor. Whenever the "emergency" button is pressed, the elevator is stopped and the emergency lamp is lit. The lamp is switched off and the elevator resumes to normal when the "start" button is pressed again.
You should produce a well structured controller model, which is decomposed in a logical way following the functionality of the controller. One subproblem is to decide on where the elevator will go if it must serve several requests for where to go. This "decision algorithm" should be clearly localized, so that it is easy to replace it by another one which is optimized for some particular purpose.

b) (15p) Equip your model with some animation, which shows the position of the elevator, the doors, and the state of buttons (lights). The user should be able to interact with the elevator by "pushing buttons".

Save your solution to this exercise; you will use it again later.

Submission

Solutions (all files) to this assignment are to be submitted via the Student Portal by **Wednesday, September 30, 2020**