SOEN331: Introduction to Formal Methods for Software Engineering Assignment 2 on Extended Finite State Machines

Martin Marcos 40041398,

Samantha Guillemette 26609198,

Deepkumar Patel 40096716

February 29, 2020

1 Driver-less car system formal specification

The EFSM of the driver-less car system is the tuple $S = (Q, \Sigma_1, \Sigma_2, q_0, V, \Lambda)$, where

$Q = \{idle, parked\ mode, manual\ mode, cruise\ mode, marked\ mode, panic\ mode, exit\}$

$\Sigma_1 = \{start, cruise\ signal, switch, arrived, unforseen, panic\ off, off\}$

$\Sigma_2 = \{beep, hazard\ off\}$

$q_0 : idle$

$V : destination = \{set, no\}$

Λ: Transition specifications

1. $\rightarrow idle$
2. $idle \xrightarrow{\text{start}} parkedmode$
3. $parked\ mode \xrightarrow{\text{off}} off?$
4. $parked\ mode \xrightarrow{\text{cruise signal [no dest]}} manual\ mode$
5. $parked\ mode \xrightarrow{\text{cruise signal [set dest] / beep}} cruise\ mode$

6. $\textit{manual mode} \xrightarrow{\text{cruise signal [set dest]}} \textit{cruise mode}$
7. $\textit{cruise mode} \xrightarrow{\text{switch}} \textit{manual mode}$
8. $\textit{cruise mode} \xrightarrow{\text{arrived}} \textit{parked mode}$
9. $\textit{cruise mode} \xrightarrow{\text{unforseen}} \textit{panic mode}$
10. $\textit{manual mode} \xrightarrow{\text{stop}} \textit{marked mode}$
11. $\textit{panic mode} \xrightarrow{\text{panic off / hazard off}} \textit{manual mode}$

The UML state diagram is shown in Figure 1

As *cruise* is a composite state, it is defined as the tuple $S = (Q, \Sigma_1, \Sigma_2, q_0, V, \Lambda)$, where

$Q = \{\textit{tailing mode}, \textit{changing lane mode}, \textit{navigation mode}\}$

$\Sigma_1 = \{\textit{i to c}, \textit{t to c}, \textit{c to t}, \textit{c to n}, \textit{n to c}\}$

$\Sigma_2 =$

$q_0 : \textit{tailing mode}$

$V :$

Λ: Transition specifications

1. $\rightarrow \textit{tailing mode}$
2. $\xrightarrow{\text{i to c}} \textit{changing lane mode}$
3. $\textit{tailing mode} \xrightarrow{\text{t to c}} \textit{changing lane mode}$
4. $\textit{changing lane mode} \xrightarrow{\text{c to t}} \textit{tailing mode}$
5. $\textit{changing lane mode} \xrightarrow{\text{c to n}} \textit{navigation mode}$
6. $\textit{navigation mode} \xrightarrow{\text{n to c}} \textit{changing lane mode}$

The UML state diagram is shown in Figure 2

As *tailing* is a composite state, it is defined as the tuple $S = (Q, \Sigma_1, \Sigma_2, q_0, V, \Lambda)$, where

$Q = \{\textit{accelerate}, \textit{decelerate}, \textit{change lane mode}\}$

$\Sigma_1 = \{\textit{obstacle}\}$

$\Sigma_2 = \{t\ to\ c, maintain\ speed, switch\ lane\}$

$q_0 : tailing\ mode$

$V : speedOfCar : \{minSpeedRange, maxSpeedRange\},$

$distanceOfCar, minSpeedRange, maxSpeedRange, minDistance : \mathbb{R}$

Λ: Transition specifications

1. $\xrightarrow{\text{[s<minSpeedRange]}} accelerate$
2. $\xrightarrow{\text{[maxSpeedRange <s>minSpeedRange]/t to c; maintain speed}} change\ lane\ mode$
3. $\xrightarrow{\text{obstacle[d<minDistance]/t to c; switch lane}} change\ lane\ mode$
4. $\xrightarrow{\text{[s>minSpeedRange or d<minDistance]}} decelerate$

The UML state diagram is shown in Figure 3

As *changing lane* is a composite state, it is defined as the tuple $S = (Q, \Sigma_1, \Sigma_2, q_0, V, \Lambda)$, where

$Q = \{maintain\ car\ speed, tailing\ mode, change\ lane\ mode, cruise\ mode, panic\ mode\}$

$\Sigma_1 = \{maintain\ speed, switch\ lane, cannot\ change\ lane\}$

$\Sigma_2 = \{c\ to\ t, panic\}$

$q_0 : changing\ lane\ mode$

$V : targetLane : \{car\ in\ t, car\ not\ in\ t\},$

$speedOfCar : \{minSpeedRange, maxSpeedRange\},$

$distanceOfCar, minSpeedRange, maxSpeedRange, minDistance : \mathbb{R}$

Λ: Transition specifications

1. $\xrightarrow{\text{maintain speed [d>= minDistance]}} maintain\ car\ speed$
2. $maintain\ car\ speed \xrightarrow{\text{[d>= minDistance]}} maintain\ car\ speed$
3. $maintain\ car\ speed \xrightarrow{\text{[d<minDistance]/c to t}} tailing\ mode$
4. $\xrightarrow{\text{maintain speed [d<minDistance]/c to t}} tailing\ mode$
5. $\xrightarrow{\text{maintain speed [s>maxSpeedRange \& s<minSpeedRange]/c to t}} tailing\ mode$
6. $\xrightarrow{\text{switch lane[car not in t]}} change\ lane\ mode$
7. $\xrightarrow{\text{switch lane[car in t]/c to n}} cruise\ mode$

8. $\xrightarrow{\text{switch lane;cannot change lane/panic}} \textit{panic mode}$

9. $\textit{change lane mode} \xrightarrow{\text{[car not in t]}} \textit{change lane mode}$

10. $\textit{change lane mode} \xrightarrow{\text{[car in t]/c to n}} \textit{cruise mode}$

11. $\textit{change lane mode} \xrightarrow{\text{cannot change lane/panic}} \textit{panic mode}$

The UML state diagram is shown in Figure 4

As *navigation* is a composite state, it is defined as the tuple $S = (Q, \Sigma_1, \Sigma_2, q_0, V, \Lambda)$, where

$Q = \{turn\, left, turn\, right, turn\, left\, ahead, turn\, right\, ahead, changing\, lane\, mode, destination\, ahead, arr$

$\Sigma_1 = \{d\ on\ left, d\ on\ right, TLA, TRA, d\ ahead, car\ at\ d\}$

$\Sigma_2 = \{turn\ left, turn\ right, dest\ ahead, car\ in\ t, n\ to\ c, switch\ lane, arrived\}$

q_0 : *changing lane mode*

V : *targetLane* : $\{car\ in\ t, car\ not\ in\ t\}$,

Λ: Transition specifications

1. $\xrightarrow{\text{d on left/turn left}} turn\ left$
2. $\xrightarrow{\text{d on right/turn right}} turn\ right$
3. $\xrightarrow{\text{TLA[car not in t]}} turn\ left\ ahead$
4. $\xrightarrow{\text{TRA[car not in t]}} turn\ right\ ahead$
5. $\xrightarrow{\text{TLA/[car not in t]}} turn\ left\ ahead$
6. $turn\ left\ ahead \xrightarrow{\text{/n to c; switch lane}} changing\ lane\ mode$
7. $turn\ right\ ahead \xrightarrow{\text{/n to c; switch lane}} changing\ lane\ mode$
8. $destination\ ahead \xrightarrow{\text{car at d/arrived}} arrived\ destination$

The UML state diagram is shown in Figure 5

2 UML state diagrams

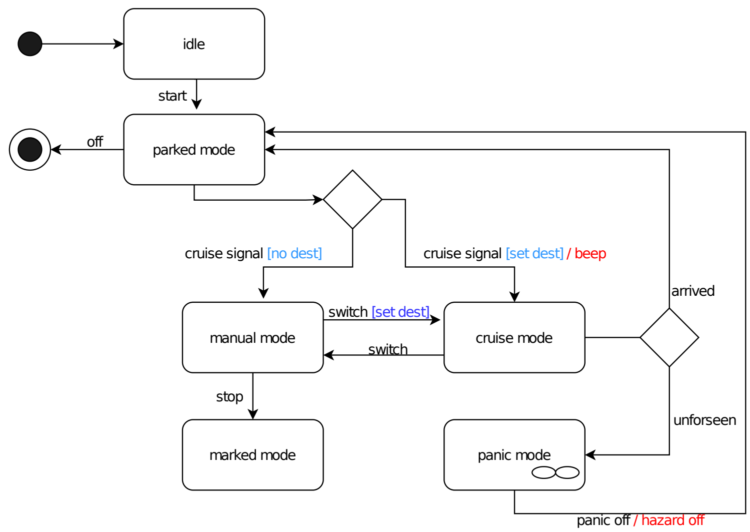

Figure 1: Main System.

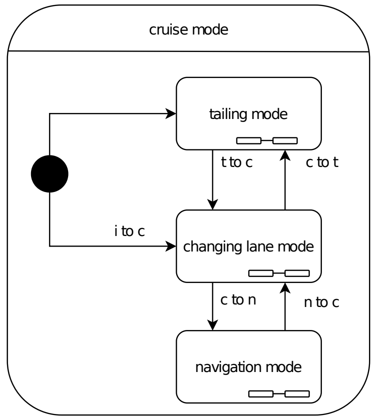

Figure 2: Cruise Mode.

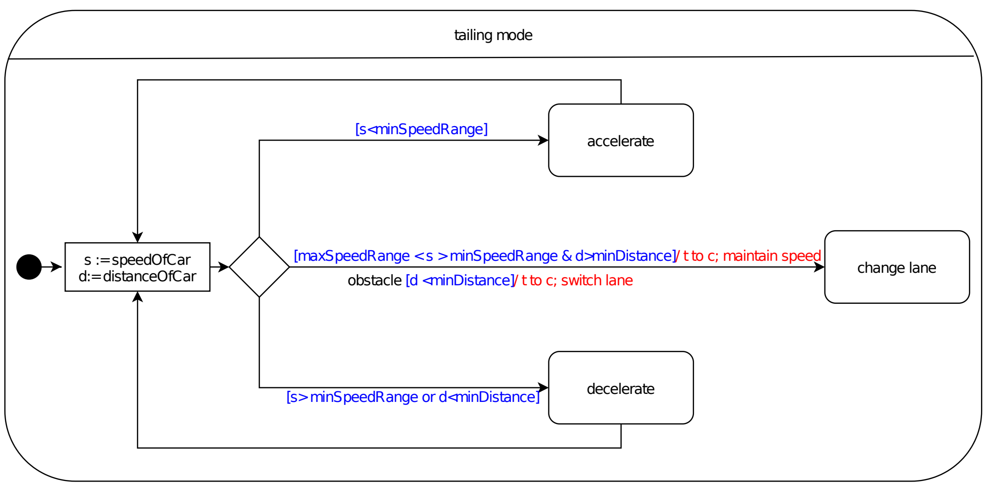

Figure 3: Tailing Mode.

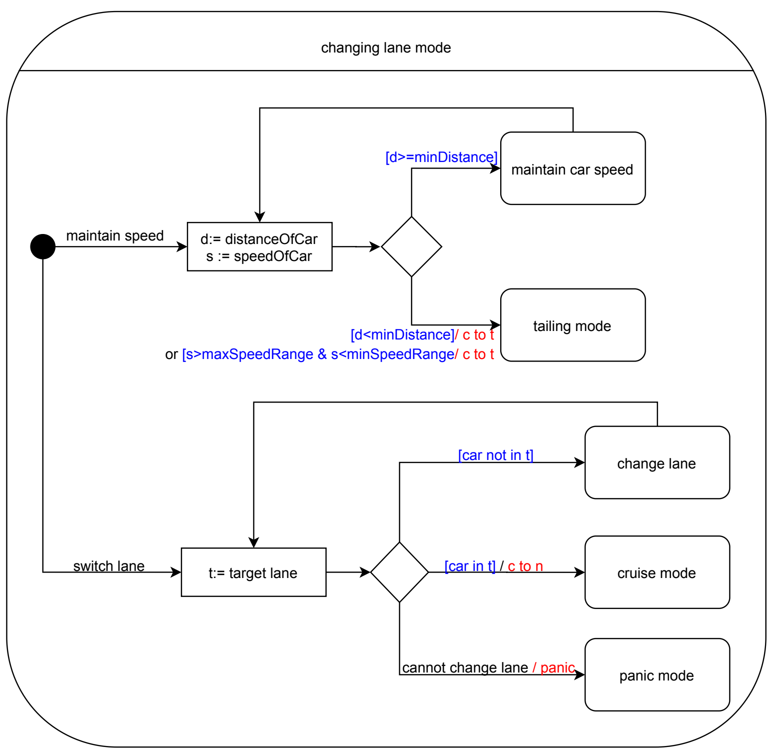

Figure 4: Changing Lane Mode.

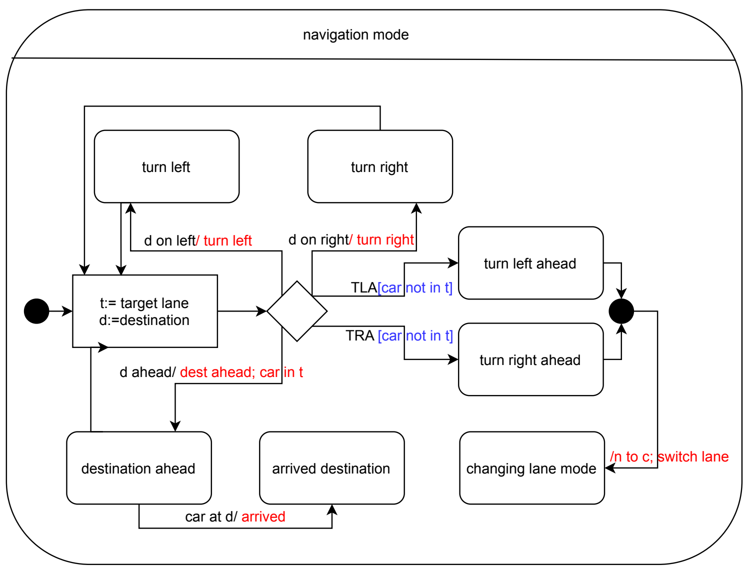

Figure 5: Navigation Mode.