Engineering and Physical Sciences

Report

B31DG – Embedded Software

Timothée Fréville

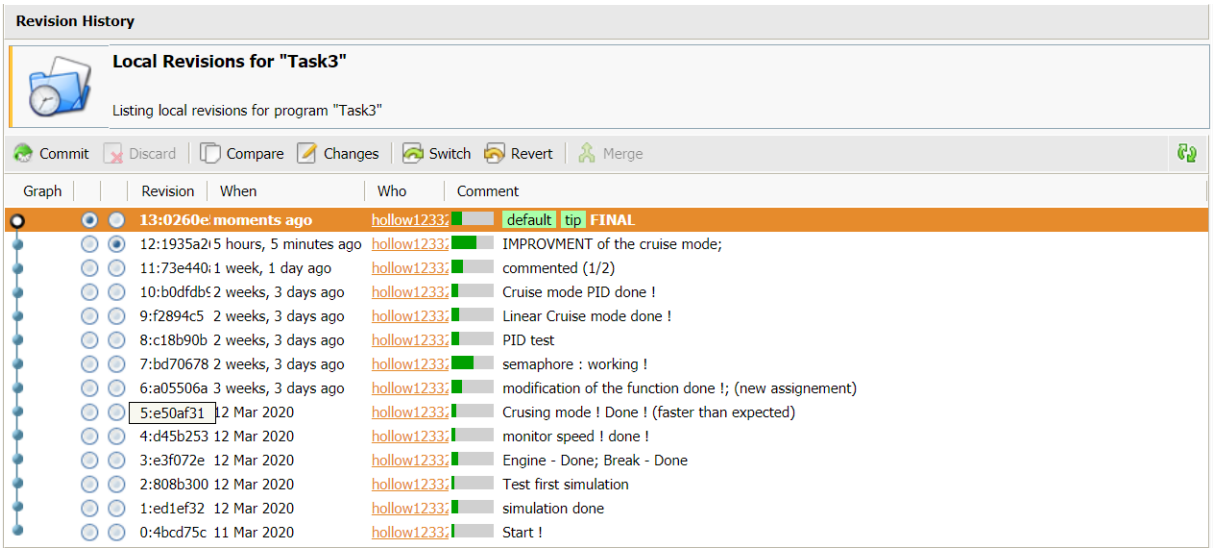
H00291146

Supervisor:

Prof. Mauro Dragone & Dr Abderrahim Halimi

Date of Submission: 20th March 2020

Commit



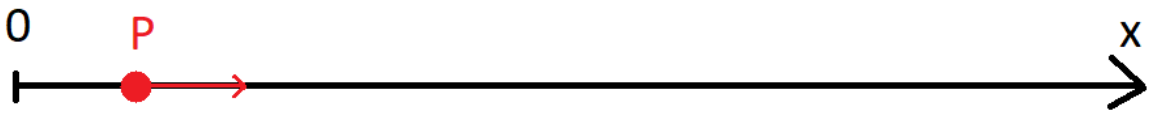
Car model

The care model is following the expectation of the assignment 3. The car will be assimilated as a point moving along a one coordinate using the 2nd law of Newton where:

$\ddot{x} : acceleration$

$\dot{x} : speed$

$x : movement$



If the acceleration is constant, we have:

$$\ddot{x(t)} = \alpha$$

$$\dot{x}(t) = \alpha t + \dot{x}(0)$$

$$x(t) = \frac{\alpha t^2}{2} + \dot{x}(0)t + x(0)$$

But our actual problem is much more complex because the α is not constant. Indeed α(t) is moving through time and only the user will determine it next value.

$$\alpha(t) = \left(acceleration(t) - brake(t)\right)$$

Besides, we are in a simulation. And we can consider the simulation as not analogical problem. Therefore, we can say that at every calculation of the acceleration at a time t will be constant. This way our equations presented before can work.

Thus, the speed will be coded:

$$\dot{x}(t) = \alpha t + \dot{x}(0)$$

$$current_{speed(t)} = \alpha t + current_{speed(t-1)}$$

$$current_{speed(t)} = \left(acceleration(t) - brake(t)\right) * t + current_{speed(t-1)}$$

And Consequently, the odometry will be coded:

$$x(t) = \frac{\alpha t^2}{2} + \dot{x}(0)t + x(0)$$

$$odom = \frac{current_{speed(t)} * t}{2} + current_{speed(t-1)t} + odom(t-1)$$

To treat the data, we will use some norms to avoid any confusion:

Distance: m

Speed: m/s

The conversion to mph will be done at the last moment. Consequently, we will do some conversion before the coding part.

m/s	Mph
40	88
22	50

Features

Here is the list of the different features to implement:

List:

Functions	Rates
Read Engine	2Hz (arbitrary)
Read Accelerator	2Hz (arbitrary)
Read Brake	2Hz (arbitrary)
Monitor speed	5Hz

LCD Display	2Hz
Cruise mode	20Hz
Simulation	25Hz

Functions:

Read engine: read the engine input and change a global variable that can only take 0 or 1.

Read Accelerator: if the cruise variable is at 0 then: read the acceleration input and change a global variable that takes +1 is the button is pressed and – 1 if it's not.

Read Brake: if the cruise variable is at 0 then: read the brake input and change a global variable that takes +1 is the button is pressed and – 1 if it's not.

Monitor Speed: limit the current speed at 88Km/h.

LCD Display shows the average value of the speed and the distance from the start point.

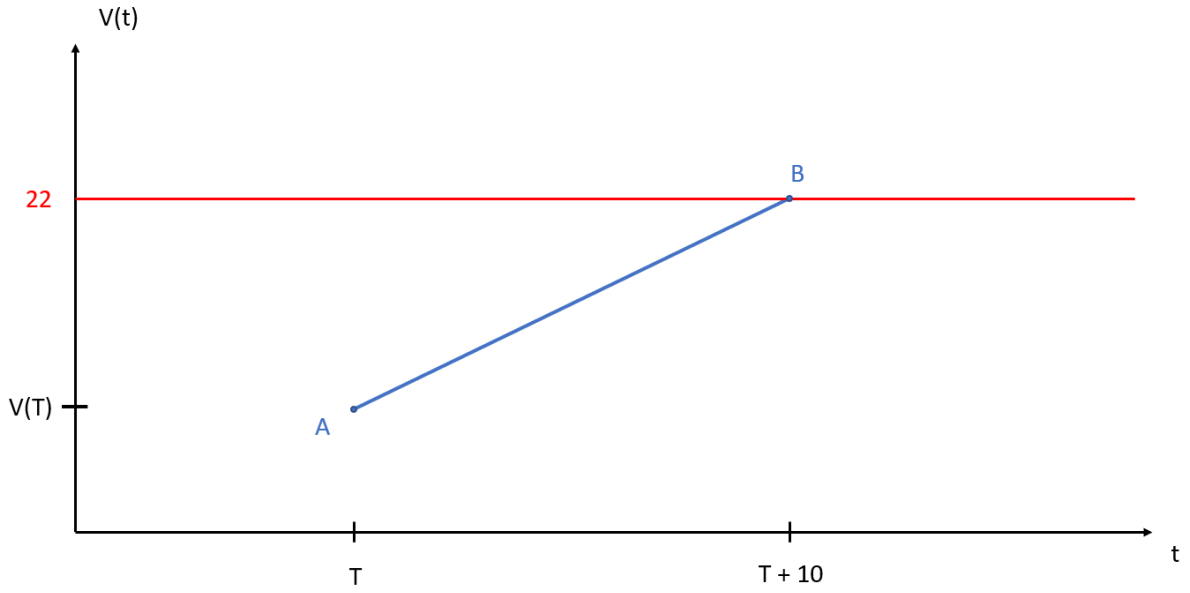
Cruise mode: if the button is pressed a global variable takes 1 and you will need to re push the button to disable the mode. Once this variable takes 1. The function will stop reading the inputs and control the acceleration and brake to reach and remains at the 55mph.

PID Controller

For the PID controller I have done two different method:

- **Linear Progression**

This approach is very simple we want the speed to draw a straight line until it reaches the speed wanted. I use simple mathematical formula to implement it. I wanted the cruise mode to reach it cruise speed in 10s.



The point A is the beginning of the cruise mode at a time T and with a speed that only the simulation knows X(T). B is the moment where the cruise mode reaches the speed wanted. We want our simulation to define the steep of this line by using this formula:

$$\alpha = \frac{V(B) - V(A)}{T + 10 - T}$$

$$\alpha = \frac{22 - V(T)}{10}$$

$\alpha : defines\ the\ steep$

Using this formula, the line will be draw for any case of scenario in terms of speed allowing the car to reach it cruise speed in 10s in a linear way.

If the speed is defined by this equation:

$$\dot{x}(t) = \alpha t + b$$

The steep in the model is literally the acceleration.

$$\ddot{x}(t) = \alpha$$

In our model:

$$\ddot{x}(t) = \alpha = Acceleration(t) - brake(t)$$

- **PID Controller**

We must keep in mind that a PID controller works only for certain type of controller which is not my case. Indeed, the acceleration gives us the speed by integration which means that if the accelerator is pressed then released the car will conserve it last speed. This method has for drawback to be

complicated to integrate PID controller. Nevertheless, we can still use the 1st order filter to answer this question.

$$\dot{x}(t) = K(1 - e^{\alpha t})$$

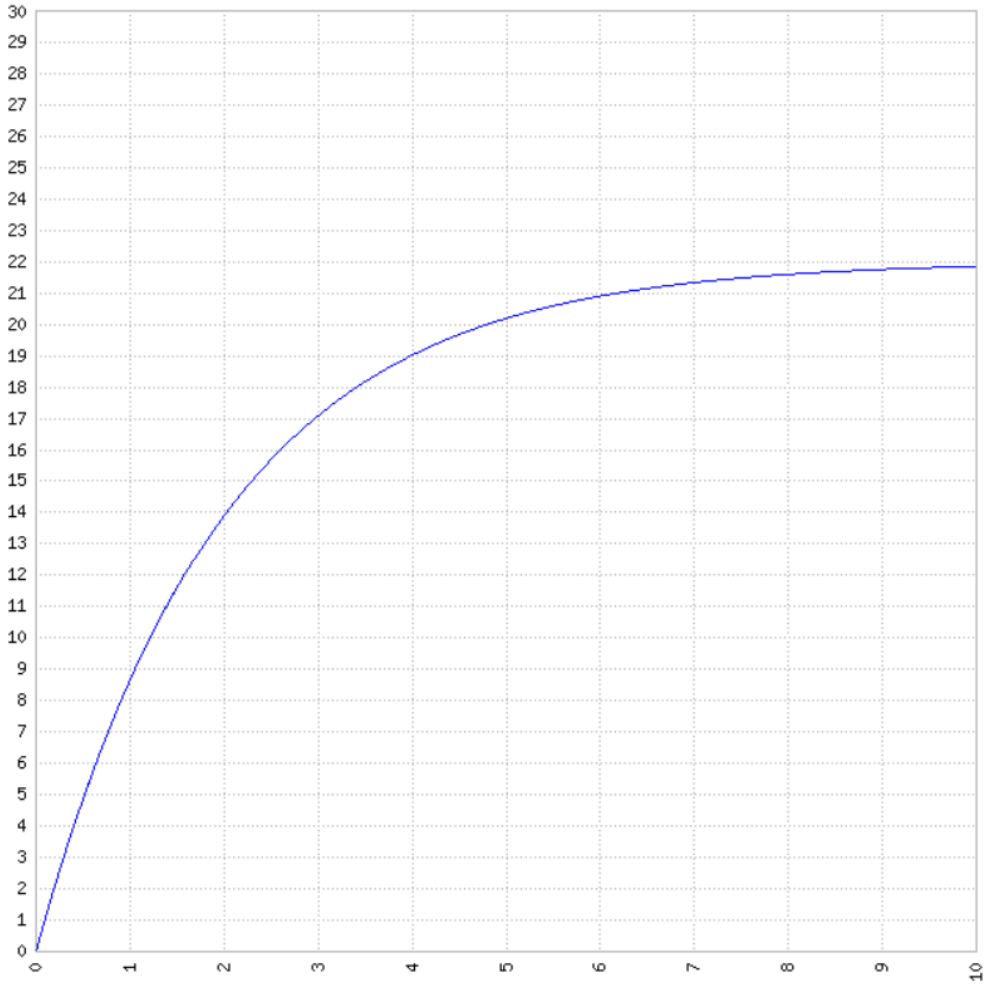
$K = 22$

$\alpha = 0.5$

We want the cruise mode to take 10s to reach 22m/s

Thus:

$$\dot{x}(t) = 22(1 - e^{\frac{-t}{2}})$$



Once we know the speed, we can derive the expression to find the acceleration.

$$\ddot{x}(t) = -\frac{1}{2} * 22 * -e^{\frac{-t}{2}}$$

$$\ddot{x}(t) = 11 * e^{\frac{-t}{2}}$$

In my case the total acceleration is (Acceleration – Brake) consequently:

$$\ddot{x}(t) = 11 * e^{\frac{-t}{2}} = Acceleration(t) - Brake(t)$$

The issue here is that 2 different function cannot be equal to an exponential. We can see here the limits of my model. However, to compensate this problem I've redefine the features of the car during the cruise mode. By doing this assumption.

$$\ddot{x}(t) = Acceleration(t)$$

Now the model and the equation can work properly. The models work from t=0 to t= infinity which means that our initial speed must be 0. In but in order to make it work for any initial speed we use the equation to find the t value that matches with the initial speed. The less the speed difference is the less time the cruise mode will take to reach it wanted speed. For a define t we have:

$$\dot{x}(t_0) = K(1 - e^{\alpha t_0})$$

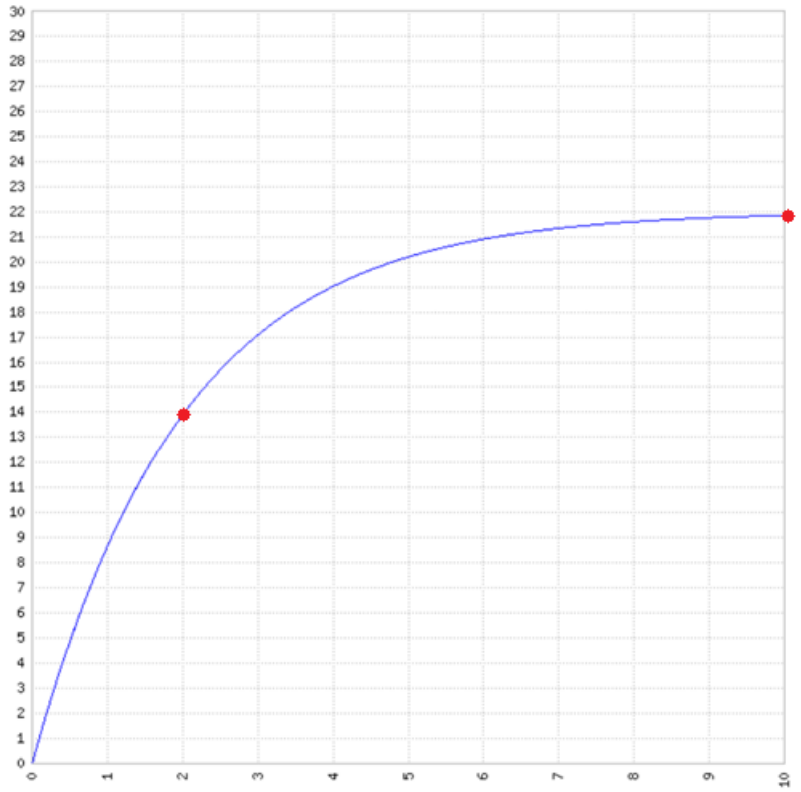
$$1 - e^{\alpha t_0} = \frac{\dot{x}(t_0)}{K}$$

$$e^{\alpha t_0} = -\left[\frac{\dot{x}(t_0)}{K} - 1\right]$$

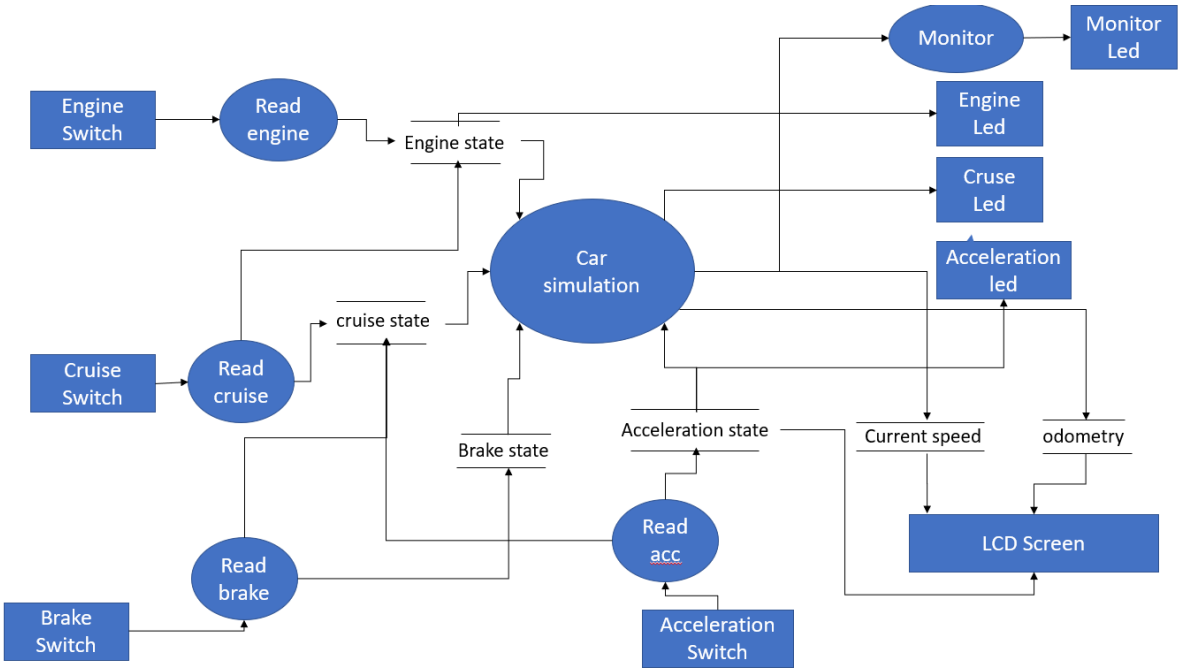
$$\alpha t_0 = \ln\left(-\left[\frac{\dot{x}(t_0)}{K} - 1\right]\right)$$

$$t_0 = \frac{\ln\left(-\left[\frac{\dot{x}(t_0)}{K} - 1\right]\right)}{\alpha}$$

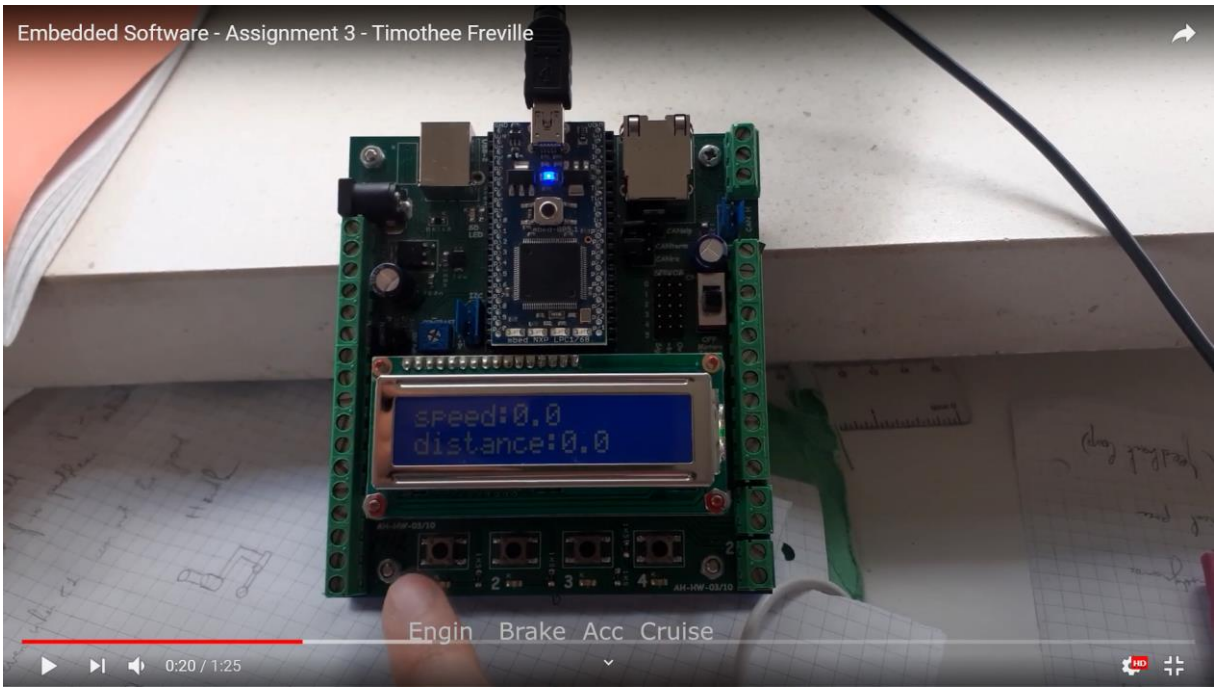
Now that we have this information we know where the speed in the curves is currently. Using this we can the equation go from t_0 to infinity. For example, in the graph below if the initial speed is 14 it means that the cruise mode will draw this curve at the t = 2.


30
29
28
27
26
25
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0
0
1
2
3
4
5
6
7
8
9
10

Dataflow


Monitor
Monitor Led
Engine Switch
Read engine
Engine state
Engine Led
Cruse Led
Car simulation
Acceleration led
cruise state
Cruise Switch
Read cruise
Acceleration state
Current speed
odometry
Brake state
Read acc
LCD Screen
Read brake
Brake Switch
Acceleration Switch

Video


Embedded Software - Assignment 3 - Timothee Freville
speed:0.0
distance:0.0
Engin Brake Acc Cruise
0:20 / 1:25