MBSD Assignment #2 A.Y. 2024/25

# Purposes

* Implement the “one pedal controller” as a Simulink model.

To implement the system, follow the description already provided in the Laboratory 1 document considering also eventual assumptions already stated in this latter document.

The safety mechanisms are not required in this laboratory.

The Simulink project has to be split into 3 files:

* **Harness.slx**, containing reference models for the controller and plant and test stimuli generation
* **Controller.slx**, containing the controller (to be developed)
* **Plant.slx**, containing the car longitudinal physical model.

The longitudinal physical model and the test stimuli generators shall demonstrate the effectiveness of the implemented controller with respect to the expected functionalities.

Consider the plant model description as an example of the comments to be inserted in the Controller Software Unit description report.

Templates of these files are available alongside this document.

The deliverable, composed of the 3 Simulink models and a PDF file obtained by filling the following pages of this document (please delete this first page), has to be provided as a .ZIP file up to **May 4th at 23:59 through “Consegna Elaborati” of the Portale della Didattica.** It shall also contain a brief report explaining the design of the controller, using the following template. It is sufficient that only one of the group members uploads it.

# Model-Based Software Design, A.Y. 2024/25

# Laboratory 2 Report

## Components of the working group (max 2 people)

* Federico Bena, 328197
* Michele Lauriola, 332679

## External interfaces of the plant

|  |  |  |
| --- | --- | --- |
| **Name** | **Direction** | **Type** |
| **Requested\_Torque\_Nm** | Input | CAN |
| **Vehicle\_Speed\_km\_h** | Outpu t | CAN |
| **Automatic\_Transmission\_Selector** | Input (from the driver to the controller) | CAN {P, R, N, D, P} |
| **Selected mode/errors** | Output (to the driver) | CAN |

## Equations of the plant

The plant considered in this model is the so-called *Vehicle Longitudinal Dynamics*.

Considering:



* the vehicle acceleration, expressed in []
* the vehicle longitudinal speed, expressed in []
* the vehicle mass, expressed in []
* the longitudinal force applied to the vehicle center of gravity, expressed in []
* the longitudinal force applied to the wheel on the terrain, expressed in []
* the longitudinal force applied to the vehicle center of gravity due to the frictions with air and terrain, expressed in []
* the moment of inertia of each one of the wheels, expressed in []
* the radius of the wheel, expressed in []
* is the angular speed of the wheel, expressed in [
* is the angular speed of the engine/electrical motor, expressed in [
* is the angular speed of the engine/ electrical motor, expressed in [
* is the angular acceleration of the wheel, expressed in [
* is the frontal surface of the car, expressed in []
* is the automobile drag coefficient
* is the average density of air at sea level in standard conditions à
* is the gearbox reduction ratio
* is the final drive reduction ratio
* is the total power train reduction ratio.

An extremely simplified model can be obtained as follow:

where is the vehicle acceleration, is its mass, is the longitudinal force applied to its center of gravity by the effects of the torque applied on the wheels, and is the sum of the friction forces on the vehicle due to wheel-terrain and vehicle-air interactions.

Considering that the torque is equally split between the two wheels (valid only on straight tracks)

the absence of slipping:

and considering the moment of inertia of the wheels , we can define the following equation, given that à .

The drag force that limits the maximum speed of the vehicle is equal to:

where:

and, as usually modeled:

By substituting the (2) equation in (1), and by integrating both sides, we obtain:

and, by substituting (3) in (6):

Remember that the integrator block of Simulink requires an initial condition corresponding to the vehicle's longitudinal speed at the beginning of the simulation, . A possible configuration of the integration block is shown in Figure 2.

During the model development, put all the needed gain to obtain as an output of the physical model a speed expressed in km/h.

To simulate the slope of the terrain, it is possible to add the gravity force as follows:

With the gravity acceleration on Earth.

Reasonable values for an electric compact car can be:

* The torque T (at the wheel) can vary in the range

Chart, line chart

Description automatically generated

Figure 1 Graph showing drag forces of tires (in orange) and air (in blue) at various speeds. It is possible to observe that, as imposed in equation (5), at 50 km/h. Below this speed, the tire drag is dominant, after that, the air drag is dominant. Moreover, it is possible to see the top speed of the car (around 230 km/h) when , with

With those values, the top speed on level ground reachable by the car is about 230 km/h, where the drag forces equal the traction force (3200 N).

Considering the reverse direction, the maximum speed reachable with a limitation of -60 Nm is about 45 km/h.

Graphical user interface, application, Teams

Description automatically generated

Figure 2 Settings window for the Integrator block of Simulink

Use these values (with a certain tolerance, for example, 10 %) to saturate the integrator block.

To make the model more realistic, it is possible to compute the torque request at the engine/motor. A typical ratio value for transmission of an electric car with a single gear can be around .

All the initialization parameters of the model are automatically loaded model by a callback of the function **init\_fn** as shown in Figure 3.

Graphical user interface, text, application

Description automatically generated

Figure 3 init\_fn callback configuration in the harness model properties.

## Description of the whole system

*Draw the I/O block diagram of the plant and the controller, showing how they interact.*

A white paper with black lines and a black and white square

AI-generated content may be incorrect.

Figure 4: Overall system

The system, represented in Figure 4, is the Simulink model that shows the interaction between the three main blocks:

• **Driver**: tests the system by giving input signals.

• **Controller**: implements the control logic for the one-pedal functionality.

• **Plant**: a model of the Vehicle Longitudinal Dynamic behavior.

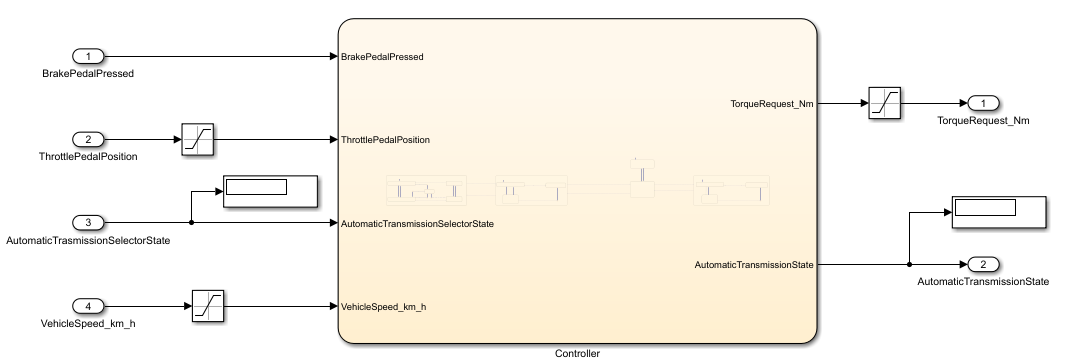


Figure 5: Controller block

The Figure 5 shows more in detail the controller block, it can be noticed that the controller outputs are the Automatic Transmission State and the Torque Request which is given as input to the Plant. The only output of the Plant is Vehicle Speed, which is also fed back to the Controller to be used in the implemented logic.

Some saturation blocks are used to restrict the values of the inputs *ThrottlePedalPosition*, *VehicleSpeed\_km\_h* and the output *TorqueRequest\_Nm*.

In the table below are reported constant values used inside controller block:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Value** | **Unit** | **Data Type** | **Implementation** |
| MAX\_SPEED\_FORWARD | 240 | km/h | Single | Upper limit *VehicleSpeed\_km\_h* |
| MAX\_SPEED\_BACKWARD | 60 | km/h | Single | Lower limit *VehicleSpeed\_km\_h* |
| MAX\_RDB\_ENGAGE\_SPEED | 0.5 | km/h | Single | Stop/Restart substate transitions |
| MAX\_TORQUE | 80 | Nm | Single | Upper limit *TorqueRequest\_Nm*  (Lower limit with minus sign) |
| MAX\_TORQUE\_REVERSE | 40 | Nm | Single | For Acceleration substate in Reverse state |

# Controller SW UnIit specifications

*Provide a brief description of the Controller functionalities and its interfaces.*

The controller is responsible for selecting the correct Transmission State and requesting the correct amount of Torque from the motor. In order to do so the required functionalities are:

* State of the Brake Pedal.
* Position of the throttle pedal, meaning its travel angle.
* State of the Automatic Transmission Selector
* Feedback from the plant of the Vehicle Speed

## Interfaces

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Unit** | **Type[[1]](#footnote-3)** | **Data Type[[2]](#footnote-4)** | **Dimension** | **Min** | **Max** |
| BrakePedalPressed | - | *Input* | Boolean | 1x1 | 0 | 1 |
| ThrottlePedalPosition | - | *Input* | Single | 1x1 | 0 | 1 |
| AutomaticTransmissionSelectorState | - | *Input* | Enum: TransmissionState | 1x1 | Park(0) | Brake(4) |
| VehicleSpeed\_km\_h | Km/h | *Input* | Single | 1x1 | -60 | 240 |
| TorqueRequest\_Nm | Nm | *Output* | Single | 1x1 | -80 | 80 |
| AutomaticTransmissionState | - | *Output* | Enum: TransmissionState | 1x1 | Park(0) | Brake(4) |

The data type ‘Enum: TransmissionState’ refers to a given Matlab function that defines, both numerically and literally, the 5 possible States of the automatic transmission:

* Park(0)
* Reverse(1)
* Neutral(2)
* Drive(3)
* Brake(4)

Here is a concise overview of the variables used in the Controller block:

* **BrakePedalPressed:** is a Boolean signal (pressed or not) used in the controller logic to implement a safety check for the transition between certain States (P,R,D,N).
* **ThrottlePedalPosition:** is the normalized position of the throttle pedal used by the controller to request, depending also on the current State, an amount of Torque to the motor.
* **AutomaticTransmissionSelectorState**: is the Input variable to the controller which specifies the Transmission State selected by the driver.
* **VehicleSpeed\_km\_h**: is taken directly from the Plant Output as feedback, and it is used within the controller for checking the transition between some States and Substates
* **AutomaticTransmissionState**: is an Output variable of the controller, that specifies the current Transmission State to be displayed on the vehicle dashboard.
* **TorqueRequest\_Nm**: is an Output variable of the controller, which is directly fed to the Plant as its only Input.

*Draw the Finite State Machine (FSM) representing the controller logic*

A close-up of a card

AI-generated content may be incorrect.

Figure 6: Controller FSM

* 1. Figure 6 shows the FSM designed to implement the controller. Its states correspond to the ones of the Automatic Transmission Selector with Park being the default state, since at the start the car is always parked.
  2. The *AutomaticTransmissionState* variable is set upon entering each State, and to execute the commands relative to the *TorqueRequest\_Nm* during the permanence in each State.

*Comment on the design choices of the FSM, which are not trivial to be understood just by analyzing the controller logic.*

A computer screen shot of a diagram

AI-generated content may be incorrect.

Figure 7: Brake State and Transitions

* + When the driver selects transmission mode “B” AND, at the same time, the throttle pedal is pressed until the acceleration zone (above one third of its travel angle), the transition from “Drive” to “Brake” is performed.
  + When the driver selects any transmission mode other than “B”, the transition from “Brake” to “Drive” is performed (and potentially it will propagate to other States through “D”).

Analyzing more in detail the “Brake” state, five substates are implemented: “Acceleration”, “RegenerativeBraking”,” Restart”, “EmergencyStop” and “Stop”.

The default transition is assigned to the first one, as when exiting from the “Drive” state, the vehicle is in a state of acceleration. In this substate, the *TorqueRequest\_Nm* is computed through the equations defined in the “Hazard Analysis and Risk Assessment” section.

Once the throttle pedal moves below one third of its travel angle, thus exiting the acceleration region, the regenerative braking equation is applied, a negative *TorqueRequest\_Nm* is provided to the plant.

If the pedal remains in the braking region when the vehicle velocity reaches a selected threshold of 0.5 km/h (MAX\_RDB\_ENGAGE\_SPEED), the transition to the “Stop” substate is performed, which sets the *TorqueRequest\_Nm* to zero, preventing the vehicle from moving in the reverse direction after stopping. Frome here, the “Restart” substate was implemented, in order to match the request described in the “HARA” section, for which the vehicle starts to move slowly when the driver presses the throttle pedal until the neutral point (one third). This is done by providing a low initial torque (*TorqueRequest\_Nm* = MAX\_TORQUE/20) until the *VehicleSpeed\_km\_h* exceeds the threshold value.

The part just described explains the operation of the Brake state without considering the use of mechanical brakes, whereas the following part justifies the implementation of this functionality (*BrakePedalPressed*) through an additional “EmergencyStop” substate.

From any substrate when the brake pedal is depressed (*BrakePedalPressed* == 1) the transition to this substrate is made. The only exception concerns “Restart” where *BrakePedalPressed* causes the transition to “Stop” (considering that 0 <= VehicleSpeed\_km\_h < MAX\_RDB\_ENGAGE\_SPEED). In ‘EmergencySTop’ the torque demand is set to zero, releasing the brake (*BrakePedalPressed* == 0) transitions to the regenerative braking state, from which it is possible to move to the other substates based on *ThrottlePedalPosition* and *VehicleSpeed\_km\_h*. Further analysis, reported at the end of the document, confirmed the validity of this implementation against a more complex one where the transitions out of the emergency stop also directly take into account the *ThrottlePedalPosition*.

A white square with blue and red text

AI-generated content may be incorrect.

Figure 8: Drive State and Transitions

* 1. Figure 8 displays in detail the Drive state and the logic for its transition with the Neutral state:
  + The transition to Neutral is performed when the *AutomaticTransmissionSelectorState* is set to 2 (Neutral), but also when it is set to 0 (Park) or 1 (Reverse), as the latter two states will be reached by passing through Neutral.
  + The transition from Neutral to Drive is performed only when the *VehcleSpeed\_km\_h* is above -5 km/h AND the brake pedal being pressed AND the transmission state being either in Drive or Brake mode.
  1. The Drive state is composed of three substates: “Acceleration”, “Stop” and ”Restart”.
  2. The default state is assigned to “Acceleration”, since it is assumed that, from the Neutral state, the driver selects this mode to start moving the vehicle. The “Acceleration” substate requests for a torque proportional to the ThrottlePedalPosition:
  3. If the brake pedal is pressed “Stop” is entered, so that no positive torque is requested, and the vehicle stops moving.
  4. Lastly, the “Restart” substate was implemented, in order to match the request described in the “HARA” section (in the same way as for the Brake state) for which the vehicle starts to move slowly when the brake pedal is released.
  5. Once the driver releases the brakes AND presses the throttle pedal, the system goes back to the “Acceleration” substate.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 9: Reverse State and Transitions

* 1. Figure 9 displays in detail the Reverse state and the logic for its transition with the Neutral state:
  + The transition to Neutral is performed when the *AutomaticTransmissionSelectorState* is set different to 1 (Reverse).
  + The transition from Neutral to Reverse is performed only when the *VehcleSpeed\_km\_h* is less than 5 km/h AND the brake pedal being pressed AND the transmission state being in Reverse Mode.

The logic for the Reverse state is exactly the same as for Drive with the only difference that the *TorqueRequest\_Nm* is computed prortionally to the MAX\_TORQUE\_REVERSE (instead of MAX\_TORQUE):

A screenshot of a computer

AI-generated content may be incorrect.

Figure 10: Park and Neutral State and Transitions

* 1. Figure 10 displays the remaining two States and the transition logic between them:
  + The transition from Neutral to Park occurs when the *VehicleSpeed\_km\_h* is below 5 km/h in absolute value, AND the brake pedal pressed AND the transmission state set to 0 (Park).
  + The transition from Park to Neutral occurs when the brake pedal is pressed AND the transmission is set to any state other than 0 (Park).
  1. Both Neutral and Park states reduce the torque request to zero, as Neutral is used as a transition state between vehicle acceleration and reverse motion, and Park is assumed to be used when the vehicle is stopped.

*Comment with plots of the results obtained from the test cases (it is suggested to use the Data Inspector)*

A graph with lines and numbers

AI-generated content may be incorrect.

Figure 11: Simulation Results

A graph of a train

AI-generated content may be incorrect.

Figure 12: Sparklines Results

The figures above show the plots, obtained with the Simulink Data Inspector tool, of the evolution in time of all the data variables of the system. In Figure 11 all the variables are plotted together while in Figure 12 the several plots are grouped depending on their y-axis. From them it is possible to assess the validity of the state machine.

* The vehicle starts in the Park state with the brake pedal pressed and null speed.
* The driver then switches to Brake mode, the controller computes positive torque and the vehicle speed starts increasing. We can notice how the *AutomaticTransmissionState* variable, given as output by the controller, slightly lags behind the automatic selector sate when transitioning, stopping briefly at Drive until the throttle pedal position is in the acceleration region.
* The torque request is correctly saturated to the value of 80 Nm when the throttle pedal is fully pressed (until the target speed of 90km/h is reached), and to the value of - 80Nm when the throttle pedal is fully released in the regenerative braking state.
* Close to 48 seconds into the simulation, the velocity of the vehicle reaches 0 km/h, while the throttle pedal is still fully released in Brake mode, and the controller correctly transitions to the “Stop” substate and stops the torque request, preventing the vehicle from moving in the reverse direction.
* The simulation ends with a switch to the “Reverse” state 80 seconds in, with the reverse torque request saturating to -40 Nm to reach the target speed of -19.3 km/h.

The plots below, generated with the Simulink Data Inspector, track the time evolution of every system variable. Figure 11 overlays all variables, while Figure 12 groups them by their y-axis. From these plots, it is possible to verify the state machine’s behavior:

* Initially, the vehicle is in the Park state: the brake pedal is pressed, and speed is zero.
* The driver shifts into Brake mode, prompting the controller to compute positive torque and the vehicle speed to rise. Notice that the controller’s *AutomaticTransmissionState* output slightly lags the automatic selector state during transitions, pausing briefly at Drive until the throttle pedal enters the acceleration region.
* When the throttle pedal is fully pressed, the torque request correctly saturates at 80 Nm (until the vehicle reaches 90 km/h). Conversely, in regenerative braking when the pedal is fully released, it saturates at - 80 Nm.
* Around 48 s into the simulation, the vehicle slows to 0 km/h while still in Brake mode with the pedal fully released. The controller then transitions to the Stop substate and zeroes the torque request, preventing any reverse motion.
* Finally, at 80 s, the simulation switches to the Reverse state, where the reverse torque request caps at - 40 Nm to achieve the target speed of - 20 km/h.

As a conclusion, from the simulation results it is possible to state that the system behaves as expected and respects its input requirements.

*Further analysis on the Brake state*

As noted above, to confirm the validity of the Brake-state model, further analyses were conducted comparing it with the more detailed version shown in Figure 13.

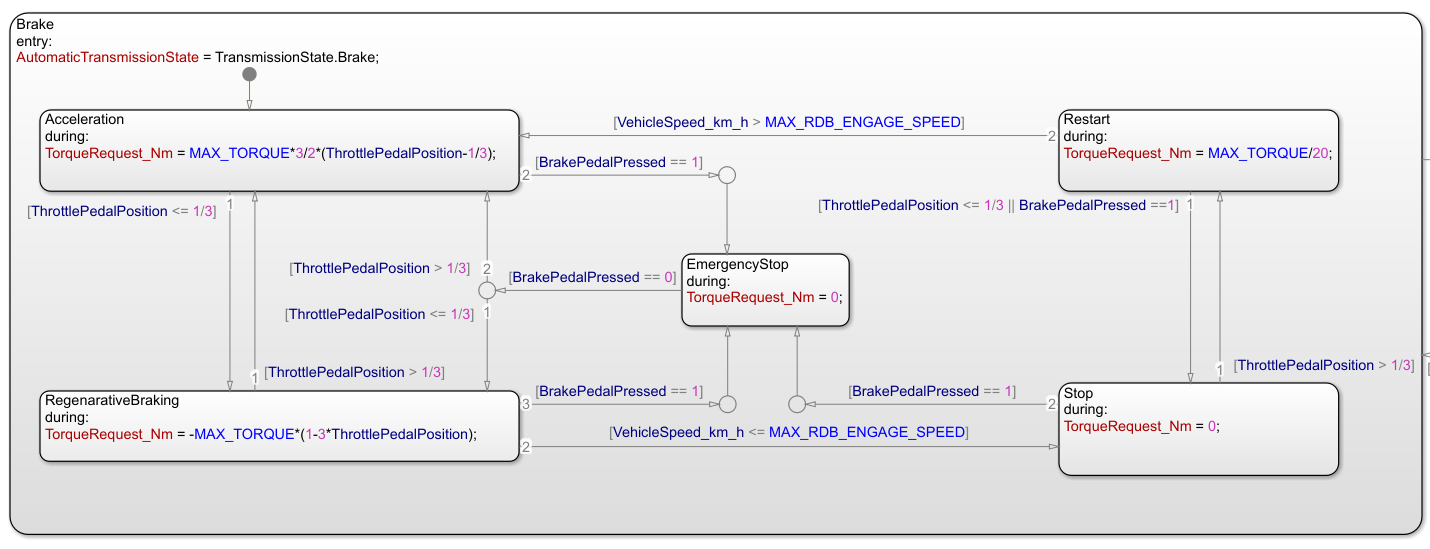


Figure 13: New Brake State

The driver was modified so that, upon brake-pedal release, the throttle pedal position is emphasized: following the first braking the throttle remains in the braking region, and following the second it moves into the acceleration region.

The two controllers are expected to differ in behavior. In the implemented version, after the “EmergencyStop” the output always transitions to “RegenerativeBraking”, that’s the substate in which accelerator-pedal position is controlled. In contrast, in the more detailed model the transition after “EmergencyStop”, to “Acceleration” or “RegenerativeBraking”, is determined directly by the *ThrottlePedalPosition*.

A graph of a graph

AI-generated content may be incorrect.

A graph of a graph

AI-generated content may be incorrect.

Figure 14: Results: (Left) Implemented Version; (Right) More Complex Implementation

Figure 14 presents the results for both Brake state versions to compare their performance. As no discrepancies are observed, it is confirmed that the more complex state offers no improvement in output behavior.

1. Input, Output, Local, Global, Volatile [↑](#footnote-ref-3)
2. Struct, Double, Integer, Enum, Boolean, etc… [↑](#footnote-ref-4)