KUNGLIGA TEKNISKA HÖGSKOLAN

MSc Vehicle Engineering

A.Y. 2024/25

Torque Vectoring control system

Carlo Vittorio Colucci:
carlovittorio.colucci@kthformulastudent.se

May 06, 2025

Projects introduction

Formula Student was created in the United States of America (USA) in 1981. In the beginning, it was a small competition in terms of its participation, but it included 6 teams from all over the states and judges from all over the American automotive industry. Now, competitions all over the world are organized. The competition consists of building a car to compete in four dynamic events, Autocross Acceleration, Skid pad, Endurance and Efficiency. In static events, the students must prove their understanding and decision-making of the design and manufacturing process, they must present a whole business plan based on the race car. Now official events are hosted in 5 out of 7 continents and they comprehend Electric Vehicle (EV) and Driverless Vehicle (DV) categories while the Combustion Vehicle (CV) category will be removed soon [4].

Contents

1 Project Development Process

The team tried to follow the good engineering practice in terms of the project development process. Regarding the control group and who is going to continue also in Period 2 tries to follow the workflow below.

Period 1:

- problem description
- benchmarking, analysis of the current solution
- literature research
- collecting requirements
- concepts
- control theory, equations
- setting up simple simulation environment(s)
- choosing a concept to move on with

Period 2:

- implementation
- simple model testing
- testing in more complex simulation
- possibly testing on the FS car
- debugging, improving
- comparison of the simulations, current solution and new method
- finishing the documentation, conclusion, further proposals

These bullet points more or less cover the general V-model development process. Period 1 aims to deal with the left and Period 2 with the right stem of the letter "V". However, in the end, we might not have the opportunity to test our solutions on the FS car, the final step would be the verification and validation between the simulation and real test data.

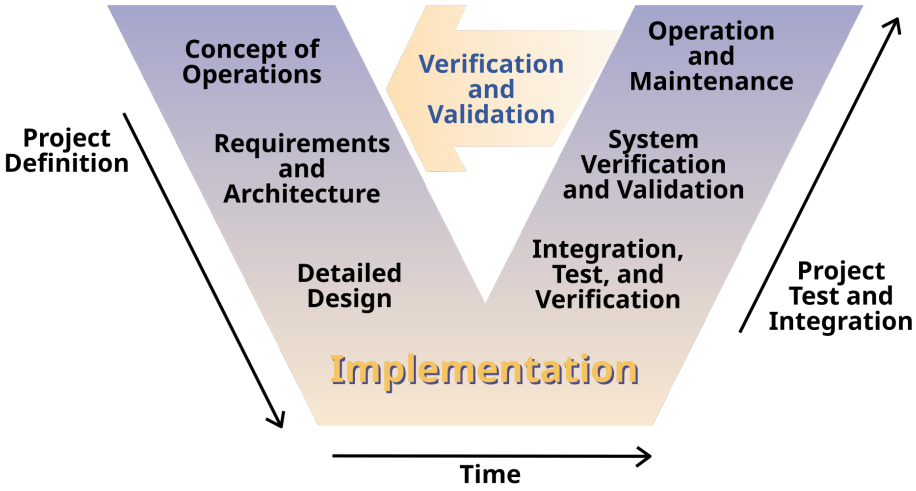

Figure 1: General V-model development process (source: Wikipedia)

2 Physical assembly behind the control system

The most recent race car of the KTH Formula Student team has a four-wheel-drive system. The control software that can be implemented on the car has been designed in Simulink and then imported into the dSpace framework. This will then be automatically converted into C-code and subsequently uploaded into the dSpace box in the car. The control software runs with a frequency of 200 Hz. However, the inertial measurement unit (IMU) measures the acceleration and yaw rate with 50 Hz, therefore the torque vectoring algorithm works in that frequency. The wheel speed is measured with 200 Hz, thus the traction control logic intervenes the motor controller with that rate. The applied motors are permanent magnet synchronous motors (PMSM) manufactured by the AMK company. On the front hoop, there is a GNSS (Global Navigation Satellite System) mounted which sends velocity data for the control system. It can be utilized e.g. in the state estimation and traction control.
For what concerns the vehicle parameters, the most relevant ones are collected in the table below:

Wheelbase	l	$1.535m$
Weight distribution (front)	w_d	46%
Center of gravity height	h_G	$0.28m$
Track width	t	$1.2m$
Tire radius	R	$0.23m$
Tire loaded radius	R_l	$0.22m$
Vehicle mass	M	$250kg$
Vertical polar moment of inertia	J_z	$115.4kg \cdot m^2$
Drag coefficient	C_x	1.5
Lift coefficient	C_z	4
Front section	S	$1.16m^2$
Transmission ratio	τ_{GB}	1/14

Table 1: Vehicle parameters

Instead, the tire data considered are referred to the Hoosier 18x7.5-10 R25B RW8 tire set. After a filtering of the raw data coming from testing, a final result of the lateral force as a function of the side slip angle follows:

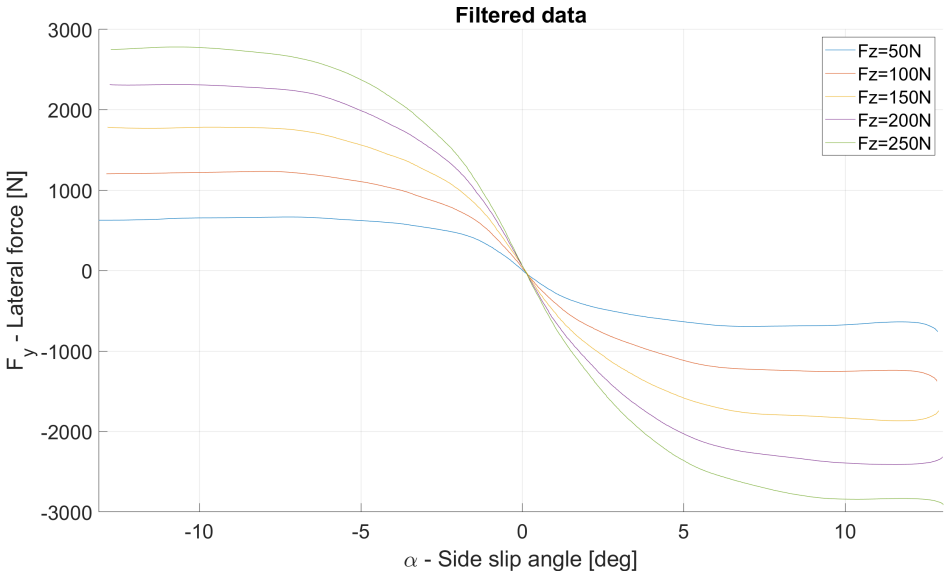

Figure 2: Lateral force filtered data

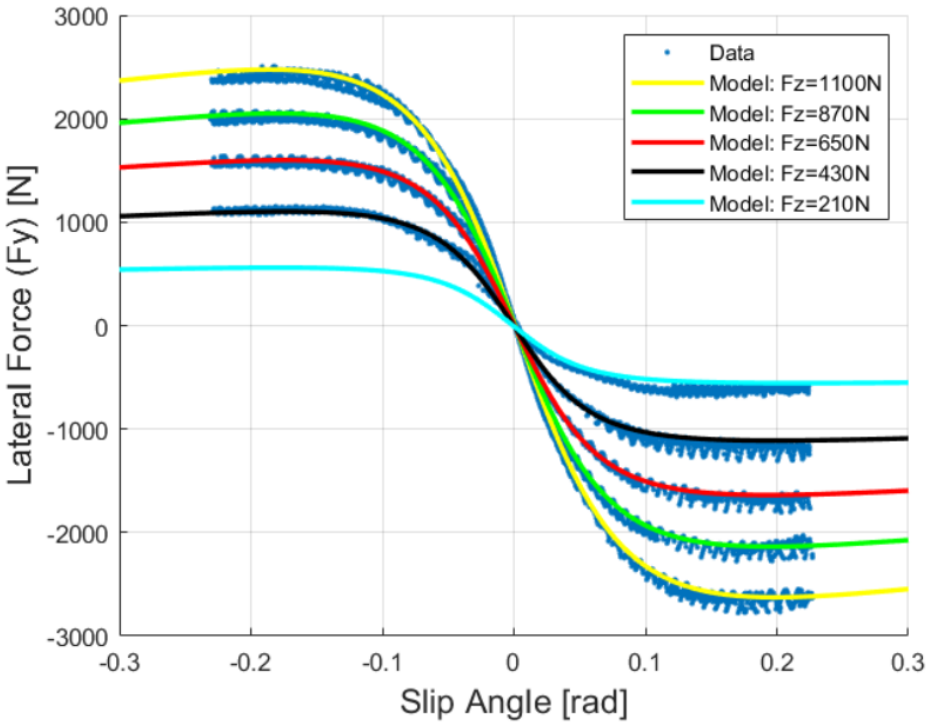

Figure 3: Fitted Magic Formula tire model in MATLAB

3 Related competition rules

Regarding the vehicle controls there are not many limitations in the rulebook:

1. The TS power at the outlet of the TSAC (Tractive System Accumulator Container) must not exceed 80kW.
2. Regenerating energy is allowed and unrestricted.
3. Wheels must not be spun in reverse.
4. A fully released accelerator pedal in manual mode must result in a wheel torque of $\leq 0Nm$.

4 Torque vectoring control

Torque vectoring is a system used in vehicles to improve handling, stability and performance by controlling the torque allocation for each wheel, depending on the driving conditions.
This control system is an extension of the mechanical differential concept for all the vehicle 4 wheels, exploiting the FS vehicle property of having 4 in-wheel motors.

To analyse this control system, it is useful to consider it as split into two main parts:

- **Upper level control (ULC):** a specific controller (i.e. PID, LQR, etc.) is used to assess the vehicle amount of rotation. The input provided to the controller can be a current vehicle state or vehicle control input;
- **Lower level control (LLC):** control algorithm allocating the torques provided by the EMs on each wheel, to deliver the target rotation.

4.1 Current control system

The PID controller is widespread in the industrial field in general. It is mainly effective for simple linear systems but, with proper adaptations, it can be used for more complex applications as well. The ULC controller used for the DeV18 (FS 2023 vehicle) is an example. Concerning the LLC, a simple logic based on a direct allocation of the ULC output (not conceived to be a yaw moment) was adopted.
This embedded system was not designed to be simulated, and its overall architecture could be improved: starting from the reference generation logic, going on with the torques allocation. Starting from scratch is the quickest and most effective way undertaken.

4.2 Control system architecture

In this Section the idea is to introduce a possible control system architecture to be used as a starting point to develop different logic of ULCs and LLCs.

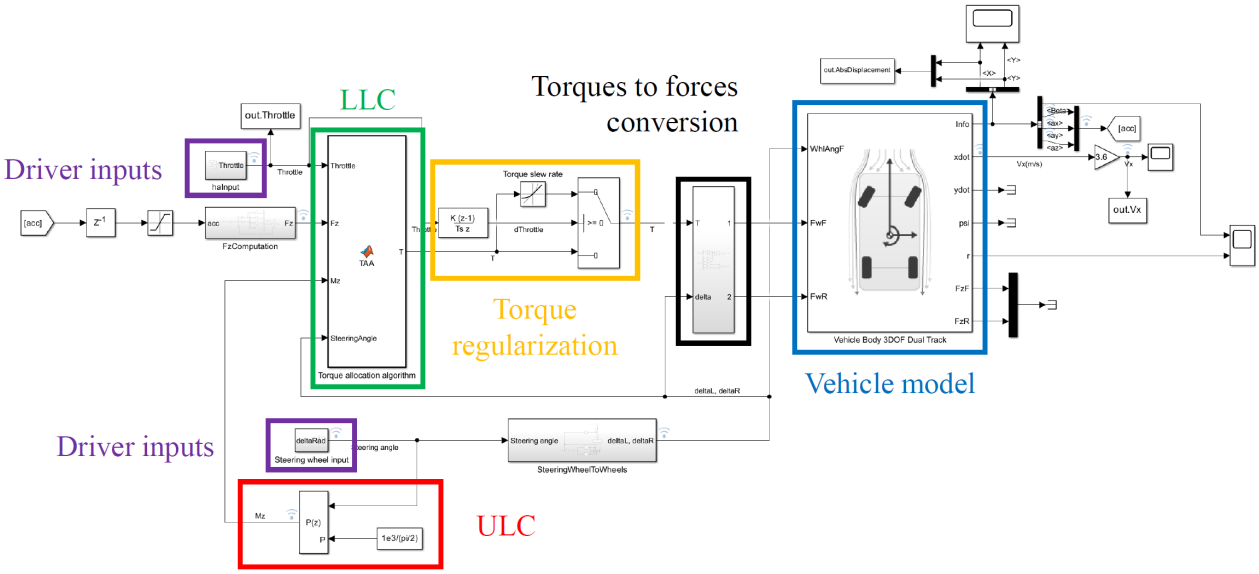

Figure 4: TV architecture

An explanation of the control flow follows:

1. *Driver inputs*

The driver inputs (throttle and steering wheel angle) are provided to the ULC and to the vehicle model. The driving scenario is introduced in Section 4.3.

2. *From steering wheel angle to tire steering angles*

The steering wheel angle is used to evaluate the steering angles at the tire level: the effective ones used by the LLC and the vehicle model. The conversion happens in two main steps:

- Steering wheel angle to rack travel by means of a gain;
- Rack travel to steering angles at tire level. Two separate LUTs (look-up-table) are used to account for the Ackerman angle. The LUT values are retrieved by the vehicle CAD model.

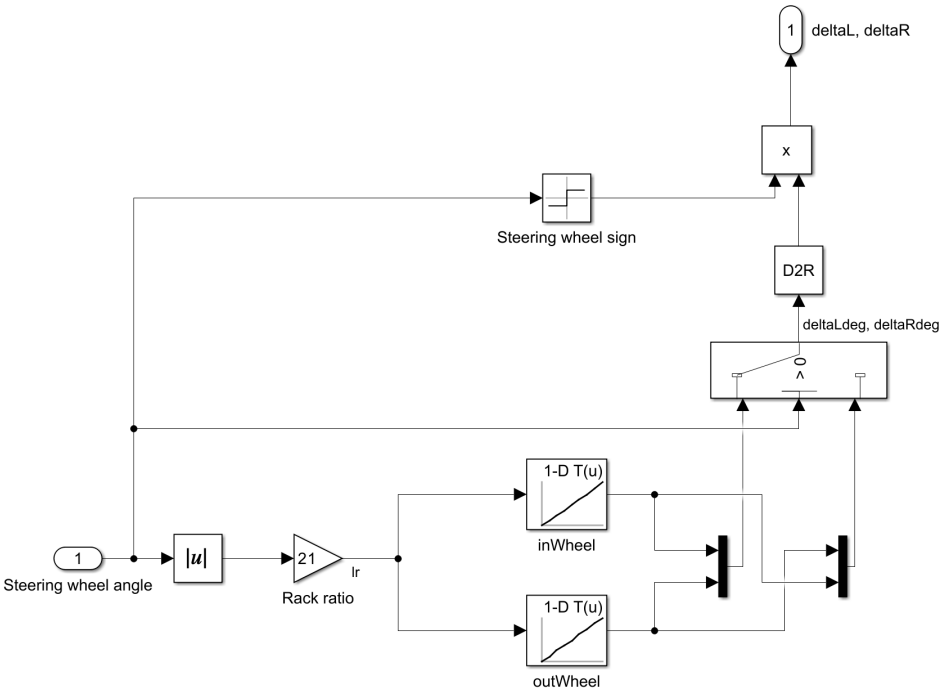

Figure 5: Steering wheel angle to tire steering angles

3. From accelerations to vertical loads

The accelerations on the three directions (a_x, a_y, a_z) result in the correspondent inertial forces:

$$F_x = M \cdot a_x; \quad F_y = M \cdot a_y; \quad F_z = M \cdot a_z \tag{1}$$

where M is the vehicle mass.

These forces are balanced by vertical forces on the four wheels. To compute them, an equilibrium of forces and moments is undertaken according to the graphic representation shown below:

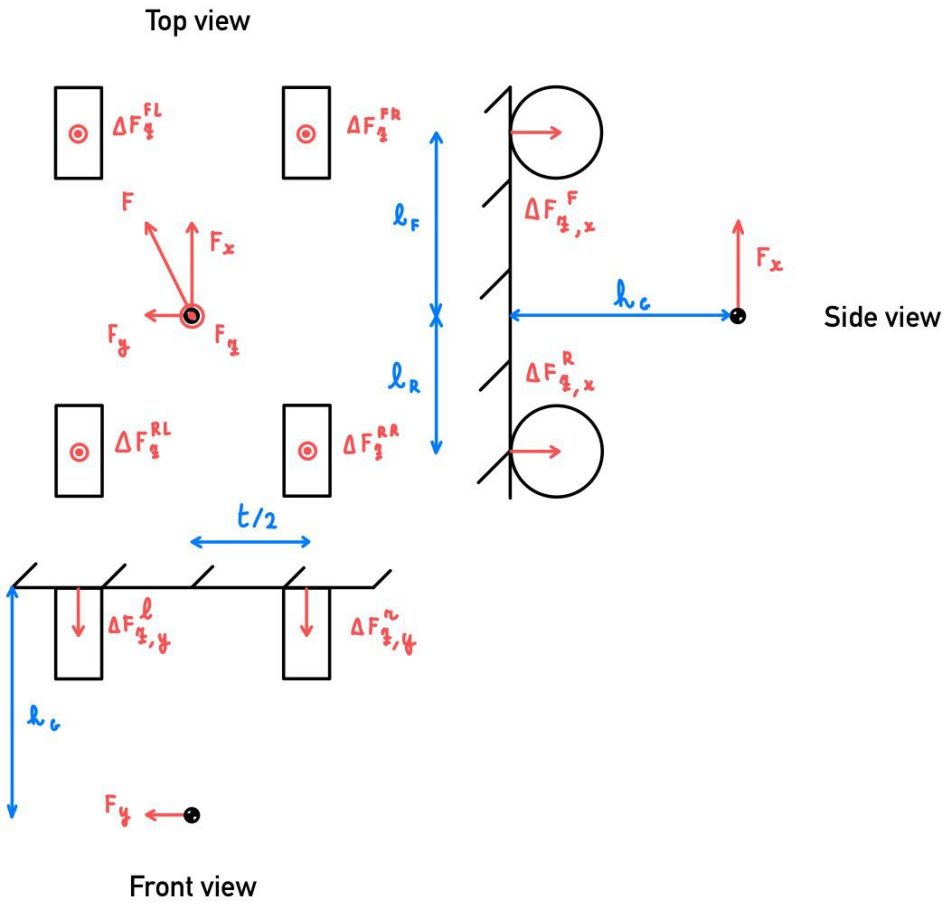

Figure 6: Moments balance

For the vertical inertial force, it is enough to equally split it on the four wheels.
The computed vertical forces are dynamic variations with respect to the static force of gravity, therefore they are denoted with the Δ symbol.
The results from the equilibrium are categorized into forces on the two longitudinal axles (front and rear) and forces on the two sides (left and right):

$$\Delta F^F_{z,x} = F_x \cdot \frac{h_g}{l};\ \Delta F^R_{z,x} = -\Delta F^F_{z,x};\ \Delta F^l_{z,y} = F_y \cdot \frac{h_g}{t};\ \Delta F^r_{z,y} = -\Delta F^L_{z,y} \tag{2}$$

The dynamic variation of vertical forces follows:

$$\{\Delta F_z\} = \frac{1}{2} \cdot [\Delta F^F_{z,x} + \Delta F^l_{z,y};\ \Delta F^F_{z,x} + \Delta F^r_{z,y}; \Delta F^R_{z,x} + \Delta F^L_{z,y}; \Delta F^R_{z,x} + \Delta F^R_{z,y}] - \frac{F_z}{4} \tag{3}$$

where F_z is the inertial force introduced in Equation 1.
The static forces are:

$$\{F_{z,static}\} = \frac{M \cdot g}{2} \cdot [w_d;\ w_d;\ (1 - w_d);\ (1 - w_d)] \tag{4}$$

where w_d is the weight distribution between front and rear, defined as:

$$w_d = \frac{M_F}{M} = 46\%$$

The on-line values of vertical forces are given by the summation of the static values with the dynamic variations. Combining Equations 2, 3 and 4, the result is:

$$\{F_z\} = \{F_{z,static}\} + \{\Delta F_z\} \tag{5}$$

The Simulink implementation of Equations 2, 3, 4 and 5 is shown below:

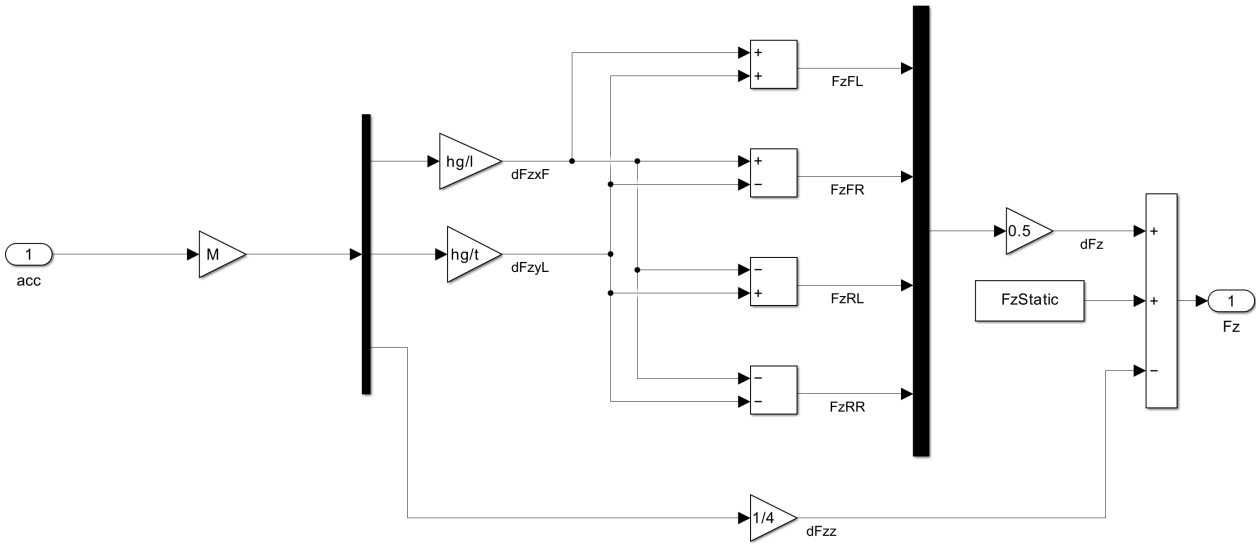

Figure 7: Vertical forces computation

4. Upper Level Control

The ULC controls the magnitude of rotation of the vehicle. Its output corresponds to the vehicle yaw moment (Mz).

The choice of the input to be provided to the ULC strongly affects the vehicle response and handling. The kind of controller implemented determines the smoothness of the control as well. Since all these topics are crucial for the TV development, they are investigated later on in Section 4.4.

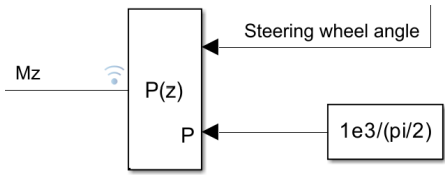

Figure 8: Upper Level Control example

5. Lower Level Control

The LLC is an algorithm dealing with the actuation of the target yaw moment. Its name is Torque Allocation Algorithm (TAA). A lot of different logic can be developed: as an example, in Section 4.5 two different logics are introduced, with different levels of complexity.

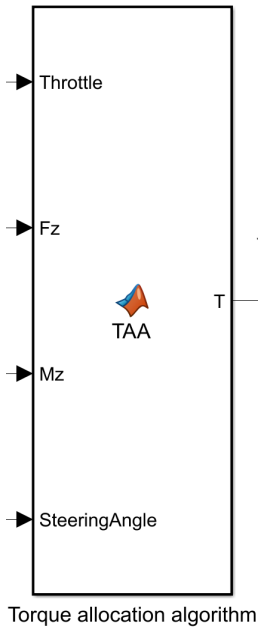

Figure 9: Lower Level Control

6. Torque smoothing

The torque smoothing avoids abrupt torque changes commanded by the control. This strategy aims to improve the comfort for the vehicle's occupants.
For safety reasons, this condition is imposed only when the driver accelerates: when the throttle pedal is released, the vehicle must instantaneously ensure the desired deceleration, without limiting it. The torque slew rate is set to $8N \cdot m/s$. This value is chosen on the basis of the current vehicle simulation and vehicle model, but it probably needs a more accurate tuning during a track testing session.

7. From torques to forces

The torques are converted into longitudinal forces to feed the vehicle model. The equation behind the conversion follows:

$$\{F_x\} = \frac{1}{\tau_{GB} \cdot R_l} \cdot \{T\} \cdot [cos(\delta_L);\ cos(\delta_R);\ 1;\ 1] \tag{6}$$

where the τ_{GB} and R_l values are introduced in Section 2, and the products in the equation are scalar products. The implementation within Simulink is:

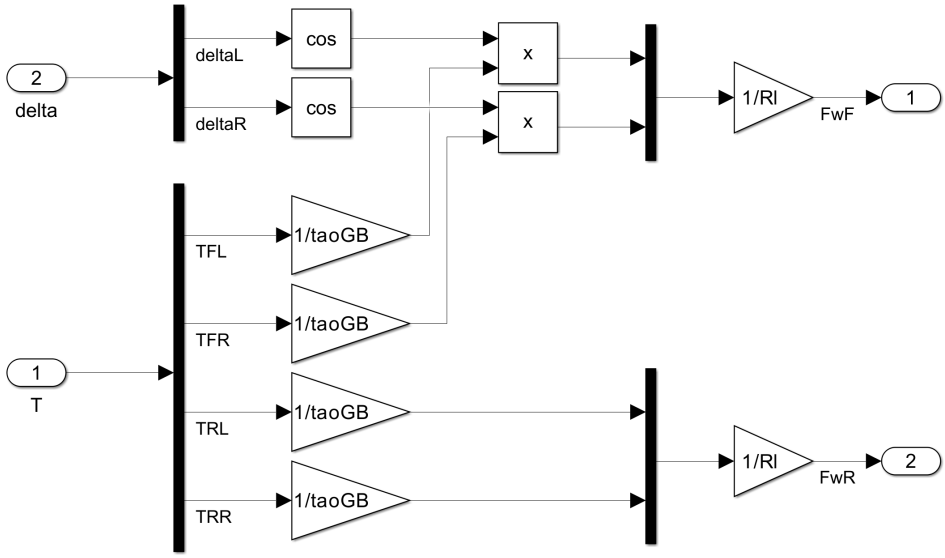

Figure 10: Torques to forces conversion

8. Vehicle model simulation

All vehicle inputs (steering angles, longitudinal forces) are provided to the vehicle model, that is used to simulate the real vehicle behaviour. Its outputs are the vehicle states that, on a normal vehicle, are usually sensed or observed. For a more deep explanation of the adopted vehicle model, look at Section 4.6.

4.3 Driving scenario

The goal of the simulation is to analyze how a vehicle with TV control behaves, with respect to a vehicle without controls. The idea is to provide aggressive driver inputs that can be challenging for the vehicle handling.
Regarding the throttle pedal, only an acceleration is considered, since that's the current TV control development condition. The pedal trend is a steep ramp until 3 seconds, and then it stabilizes at a high throttle value (80%):

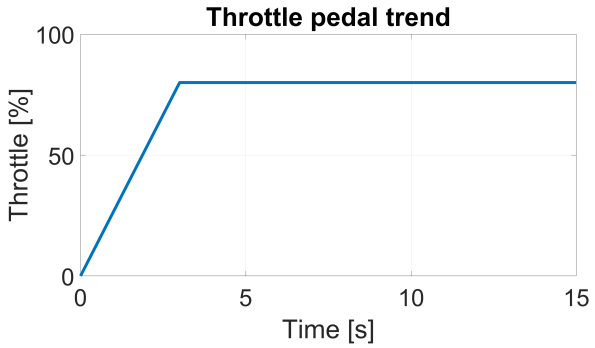

Figure 11: Throttle pedal trend

For what concerns the steering wheel input, two aggressive changes of direction are recreated: the driver steers to the left and then to the right, with a final steering wheel re-centering. Almost the maximum steering wheel angle available is exploited (60°):

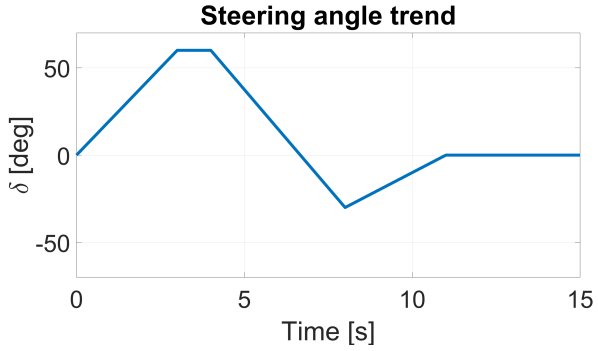

Figure 12: Steering wheel angle trend

4.4 Upper Level Control

As anticipated in Section 4.2, the choice of the input for the ULC strongly affects the vehicle response and handling. Two concepts are presented in the next two Sections: one very simple to implement and to tune, and another one more complex but more focused on the vehicle handling optimization.
Another important aspect in the ULC is the choice of the controller to use. In Section 4.4.3 different solutions are introduced and investigated, and the final choice is discussed.

4.4.1 Steering wheel angle dependency

The steering wheel angle is directly related to the vehicle rotation: the more the driver steers, the more he wants the vehicle to rotate while cornering.

As explained in Section 4.2, M_z is an equivalent control input for the vehicle, imposing its magnitude of rotation. Thus, it is reasonable to directly link it to the steering wheel angle. It is straightforward to introduce a simple P controller to undertake this job (Fig. 8). At the moment, a fixed value is selected. The idea is to provide $1000N \cdot m$ when the driver uses almost the maximum steering wheel angle available (90°). This yaw moment value is based on the analysis of past track testing sessions, where the vehicle presented a yaw acceleration of $4rad/s^2$. Considering the vehicle polar moment of inertia $J_z = 115.4kg \cdot m^2$ (Section 2), this acceleration leads to an inertial moment $M_i = J_z \cdot \alpha \approx 460N \cdot m$. The target is to double this moment, ending up to the previously mentioned value.
This gain value is just provisional; it is necessary to tune it by means of track testing sessions. The tuning will be operated by means of a knob on the steering wheel that the driver can control, to decide the intensity of the TV and, for safety reasons, to disable it.

A more interesting idea is to make this controller active and depending on the speed. It is reasonable to suppose that the control has to be more aggressive at low speeds, but more quiet at high speeds, for safety reasons. The P gain would be retrieved consulting a look-up table (LUT) previously built offline. It is possible to define the LUT values by means of an optimization process, where different P values are tested for each speed level, and the one leading to optimal handling results is selected. To see what performance parameters can be involved in the optimization process, refer to Section 4.7.1.
An illustration of how the result can be implemented in Simulink is shown below:

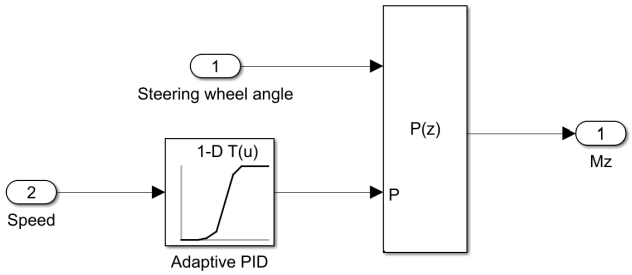

Figure 13: Adaptive PID example

To conclude, this control strategy is very simple to implement and independent from vehicle state measurements, making it reliable. The tuning phase is crucial to reach a satisfactory vehicle behaviour for the driver. This ULC strategy is used as a baseline for the TV track testing session.

4.4.2 Neutral steering vehicle target

An alternative ULC logic is to make the vehicle neutral steering. This condition leads the vehicle to undertake the minimum cornering radius possible, optimizing its handling.
This target is reached by defining a reference for specific vehicle states. This reference is then used to compute an error, with respect to the current states, to be minimized .
Useful assumptions for the reference characterization are taken:

- The vehicle is a rigid body moving on a flat ground (roll, pitch and heave are neglected);
- The vehicle is in steady-state conditions;
- Secondary assumptions, such as no rolling resistance or lift force.

The choice of the states to control is driven by the vehicle degrees of freedom related to the handling:

1. ψ: yaw;
2. β: vehicle side slip angle;
3. ϕ: roll;
4. ϕ_i: roll on the two axles (front and rear).

For ease of working, the ULC deals only with yaw rate and vehicle side slip angle.
Their references are computed starting from the vehicle equations of motion.
Arranging these equations it is possible to end up with the following set of gains:

- Curvature gain $\frac{1/R}{\delta}$;
- Vehicle side slip angle gain $\frac{\beta}{\delta}$;
- Lateral acceleration gain $\frac{a_y}{\delta}$.

The lateral acceleration gain is directly related to the yaw rate. Indeed it is used, coupled with the side slip angle gain, to compute the two references. Their equations follow:

$$\beta_{REF} = \frac{l_r}{l} \cdot \frac{1 - \frac{M \cdot l_f \cdot V_x^2}{l_r \cdot l \cdot C_r}}{1 + \frac{K_{US} \cdot V_x^2}{g \cdot l}} \cdot \delta \tag{7}$$

$$\omega_{\psi,REF} = \frac{V_x / l}{1 + \frac{K_{US} \cdot V_x^2}{g \cdot l}} \cdot \delta \tag{8}$$

where:

- l_f, l_r : CoG distances from the front and rear axles;
- l : wheelbase;
- M : vehicle mass;
- V_x : longitudinal speed;
- C_f, C_r : front and rear axles cornering stiffness. It is important to stress that these two values are calculated on-line consulting the LUT based on the available tires data, but the vertical force is considered constant as an assumption, so no related tire saturation effects are involved;
- K_{US} : under steering coefficient. This parameter is crucial to determine and to control the vehicle behaviour. It's a Reference Generator block input, since it can be quickly adjusted to fit the driver requirements. As anticipated before, an optimal control strategy is to set this parameter equal to 0, making the vehicle neutral steering;
- g : gravity acceleration;
- δ : steering angle.

From a preliminary implementation and analysis of this ULC, it has been observed that, also in this case, two simple P controllers to minimize the state errors are an effective solution. Their values must always be tuned by track testing sessions, but from simulation, the two following P values looked like a good trade-off:

$$P_\beta = 3000, \;\; P_\psi = 1000$$

An example of how this ULC logic can be implemented in Simulink is shown below:

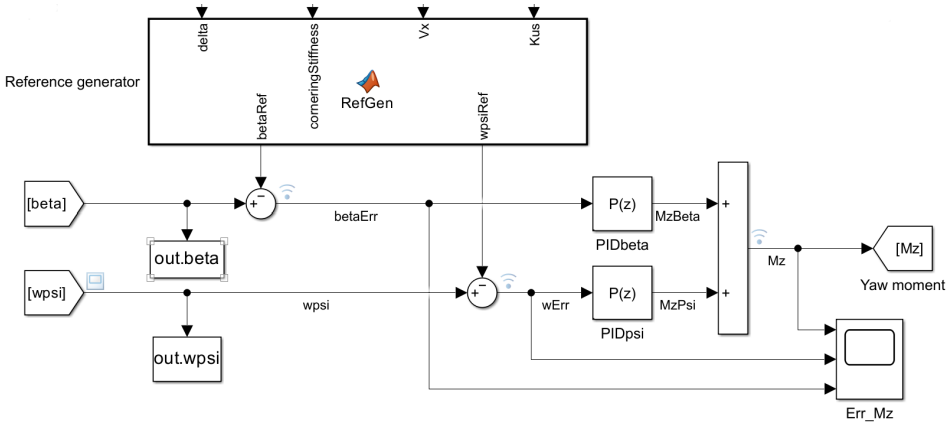

Figure 14: Neutral steering vehicle architecture

This control logic follows a more accurate criterion with respect to the one introduced in the previous Section, leading to better results, but it is more tricky to implement.
For instance, it requires accurate vehicle data, such as the tire model to evaluate the current cornering stuffiness.
Moreover, it is strongly influenced by the measurement of the current vehicle states: an effective control is based on reliable measurements. The yaw rate is sensed very accurately from the IMU (Inertial Measurement Unit), while the vehicle side slip angle is more difficult to be sensed. Currently, its measurement is obtained by coupling the IMU data to the GPS ones, but it is possible that it is not accurate enough. This is the reason why it must be validated and possibly corrected, by internally developing a Kalman filter. This necessity leads to the adoption on the FS vehicle of a non-contact optical sensor, very accurate for this kind of measurement. Its name is Kistler Correvit S-CE, the explanation of how this sensor is implemented and an introduction to preliminary track test results is shown in Section 4.8.

4.4.3 ULC controller

After a benchmarking analysis, it is observed that alternative controllers suitable for the ULC are the following:

- LQR (linear quadratic regulator) controller;
- Fuzzy logic;
- MPC (model predictive control).

The LQR is a controller based on a cost function minimization, aiming to retrieve the optimal control input leading to the states energy minimization. This controller is suitable for linear systems, but in the cases of more complex non-linear systems like the FS vehicle model, it requires the implementation of a model linearization process.
About the fuzzy logic, this kind of rule-based controller is very effective and quite easy to consult since it is based on simple conditions linguistically expressed to which weight function (called membership functions) are assigned. This controller was explored in a previous work [1], thus it is not deeply discussed in this project report.
Finally, the MPC working principle is similar to the LQR one, with the difference that:

- It can directly work with non-linear systems;
- The objective function is dependent from the model tracking error;

- The minimization works not only on the current states, but it is extended to their prediction over a given time horizon.

The MPC controller has been the most investigated one, since it looked leading to promising results, but the trade-off between complexity of implementation and obtained advantages is not worth it.
Quite similar conclusions are drawn for the other controllers. So, eventually, the chosen controller for the ULC is the PID. As anticipated before, for both the different ULC logics, it is enough to have only the proportional term, since the derivative and the integral ones lead to a worsening of the results and to a higher complexity during the tuning phase.

4.5 Lower Level Control

As for the ULC, two different algorithms are developed for the LLC: one is quite basic, but not computationally demanding, the other one allows for a more flexible torque limits handling and for a more clever allocation principle.

4.5.1 Basic TAA

The TAA is the intermediary between controlled yaw moment and vehicle model: the yaw moment must be converted into forces to be subtracted/added to the ones requested by the driver. A simple logic is introduced in this algorithm: the forces, generating a yaw moment equivalent to the controlled one, are evenly split between left and right side. As an approximation, they are all directed along the vehicle longitudinal direction. In reality, this is not true since, depending on the current steering angle, the forces on the front tires are made up of two components (one longitudinal, the other lateral), contributing in a more complex way to the yaw moment generation.
A possible improvement consists in considering the proportionality between the forces allocation and the current steering angle. Furthermore, it would be very interesting to implement the traction control within this algorithm, in order to assign force to each tire depending on its current longitudinal slip and, so, on its available grip level.
The biggest challenge to face with the design of this algorithm is to deal with tricky situations arising when the upper and lower torque limits are reached.
First of all these torques capabilities and the torque request are commuted into forces at ground level, since it is more easy to deal with forces within the whole algorithm; then, the mid value between the two limits is computed and compared with the current force demand. Two conditions can verify:

- the force demanded by the driver is over the force mid value: the variation of forces on the two sides is equally distributed and the current force demand is closer to its upper limit, thus the upper torque limit will be reached before, in case of saturation.
  In the saturation case it is still possible to recover the yaw moment gap, not covered by the saturated side, by diminishing the force on the other side, until the desired yaw moment is guaranteed or the lower limit is reached. Of course the total force deployed is lower, but the vehicle handling is still improved;

- The force demanded by the driver is below the mid value: in this case the lower torque limit is closer and reached earlier. In the case of saturation, it is only possible to add on the other side an amount of force equal to the one subtracted to the saturated side. This is done in order to avoid increasing the torque delivered above the driver request, that would also violate what stated in the FS regulation (Section 3).

The code implementation of this logic is reported here:

```
FxMaxSide = trqUpperLimit * 2 / taoGB / Rl; % [N]
FxMinSide = trqLowerLimit * 2 / taoGB / Rl; % [N]
FxMidSide = (FxMinSide+FxMaxSide)/2; % [N]

trqDemand = EMtrqAvailable * 4 * pedals; % [N*m] total maximum
   torque demanded by the driver
trqWheels = trqDemand / taoGB;
Fx0 = trqWheels/ Rl; % [N] total maximum force demanded by the driver

FxSide0 = [Fx0/2, Fx0/2]; % [L R]

dFx = Mz/t; % [N]
dFxR = dFx;
dFxL = dFx;

FxSideProv = [FxSide0(1)-dFxL, FxSide0(2)+dFxR]; % [L R]

FxSide = FxSideProv;
side = [1, 2]; % [L, R]
% The TV operates mirrored for yaw moments of opposite sign:
if Mz < 0
    side = fliplr(side); % [L, R]
```

```
22 end
23
24 if Fx0/2 >= FxMidSide & FxSideProv(side(2)) >= FxMaxSide
25     % Upper saturation reached
26     FxSide(side(1)) = max(FxMinSide, FxSideProv(side(1)) -
          (FxSideProv(side(2))-FxMaxSide)); % Left side (for Mz>0)
27     FxSide(side(2)) = FxMaxSide; % Right side (for Mz>0)
28 elseif Fx0/2 < FxMidSide & FxSideProv(side(1)) < FxMinSide
29     % Lower saturation reached
30     FxSide(side(1)) = FxMinSide; % Left side (for Mz>0)
31     FxSide(side(2)) = min(FxMaxSide, FxSide0(side(2)) +
          (FxSide0(side(1))-FxMinSide)); % Right side (for Mz>0)
32 end
33 % The torque on the two axles is simmetrycally distributed in a first
34 % moment:
35 FxProv = [FxSide(1)/2, FxSide(2)/2, FxSide(1)/2, FxSide(2)/2]; % [N]
      [FL, FR, RL, RR]
```

The single factor affecting the front-rear torque distribution is the regulation of the front-rear torque knob, that is a potentiometer positioned on the steering wheel that can be tuned by the driver, to decide the torque distribution: going from rear wheels drive to front wheels drive, with the standard setting of all wheels drive. This kind of implementation is useful not only to emulate different typologies of vehicle just by means of a control, but also to make the whole system fault tolerant: when an inverter doesn't work properly, for instance, the entire axle can be disabled and setting this parameter to FD or RD (depending on the case), so that it will still be possible to drive the vehicle ensuring TV capabilities.

Figure 15: Front rear torque knob

```
Fx = FxProv;
if FRtrqKnob >= 0.5
    Fx(3:4) = 2*(1-FRtrqKnob) * Fx(3:4);
else
    Fx(1:2) = 2*FRtrqKnob * Fx(1:2);
end
FxFL = Fx(1); FxFR = Fx(2); FxRL = Fx(3); FxRR = Fx(4);
```

4.5.2 Advanced TAA

Developing a robust algorithm for the torque allocation is a tricky task to carry out, since different constraints must be satisfied and deciding how to split the torques to achieve the target yaw moment can be challenging and can have a variety of alternatives. All these requirements are easily handled by setting up an optimization problem.
The Matlab function exploited for this task is `quadprog`, useful to implement quadratic programming.
The states to be optimized are the torques on the four wheels:

$$\eta = [T_{FL};\ T_{FR};\ T_{RL};\ T_{RR}]$$

The basic idea is that the more a tire is vertically loaded, the more it is able to deliver longitudinal force (without considering saturation). To exploit this principle, it is desired that for the two wheels on each side the ratio of their assigned torques is kept as close as possible to the ratio between the front and the rear vertical loads. Moreover, the control is allowed to diminish the delivered torque (within a limit), with respect to the one requested by the driver, but paying a certain cost.
The cost function is defined as:

$$J(\eta) = \left(\frac{T_{FL}}{T_{RL}} - \frac{F_{zFL}}{F_{zRL}}\right)^2 + \left(\frac{T_{FR}}{T_{RR}} - \frac{F_{zFR}}{F_{zRR}}\right)^2 + \gamma \cdot (\sum_{i=1}^{4} T_i - T_{demand})^2 \tag{9}$$

To avoid mathematical issues related to the divisions, Equation 9 is rearranged presenting only multiplications:

$$J(\eta) = (F_{zRL} \cdot T_{FL} - F_{zFL} \cdot T_{RL})^2 + (F_{zRR} \cdot T_{FR} - F_{zFR} \cdot T_{RR})^2 + \gamma \cdot (\sum_{i=1}^{4} T_i - T_{demand})^2 \tag{10}$$

The squares are introduced to make this cost function quadratic, and so to ensure a minimum to the optimization problem. This function, to be read by Matlab, must be written in the

form:

$$J(\eta) = \frac{1}{2} \cdot \eta^T \cdot H \cdot \eta + f^T \cdot \eta \tag{11}$$

where H is the Hessian matrix and f is the gradient:

$$f = \nabla J(T) = \begin{bmatrix} \frac{\partial J}{\partial \eta_1} & \dots & \frac{\partial J}{\partial \eta_4} \end{bmatrix}^T$$

$$H = \begin{bmatrix} \frac{\partial^2 J}{\partial \eta_1^2} & \cdots & \frac{\partial^2 J}{\partial \eta_1 \partial \eta_4} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 J}{\partial \eta_4 \partial \eta_1} & \cdots & \frac{\partial^2 J}{\partial \eta_4^2} \end{bmatrix}$$

Both of them are numerically computed at a generic point $\eta = [0;\ 0;\ 0;\ 0]$. To do so, the difference quotient is used:

$$f = \begin{bmatrix} \frac{J(\eta_1+dT,\ \eta_2,\ \eta_3,\ \eta_4)}{dT} & \dots & \frac{J(\eta_1,\ \eta_2,\ \eta_3,\ \eta_4+dT)}{dT} \end{bmatrix}^T$$

```
% Cost function defintion:
J = @(T) (Fz(3)*T(1) - Fz(1)*T(3))^2 + ...
         (Fz(4)*T(2) - Fz(2)*T(4))^2 + ...
         gamma*((T(1) + T(2) + T(3) + T(4)) - trqDemand)^2;

% Numerical computation of the cost function gradient:
T0 = zeros(4,1); % Generic point within the cost function domain
dT = 1e-5;       % Small step for finite difference
f = zeros(4,1);  % Gradient initialization
for i = 1:4
    Tincremental = T0;
    Tincremental(i) = Tincremental(i) + dT;
    f(i) = (J(Tincremental) - J(T0)) / dT;
end
```

Concerning the Hessian matrix terms, the difference quotient is applied twice for their computation:

$$\frac{\partial^2 J}{\partial \eta_i \partial \eta_j} \approx \frac{J(\eta_i + dT, \eta_j + dT) - J(\eta_i + dT, \eta_j) - J(\eta_i, \eta_j + dT) + J(\eta_i, \eta_j)}{dT^2}$$

```
% Numerical computation of the cost function Hessian matrix:
H = zeros(4,4); % Hessian matrix initialization
for i = 1:4
```

```
    for j = 1:4
        Tincremental1 = T0; Tincremental2 = T0; Tincremental3 = T0;
        Tincremental1(i) = Tincremental1(i) + dT;
        Tincremental1(j) = Tincremental1(j) + dT;
        Tincremental2(i) = Tincremental2(i) + dT;
        Tincremental3(j) = Tincremental3(j) + dT;
        H(i,j) = (J(Tincremental1) - J(Tincremental2) -
           J(Tincremental3) + J(T0)) / dT^2;
    end
end
```

A warm start set of values is provided to the `quadprog` function in order to enhance the optimal point searching time:

$$T_{WS} = [1;\ 1;\ 1;\ 1] \cdot \frac{T_{demand}}{4}$$

```
trqDemand = 4 * EMtrqAvailable * Throttle; % [N*m] total maximum
   torque demanded by the driver
Tws = ones(4,1)*trqDemand/4;
```

The third term in Equation 9 presents a gain γ, that is introduced to assign a higher weight to the delivered torque decrease, when the yaw moment is lower. Indeed, it is defined as:

$$\gamma = \begin{cases} \frac{\gamma_0}{M_z} & M_z > \epsilon \\ \frac{\gamma_0}{\epsilon} & M_z < \epsilon \end{cases}, \quad with\ \epsilon = 3N \cdot m,\ \gamma_0 = 500N \cdot m$$

The idea is that at a lower controlled yaw moment corresponds a lower overall control effect from the LLC.

```
eps = 3;
gamma0 = 5e2;
if abs(Mz)>eps
    gamma = gamma0/Mz;
else
    gamma = gamma0/eps;
end
```

Using the `quadprog` function it is possible to directly feed upper torque limits coming from the traction controller, as well as lower torque limits for the regenerative braking. Since the

traction control is still in a development phase, the upper torque limits correspond to the maximum deliverable torque from the EM. The braking scenario is not considered in this development phase, so the lower torque bounds are set equal to 0:

$$u_b = [1;\ 1;\ 1;\ 1] \cdot T_{EM,max}$$

$$l_b = [0;\ 0;\ 0;\ 0]$$

```
lb = zeros(4,1);
ub = trqMax * ones(4,1);
```

Defining constraints functions is very easy as well. Two kind of constraints can be imposed:

- Soft constraints: the states must be lower than a certain set of thresholds;
- Hard constraints: the states must be equal to a certain set of values.

Imposing soft constraints it is possible to avoid that the torque delivered exceeds the torque demand, violating the FS rules, and to avoid that the torque delivered diminishes too much with respect to the demand, impacting on the drivability. A maximum decrease of the 20% is allowed. To impose these constraints a matricial form is exploited:

$$[A] \cdot \{\eta\} \leq \{b\}$$

where:

$$A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 \end{bmatrix}, \quad b = \begin{bmatrix} T_{demand} \\ -0.8 \cdot T_{demand} \end{bmatrix}$$

```
A = [1  1  1  1
     -1 -1 -1 -1];
b = [trqDemand; -0.8*trqDemand];
```

In the same way, but using hard constraints, it is imposed to the control to find a solution that guarantees the requested yaw moment:

$$[A_{eq}] \cdot \{\eta\} = \{b_{eq}\}$$

where:

$$A_{eq} = \begin{bmatrix} \left[-\frac{t}{2} \cdot cos(\delta_L) + l_f \cdot sin(\delta_L) \right] \cdot \frac{1}{\tau_{GB} \cdot R_l} \\ \left[\frac{t}{2} \cdot cos(\delta_R) + l_f \cdot sin(\delta_R) \right] \cdot \frac{1}{\tau_{GB} \cdot R_l} \\ -\frac{t/2}{\tau_{GB} \cdot R_l} \\ \frac{t/2}{\tau_{GB} \cdot R_l} \end{bmatrix}^T, \quad b_{eq} = \begin{bmatrix} M_z \end{bmatrix}$$

```
Aeq =
   [(-t/2*cos(SteeringAngleL)+lf*sin(SteeringAngleL))/(taoGB*Rl),...
(t/2*cos(SteeringAngleR)+lf*sin(SteeringAngleR))/(taoGB*Rl),...
-t/2/(taoGB*Rl),...
t/2/(taoGB*Rl)];
beq = Mz;
```

When solving the optimization problem, it is normal to imagine that there are conditions in which the controlled yaw moment cannot be reached by the torque allocation. In these conditions, the used `quadprog` returns an error code. By constantly reading this code, it is possible to tackle this situation: the target yaw moment is decreased until a feasible value can be provided by the control.

```
options = optimoptions('quadprog', 'Algorithm', 'active-set');
[T, ~, exitFlag] = quadprog(H,f,A,b,Aeq,beq,lb,ub,Tws,options);
while exitFlag ~= 1
    Mz = Mz*0.995;
    beq = Mz;
    options = optimoptions('quadprog', 'Algorithm', 'active-set');
    [T, ~, exitFlag] = quadprog(H,f,A,b,Aeq,beq,lb,ub,Tws,options);
end
```

4.6 Vehicle model

The torque vectoring control requires a dual track vehicle model in order to properly simulate its influence on the vehicle behaviour. For this purpose, the Vehicle Body 3DOF is exploited, belonging to the Simulink library Vehicle Dynamics Blockset [3].
All the vehicle characteristic parameters, inputs of the block, are introduced in Section 2. The external inputs of the block are the steering angles at tire level and the longitudinal forces (see Section 4.2).

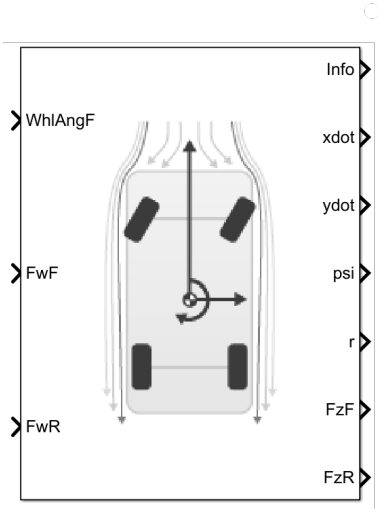

Figure 16: Vehicle Body 3DOF Dual Track

This vehicle model is accurate enough for the TV simulation purposes. In the future it would be interesting to test the same controller with a more accurate simulator, such as the Simulink Simscape one, currently under development by the team, or the CarMaker one.

4.7 TV simulation

In this Section, some performance parameters are introduced and compared for the different simulations undertaken, with the possible combinations of ULCs and LLCs.

4.7.1 Performance parameters

During the development of a new system, it is important to set clear and feasible targets to be reached, in terms of performance parameters. They can be even useful to quantify the effectiveness of one control solution with respect to another one.
For this sake, five fundamental parameters are introduced, even if not all of them are tracked during the simulations:

1. Under steering coefficient mean value

The under steering coefficient, as explained in Section 4.4.2, is directly related to the vehicle behaviour. Since the overall amplitude of this signal over the time interval is relevant, the RMS value is chosen as a performance parameter, as it is the case for the next introduced signals. The K_{US} target value is set equal to 0, thus an enhanced performance is reached

when the controller minimizes its RMS value.

2. Vehicle side slip angle RMS value

A higher value of vehicle side slip angle physically means that the vehicle points more towards the turn centre when cornering. Keep in mind that an increase of this value doesn't always result in an increase of performance, since its influence on the vehicle handling is strictly dependent on the current understeering coefficient.
Since the overall amplitude of this signal over the time interval is relevant, the RMS value is chosen as a performance parameter, as it is the case for the next introduced signal.

3. Yaw rate RMS value

Intuitively, the yaw rate indicates how fast the vehicle rotates on itself, thus an increase of the RMS value related to its signal, combined with a decrease of the under steering coefficient mean value, means that the vehicle rotates faster, being more stable at the same time. This outcome would lead to an enhancement both on performance and on comfort.

4. Turn radius

This parameter is the most tangible value useful to practically understand the torque vectoring effects. The goal of this control is to decrease the radius value, for the same driving inputs, resulting in a more agile vehicle behaviour.
In the driving scenario introduced in Section 4.3, the starting steering manoeuvre leads to an almost closed-circle trajectory.

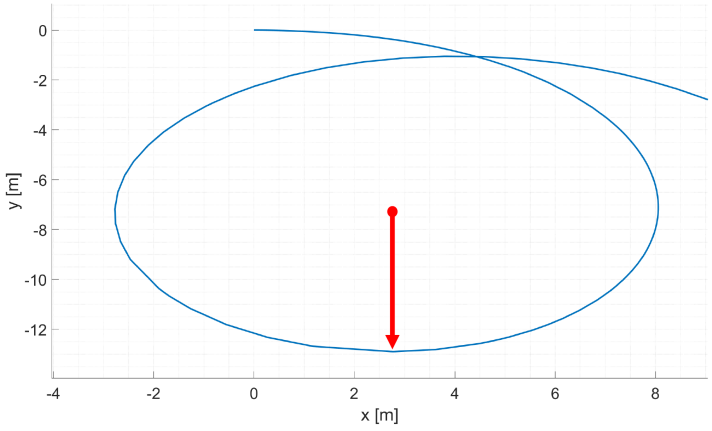

Figure 17: Turn radius example

5. Lateral acceleration enhancement

The implementation of a TV system is supposed to improve the vehicle handling, with a resulting increase in the developed lateral acceleration, for a given steering angle value [2]:

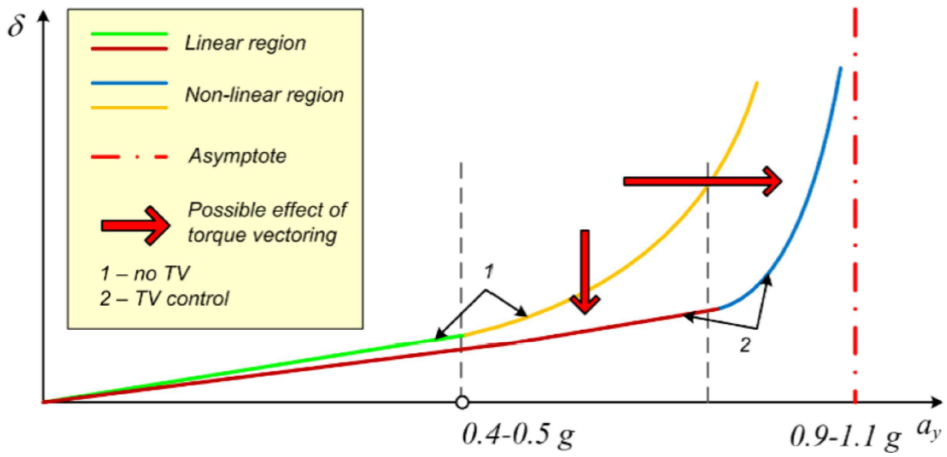

Figure 18: Performace parameter - ayVSdelta

It is expected that going beyond a steering angle threshold doesn't lead to an increase in the lateral acceleration.
An $a_{y,max}$ target of $2.0g$ with TV is set.

4.7.2 Simulation results

5 different simulations are carried out:

- Only the vehicle model without control systems is simulated to retrieve a baseline;
- The vehicle model with TV is simulated: the ULC and the LLC basic and advanced logic are tested.

Table 2: Simulation results

Parameter	No controls	ULCb, LLCb	ULCb, LLCa	ULCa, LLCb	ULCa, LLCa	U. m.
$RMS(K_{US})$	0.277	0.241	0.263	0.154	0.181	–
$RMS(\beta)$	0.056	0.086	0.091	0.093	0.073	deg
$RMS(\omega_\psi)$	0.419	0.699	0.747	0.846	0.671	rad/s
R_{turn}	–	9.7	–	4.6	5.9	m

The trajectory for each simulation is shown in the following picture:

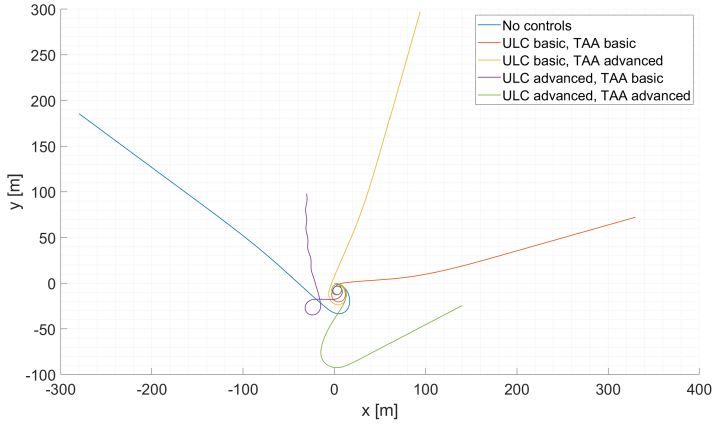

Figure 19: Results trajectories

A zoom into the starting region is useful to better analyse the controllers behaviour:

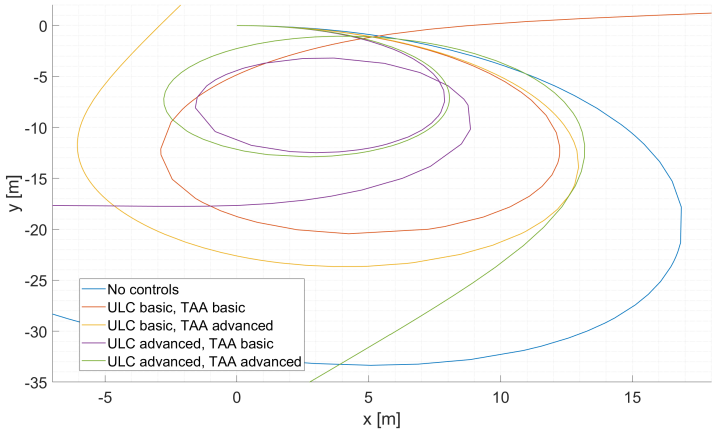

Figure 20: Results trajectories zoom

4.7.3 Results discussion

Let's split the analysis for the two different ULC logics. It is more interesting to begin with the basic logic case, since it will be the one implemented in a short time on the real FS vehicle.

Looking at the numerical results in Table 2, the advanced LLC shows an increase in the $RMS(\beta)$ and $RMS(\omega_\psi)$ values, with respect to both the vehicle without controls and the vehicle with a basic LLC. This increase stands for a higher vehicle rotation and, thus, enhanced agility. Also looking at the final trajectory (Fig. 20), the LLC basic exhibits a closer one.

Before drawing conclusions on the LLC performances, it is helpful to introduce the yaw moment loss parameter, and to take a look at its trend for the two logics:

$$M_{z,\ loss} = \frac{M_{z,\ demand} - M_{z,\ delivered}}{M_{z,\ demand}} \cdot 100; \tag{12}$$

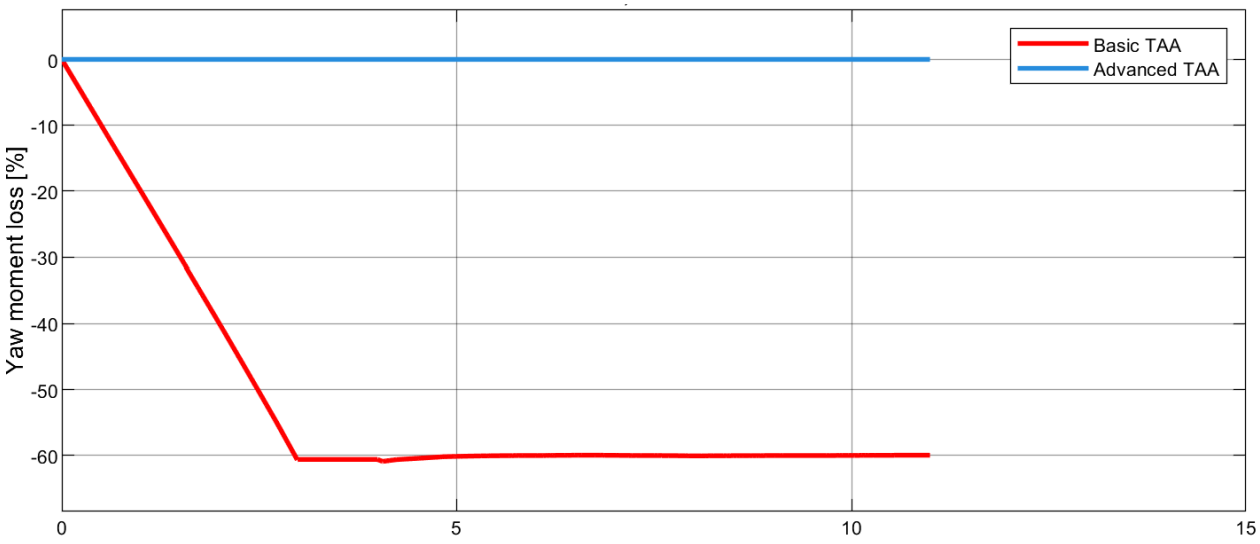

Figure 21: Yaw moment loss comparison

The last seconds of simulation are not visualized, since numerical errors arise when values equal to 0 are involved in the calculations. This condition is not important for the discussion itself.

It is noticeable how the basic LLC attends a negative loss, meaning that the yaw moment it provides is higher than the demanded one. Instead, the advanced LLC precisely satisfies the demand coming from the ULC. This outcome is very important, and it highlights how the advanced LLC logic enacts a more accurate and reliable control. At the same time, more constraints are satisfied, limiting more the deliverable yaw moment, but overall improving the drivability.

Another interesting parameter, within the set constraints, is the torque loss, defined as:

$$T_{loss} = \frac{T_{demand} - \sum_{i=1}^{4} T_i}{T_{demand}} \cdot 100; \tag{13}$$

Its value is always bounded to 20%, leaving the driver a good sense of control over the vehicle:

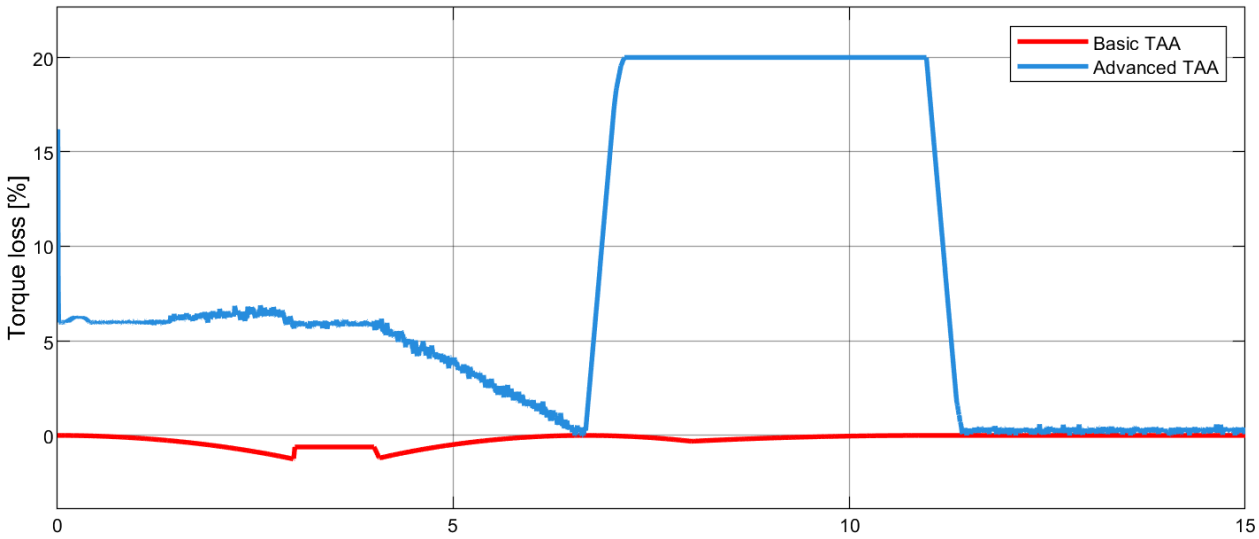

Figure 22: Torque loss comparison

It is important to consider that the currently used model doesn't account for the vertical load influence on the tire behaviour. Thus, the major advantage related to the advanced LLC is not kept during the simulation phase, even if a significant improvement is expected in a real scenario.

When moving to the advanced ULC logic, it is interesting to notice how the trajectories shrink for both the cases of LLC logics (Fig. 20), standing for an improved performance. Moreover, the overall vehicle behaviour is more close to the neutral steering condition, as highlighted by the $RMS(K_{US})$ value, reduced by about the 40% for both the cases, with respect to the baseline.

In conclusion, the advanced ULC logic leads to a further improvement in the handling, as well as in the overall performance. This result makes this logic very interesting to better investigate and tune in the next future.

4.8 Kistler sensor

As anticipated in Section 4.4.2, the vehicle side slip angle measurement, on the FS vehicle, has not been validated until now. The importance of correctly sensing this vehicle state is crucial, as highlighted in these Sections. Thus, the necessity of validating the current measurement system led to the adoption of a non-contact optical sensor: Kistler Correvit S-CE.

Figure 23: Correvit S-CE

4.8.1 Sensor adoption

The Correvit is longitudinally mounted on the back of the vehicle at a height of $30cm$ as indicated in the user manual.

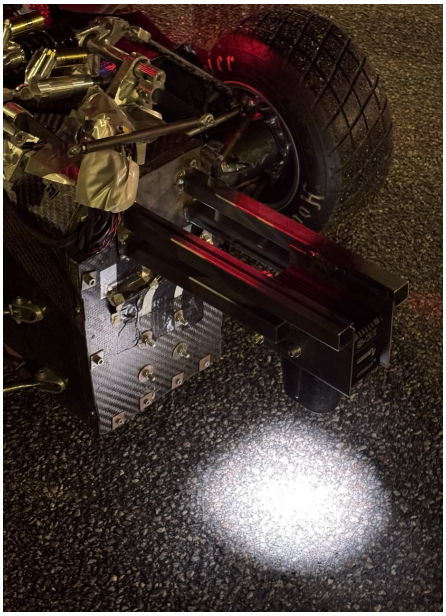

Figure 24: Correvit mounting

Its output data are two analog signals: longitudinal speed (V_x) and lateral speed (V_y). To read these analog signals by means of the ECU (electronic control unit), it is necessary to translate them into CAN bus signals. To undertake this duty, a data-logger from AIM is used: the PDM32.

Figure 25: PDM32 data-logger

Both these instruments (the sensor and the data-logger) are powered at $12V$. Since the LV (low voltage) battery pack is at $24V$, it is necessary to implement a DC-DC buck converter. The global architecture of the system implemented on the car is shown:

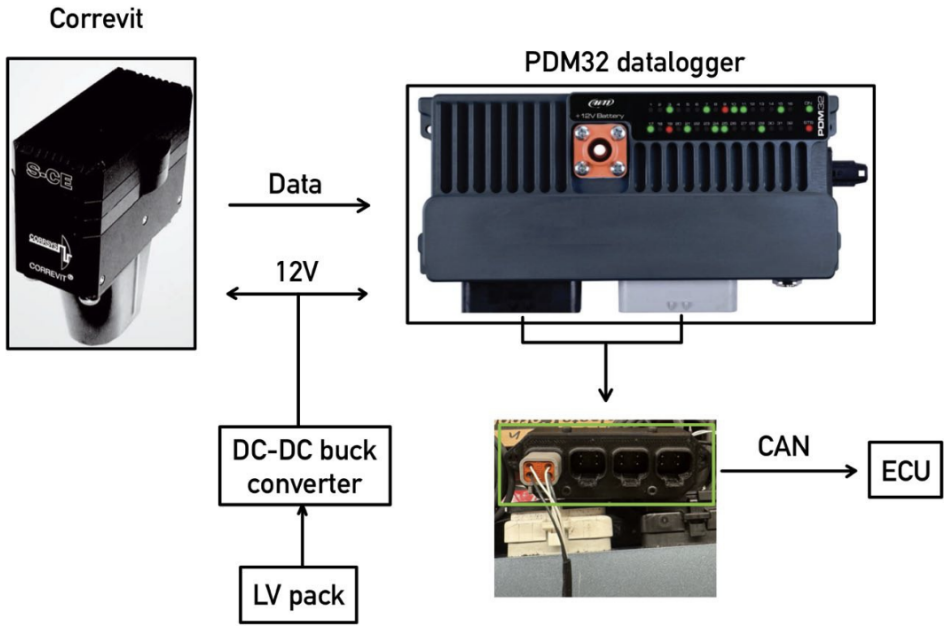

Figure 26: Sensor implementation architecture

4.8.2 Track tests results

Until now, only one track testing session has been completed, with the Correvit sensor embedded in the vehicle. Some adjustments to the hardware assembly are still needed to collect proper data. Since the lateral speed data measurement is still under study, only the longitudinal speed measurements from a run are shown below, in comparison with the data from the GPS coupled to the IMU:

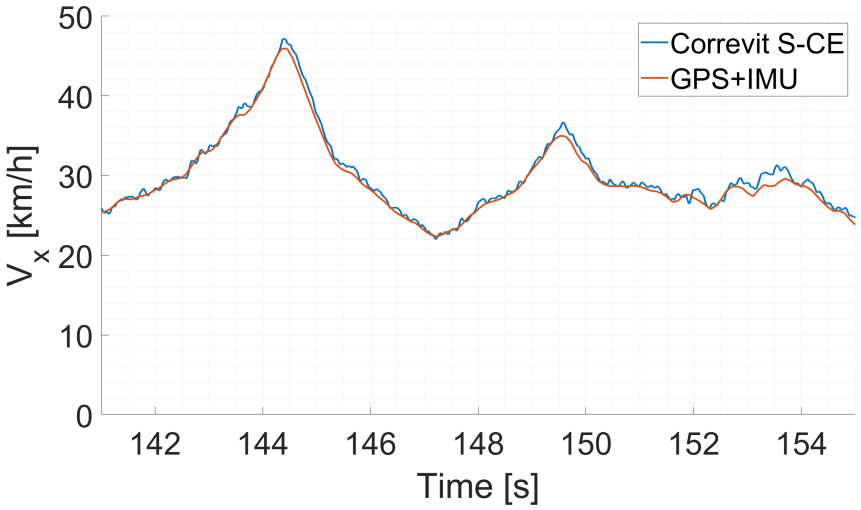

Figure 27: Longitudinal speed data

The GPS+IMU measurement method looks to be quite accurate. Indeed, the RMS value of the error between the two signals is pretty low:

$$\epsilon = \frac{V_{x,Correvit} - V_{x,GPS+IMU} \cdot 100}{V_{x,Correvit}}$$

$$RMS(\epsilon) = 2.13\%$$

Consider that the lateral speed data are particularly important as well for the overall validation. For this reason, before concluding that the current measurement method is accurate enough, it is necessary to ensure that the hardware setup is properly working and that the data are collected over different possible driving scenarios.

References

[1] V. Hallbeck et al. “KTH Formula Student VDPC Project Report”. In: (2021).

[2] Klait Bani. “Integrated Torque Vectoring and Path Following using Nonlinear Model Predictive Control”. In: (2023).

[3] Thomas Gillespie. “Fundamentals of Vehicle Dynamics”. In: (1992).

[4] R. Wikström. “Monocoque chassis design and optimization: Composite optimization of FSAE Chassis”. In: (2023).