Lab 1: Designing ABS and traction control systems

SD2231– Applied vehicle dynamics control

April 20, 2023

Alessandro Riva

Tawsiful Islam

1 Task 1. Determine control parameters

1.1 Task 1.a

The derivation of the transfer function for the mass-spring system can be found by first starting to set up the equation of motion. From there, the equation can be Laplace transformed and T(s) can be solved according to the equations below:

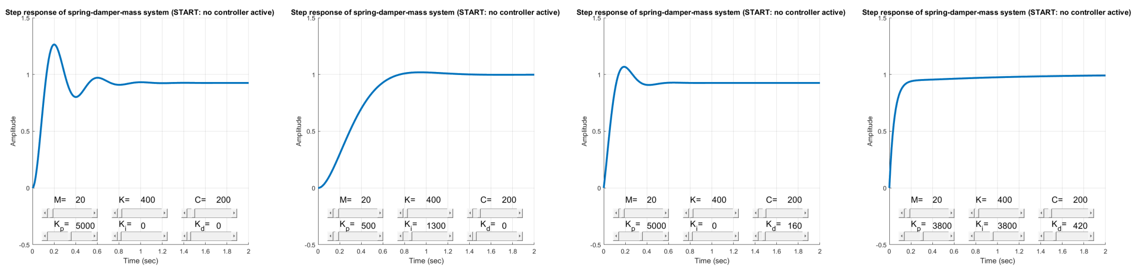
$$\begin{aligned} m\ddot{x} &= F - kx - c\dot{x} \\ mXs^2 &= F - kX - cXs \\ (ms^2cs + k)X &= F \\ \frac{1}{ms^2 + cs + k} &= T(S) = \frac{X(s)}{F(s)} \end{aligned}$$

1.2 Task 1.b

To find the control parameters for the P-, PI-, PD-, and PID-control, tuning was done on Matlab that had a desired behaviour. The parameters we got are seen in table 1 and their responses are seen in figure 1

Table 1: PID parameters for the different controllers

Controller	Kp	Ki	Kd
P-controller	5000	0	0
PI-controller	500	1300	0
PD-controller	5000	0	160
PID-controller	3800	3800	420



(a) P (b) PI (c) PD (d) PID

Figure 1: Step response for the different controllers.

1.3 Task 1.c

For open-loop stability, you could use Routh–Hurwitz criterion to determine if the open loop is stable. The criterion says that the system has poles with negative real part if the polynomial in the denominator has positive coefficients. For our system the coefficients are $\frac{c}{m}$ and $\frac{k}{m}$, where c,k,m are positive values, thus the open-loop system is stable.

To analyse the stability of the closed-loop systems, Nyquist criterion was used. One can detect instability by checking if the curve in the plot (representing the loop gain $L(i\omega) = C(i\omega)T(i\omega)$) (C: controller, T: Plant) Since $L(i\omega)$ has no unstable poles, if Nyquist plot doesn't encircle s = -1 in the zero-pole plane, then closed-loop stability is guaranteed. All the following figures in 2 show that the closed-loop systems are stable.

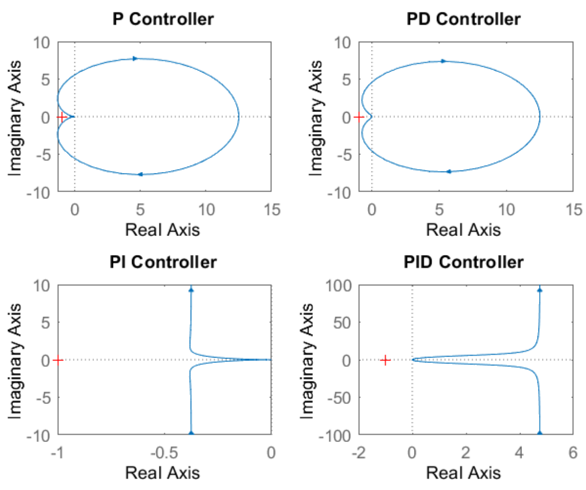
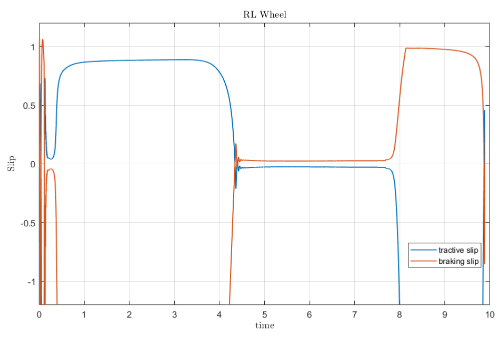


Figure 2: Nyquist plot for the closed loop systems.

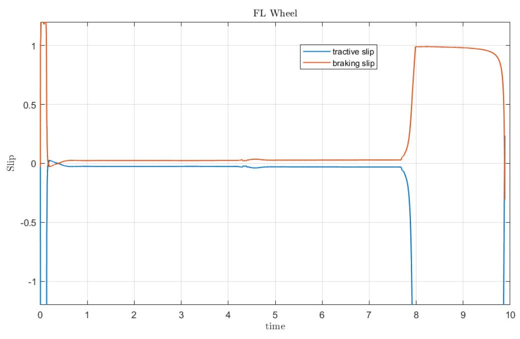
2 Task 2. Slip observer

2.1 Task 2.a

The computation of tracking and braking slip is done by implementing the formulations given in the assignment. The first issue encountered was the division by zero when computing the slip. To avoid that, a saturation block, with a small low threshold, was used to avoid the denominator being identically equal to zero.



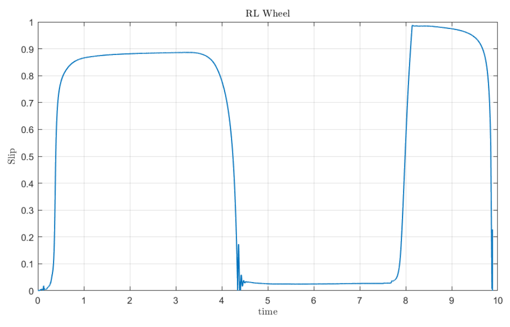
(a) RL Wheel



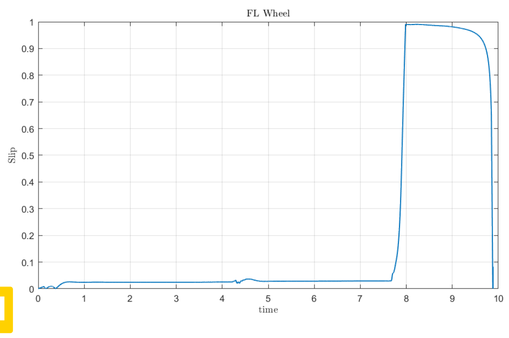
(b) FL Wheel

Figure 3: Tracking and braking slip with threshold = 0.0001 for denominator.

2.2 Task 2.b



(a) RL wheel

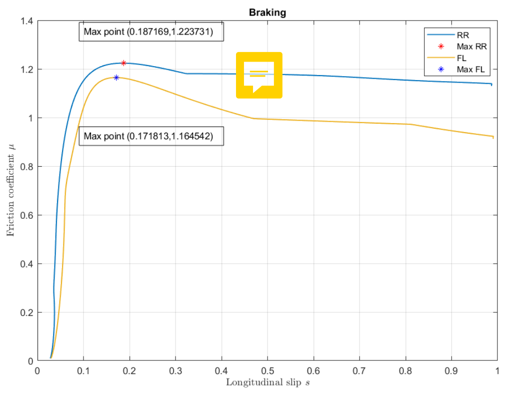


(b) FL wheel

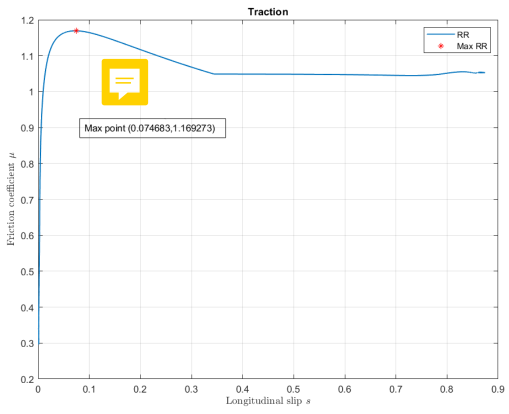
Figure 4: Unified tracking and braking slip, with threshold = 0.5.

Another evident issue is that when the numerical value of the both v_x and ω is close to zero, the values of slip are very high (even with absolute value bigger than 1, which is unfeasible). This can be fixed by selecting a bigger threshold in the saturation block (like 0.5). This method with the threshold is justified because, when the acceleration happens, the car is not moving fast enough for the slip to be interesting for us to observe. Secondly, it is clear also that, during acceleration event, braking slip values are not relevant (and vice versa) and should be disregarded. A unique plot for the slip is generated, in which only the positive value of slip between the two is reported. Such plot is more informative, cause it communicates the behaviour of the wheel in both acceleration and braking events (Figure 4a, Figure 4b).

2.3 Task 2.c



(a) RL wheel



(b) FL wheel

Figure 5: The figures show the friction coefficient against the brake (a) and traction (b) slip of the tires on the car.

The graphs in figure 5 show how the friction coefficient μ changes in relation to increasing slip when braking and accelerating. The maximum values seen in figure 5a are around 1.22 for a slip of 0.187 on the rear wheels and around 1.16 for a slip of 0.172 on the front wheels. In figure 5b, the friction coefficient is 1.17 for a slip of 0.07 on the rear wheels.

The reference traction slip we choose was 0.06 for the rear wheels. For the braking slip, 0.17 was chosen for the rear wheels and 0.15 for the front wheels.

The range around these slip values will allow the controller to work with maximum friction to accelerate and brake the car. The main reason why we chose a reference slip slightly smaller than the slip for maximum friction coefficient is to avoid the instability region, where an increase in slip translates in a loss of grip, losing therefore the slip would be uncontrollable at that point. Having this margin, gives room for some overshoot while still retaining controllability for the slip.

3 Task 3. Design PID controller

3.1 Task 3.a

The activation logic for both the TCS and the ABS is similar: The variables checked to enable the slip controllers are:

1. brake pedal / throttle pedal
2. relevant slip (tractive slip for TCS or braking slip for ABS)
3. acceleration
4. vehicle velocity (only for ABS)

First it will check that relevant slip is above a certain threshold. This threshold was set to be slightly below the reference slip. Setting the threshold at the reference slip caused oscillations in the response as the controller switched on and off when the controller tried to reduce the slip and the driver's input increased the slip when the controller was deactivated. Too low thresholds will accumulate error in the integrator too early, resulting in a larger integral action, that in turn causes bigger overshoots. When selecting our threshold, we kept in consideration also this aspect.

The activation logic also checks which pedal is used so TCS is not activated when the brake pedal is pressed.

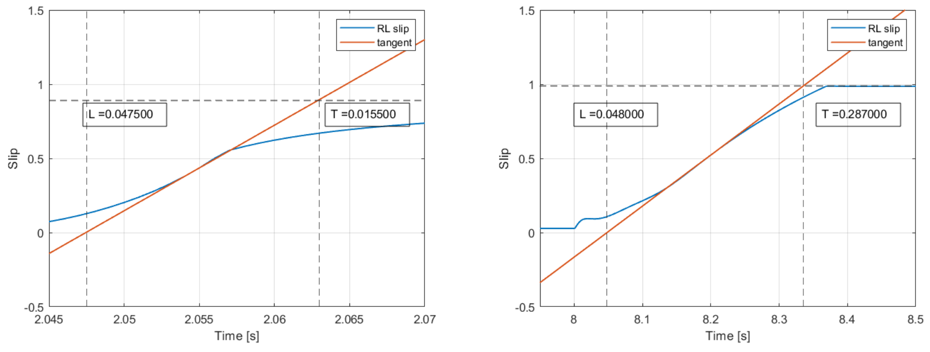
The third condition is checking the acceleration, positive acceleration when accelerating and negative when braking. This condition helps to make sure that right controller is activated if both pedals would be used for some reasons.

The additional condition on ABS checks that $v_x > v_{threshold}$. This is because, as the vehicle comes to a stop, slip naturally goes to 0 too, so we do not want the controller to force the slip to be going to the reference.

When all conditions are met, the controller will be active. If the controller is not active, the driver's signal will be outputted to the system. When the controller is active, the activation logic will output the sum of the adjustment from the controller and the driver input signal. To have the driver be always in control means that, at any time, the signal fed to the actuator should not be bigger than the driver demand. This means that, if the adjustment (added to the driver request) is positive, this will result in the driver not being in control. For this reason, if the controller gives positive signal, we will add zero on top of the driver's signal to avoid giving a signal that is more than the driver intended.

3.2 Task 3.b & 3.c

3.2.1 *Ziegler-Nicholas values*



(a) RL traction (b) RL braking

Figure 6: Computation of L, T for the braking phase and acceleration phase on the RL wheel.

As a starting point for our tuning, we used the Ziegler-Nicholas empirical values, computed from the open-loop step response of the system, in the acceleration phase and in the braking phase. This values were extracted from the transient response of the slip. However, one should remember that their computation is an approximation as they are obtained from the plots seen in figure 6.

The behaviour for the front wheel during braking is assumed to be very similar to the behaviour of the rear wheels. For this reason we settle on same L and T values for both ABS controller located in the front wheels and the back wheels.

3.2.2 *P-controller*

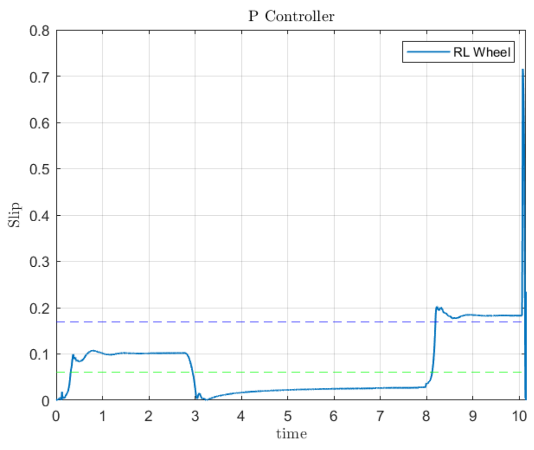


Figure 7: RL wheel with P controller.

When working with only the P-controller, we saw a clear steady-state error as seen in figure 7. By increasing Kp, it is possible to get closer to the reference, but without an integral action, a zero steady-state error can never be obtained. On the other hand, too large Kp gave an overshoot as the control signal was very large for even small errors.

Secondly, increasing Kp showed oscillations around the reference value. This oscillations can be explained by how the controller overshoots so the error changes sign. As a result, the control signal also changes, which results in another big overshoot and change in sign again. These oscillations worsen the settling time. To the limit, for very large Kp, the oscillation can also diverge. Similar oscillation did also occur during the transient phase when increasing Kp, which can be seen during braking in figure 7.

We tried the Ziegler-Nicholas values for the P-controller. As mentioned earlier there are clear overshoots and steady-state errors on the rear wheels and the front wheels. The steady-state was around 0.5 for traction and 0.2 for braking. Clearly, the Ziegler-Nicholas values did not work and gave worse traction as the gain was not large enough. After increasing the proportional gain according to table 2 we achieved following system characteristics seen in table 3.

Table 2: P-controller's gain for the different systems

Gain	Traction rear	ABS front	ABS rear
Kp	1.96	8.97	11.96

Table 3: System characteristics from the P-controllers on the left side

Wheel	Rise time	Overshoot	Settling time	Steady-state error
Traction rear	0.15 s	7.0%	0.72 s	0.04
Braking rear	0.13 s	5.3%	0.28 s	0.02
Braking front	0.14 s	5.5%	0.50 s	0.04

Increasing Kp to reduce the steady-state error causes an increase in the overshoot. At the same time, high Kp improves the rise time and worsens the settling time. If you wanted to reduce the overshoot that would come at a cost of larger steady-state mainly, but also affect the rise time negatively.

3.2.3 PI-controller

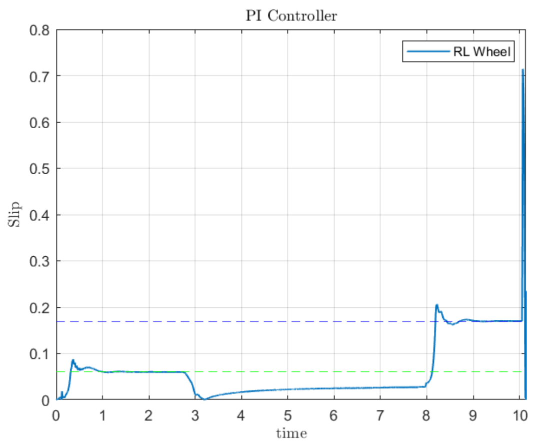


Figure 8: RL wheel with PI controller.

An integral action allows the slip to reach zero steady-state error, which was achieved as seen in figure 8. This is because the integral action is computed on the past error values, and the control action will keep on increasing until the reference is reached. This is different from the P-controller that sent a small control signal when the error was small at that time instance, which led to non-zero steady-state error.

In our implementation, ABS and TCS will always compute an adjustment for the demanded input, then the activation logic will decide whether or not to apply such correction. Therefore, since braking happens at the end of simulation, when the activation logic for ABS turns on, the integral charge of the ABS is already high, because it has been accumulating error over the whole simulation time. This is an undesired behaviour and it was fixed by supplying zero error to the integrator until **Brake pedal** $>$ **0** and **Braking slip** $>$ **threshold**.

Table 4: PI-controller's gain for the different systems

Gain	Traction rear	ABS front	ABS rear
Kp	2.61	8.97	10.8
Ki	12.4	62.3	62.3

Table 5: System characteristics from the PI-controllers on the left side

Wheel	Rise time	Overshoot	Settling time	Steady-state error
Traction rear	0.04 s	45%	0.68 s	0
Braking rear	0.09 s	17.6%	0.38 s	0.001
Braking front	0.12 s	16.6%	0.66 s	0.001

The figure 8 also shows that the integral action increases overshoot too, but it's a delayed action because it is proportional to the integral of the error. For this reason, it is important to decide when to activate the controller and use suitable thresholds, which is one of the reason why implemented our controllers as described above. Increasing Ki would increase this overshoot as the term is proportional to the integral of the error.

We did notice that large Ki could also cause oscillations in the slip response. The delayed action of the integrator, since it depends on past errors, might be non-zero even when the error is zeros, if not tuned correctly. This extra correction can cause oscillations and longer settling times. When trying with the Ziegler-Nicholas values for the PI-controller, they did help with eliminating the steady-state error for ABS, but not for TCS. For TCS we got large overshoot and oscillations when settling. The ABS behaved as desired without big overshoot and oscillations. We tuned the gains and we ended up with the gains seen in table 4. Finally, when looking at the system characteristics in table 5 after the new gains, we noticed that the steady-state error was eliminated. The rise time did also improve with the PI-controller. However, due to the delayed response from integral action, we can see the overshoot and the settling time significantly increased. The overshoot is the biggest drawback of adding an integrator to the controller.

3.2.4 PD-controller

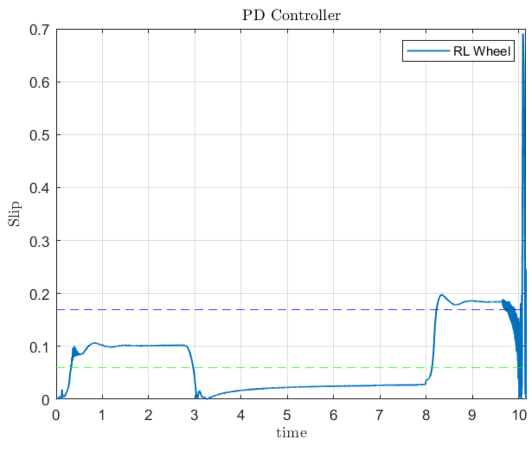


Figure 9: RL wheel with PD controller

Adding a derivative part to the P-controller helped with reducing the overshoot as seen in figure 9. However, with our system, Kd could not increase too much without seeing oscillations. Kd worsens the oscillation at the end of braking whilst reducing the overshoot. Large Kd could also amplify oscillations in the transient phase. Therefore, if the error fluctuated enough to cause the proportional term to fluctuate, the derivative part would react and also amplify the oscillations This is possibly also the reason why the oscillation at the very end is bad in figure 9 compared to the slip response from P- and PI-controller. Due to this sensitivity, the Kd usually needed to be small. Therefore, we needed to trade off between reducing overshoot and containing oscillations.

Table 6: PD-controller's gain for the different systems

Gain	Traction rear	ABS front	ABS rear
Kp	1.96	7.17	10.8
Kd	0.03	0.172	0.344

In the end, the gains we settled on are seen in table 6 where the Kp are similar to the ones in table 2, but slightly smaller to adjust to the sensitivity of the derivative part. The Kd was set very low and had a small effect on the slip response and this is mainly due to the sensitivity it had.

Table 7: System characteristics from the PD-controllers on the left side

Wheel	Rise time	Overshoot	Settling time	Steady-state error
Traction rear	0.39 s	5.0%	0.67 s	0.04
Braking rear	0.13 s	2.7%	0.23 s	0.02
Braking front	0.19 s	3.8%	0.64 s	0.04

The resulting system characteristics are seen in table 7. Compared to PI-controller in table 5, there is a steady-state error that the derivative part did not affect (as the integral part is not present). The derivative part predicts future values using the error, which helps with dampening the slip response and avoiding overshoot. Therefore, we could see a little lower overshoot compared to the P-controller's and improved settling time. We saw that the result of avoiding overshoot also resulted in longer rise time, but with almost the same settling time compared to the results in table 5.

3.2.5 *PID-controller*

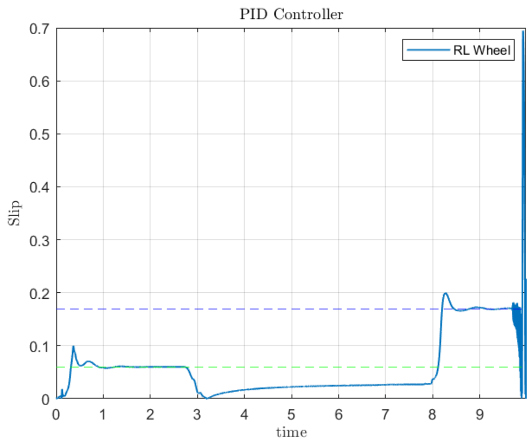


Figure 10: RL wheel with PID controller

The slip response from the PID controller ended up being similar to the PI-controller. As previously mentioned for the PD-controller, adding the derivative part caused more noise at the end of the braking in the ABS for example, which is seen in figure 10. To avoid the noise, the Kd gain needed to be relatively small to have almost no effect. In essence, the PID-controller became very similar to a PI-controller.

There were other reasons why having a derivative part also caused issues for TCS. Increasing Kd to try dampening the overshoot lead to oscillations in the transient phase of the response. This is very similar to what was seen in the response for PD. This goes back to what was discussed about PD-controller. Oscillations of this kind are not desirable as the control signal also has the same oscillations sent to the actuators. Such behaviours in the signal would put stress on the actuator and, in reality, this would also translate to jerky, unpleasant movements of the car and its passengers. Therefore, we prioritised having smoother control action at cost of having a larger overshoot just to reduce the oscillation in the transient phase, but also at the end of braking.

Table 8: PID-controller's gain for the different systems

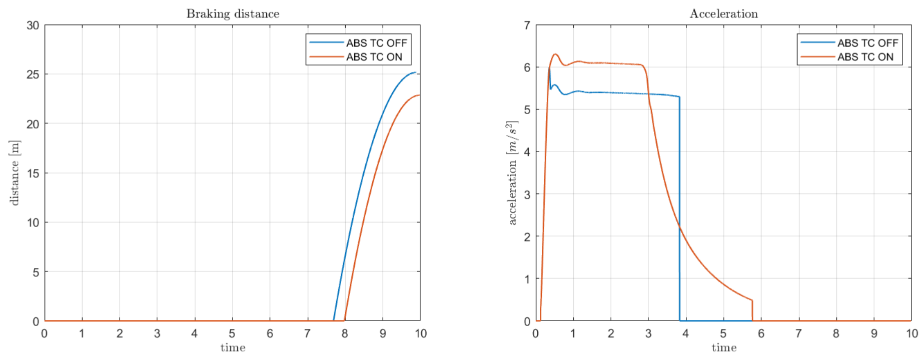
Gain	Traction rear	ABS front	ABS rear
Kp	1.57	14.4	7.17
Ki	13.7	74.7	74.7
Kd	0.010	0.172	0.172

The gains we settled on are seen in table 8. Adding the derivative part caused us to lower the Kp on some of the systems to avoid too big responses. As mentioned earlier, Kd is kept very small as we are trying to reduce the oscillations.

Table 9: System characteristics from the PID-controllers on the left side

Wheel	Rise time	Overshoot	Settling time	Steady-state error
Traction rear	0.09 s	65%	0.65 s	0
Braking rear	0.10 s	17.0%	0.38 s	0
Braking front	0.13 s	12%	0.67 s	0.001

The system characteristics in table 9 show that they are similar to the characteristics in table 5, but are very different from the characteristics for PD in table 7. Looking at the table for PID, one can say that the PID is almost like a PI-controller with the gains we chose.



(a) Braking distance (b) Acceleration

Figure 11: Comparison of the braking distance and acceleration with and without the controllers.

To analyse the overall performance of ABS and TCS, we evaluated the braking distance and the accelerations and compared it to the performance with the controllers disabled. The plot and the improvements are seen in figure 11 and table 10.

Table 10: Improvements

	Braking distance	Acceleration
Improvements	2.31 m	0.69 m/s^2

3.3 Task 3.d

It's not always necessary to use all the gains for both controllers. For example for the ABS, having a derivative part worsen the slip at the end when the speed reduces. Because the derivative part reacts to changes in the error, the slip could change significantly due to numerical issues when the it is close to zero. We noticed that this term grew and became dominant, which led to oscillations in the end. In that case, it might be suitable to only work with PI-controller to avoid this issue in cost of having a slightly larger overshoot. For the TCS, the Kd we settled on, seen in table 8, was very low and practically did not affect the control signal. As the derivative part did not have a significant effect, Kd could also be removed to ensure that the system is not sensitive to noise and amplify any oscillations during the acceleration. Having a simpler controller for TCS also makes it more robust if the overshoot we got with PI-controller is acceptable for a dry road.

4 Task 4. PID controller optimisation

4.1 Task 4.a

The implementation of an anti-windup architecture is beneficial whenever the system has saturation on the control action (like in this case, the actuator can only output a certain maximum torque). This is related to how the error is not wound down in time to react properly when the error changes sign. This windup happens because the controller commands a larger control signal, but it is limited by the actuator, resulting in slower improvements on the error, which keeps on being added up in the integrator. This integral windup results in the overshoot that has been seen earlier for the PI- and PID-controller. Therefore, to improve overshoots and settling times, an anti-windup is used that takes into account the actuator's limitation by using a saturation block and model for the signal after the actuator. This implementation of tracking then allows to wind down the integrator much faster when the error changes signs.

Table 11: PID Parameters with Anti-Windup

	Kp	Ki	Kd
TCS	1.566316	13.739612	0.0093
ABS Rear	5.98	74.74	0
ABS Front	8.61	112.1	0

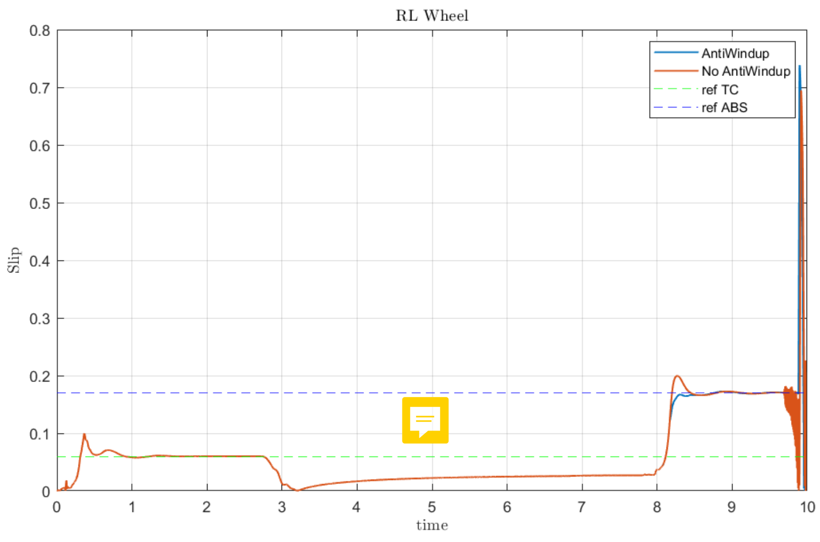


Figure 12: Comparison between performance for one wheel

After the addition of the anti-windup architecture, the PIDs were slightly retuned. In particular, the values of the Kp were lowered to eliminate small oscillations that would appear with the previous tuning. The new gains can be seen in table 11.

In the braking part, there are no longer overshoots, because the anti-windup architecture prevents the integrator from accumulating error once the control action is saturated as seen in figure 12. On the other hand, the acceleration performance remains unchanged: this is expected since anti-windup is effective in those scenarios where the error keeps the same sign for long time, and the integral charge increases (i.e. the braking phase). In the acceleration phase, since it happens in the beginning of simulation, the internal charge of the integrator does not increase as much as in the ABS.

4.2 Task 4.b

To change the road friction during the acceleration and braking event, we designed a logic that supplies different values for μ based on the velocity and acceleration of the car. the result is that:

- during acceleration, when $10 < v_x < 20[m/s], \mu$ changes from 1 to 0.7, then is resetted to 1
- during braking, when $v_x < 10, \mu$ switches from 1 to 0.3

At first, the controller with anti-windup (which was the one that delivered the best behaviour overall) is used. The abrupt changes in road friction cause the slip to suddenly change, so sudden spikes are expected.

A retuning of the controller can be done, with the aim of both reducing the intensity of the spikes and improving the settling times. The new gains we ended up with are seen in table 12.

Table 12: Retuned PID with Anti-Windup

	Kp	Ki	Kd
TCS	2.74	13.74	0.0186
ABS Rear	7.18	74.75	0
ABS Front	7.18	149.45	0

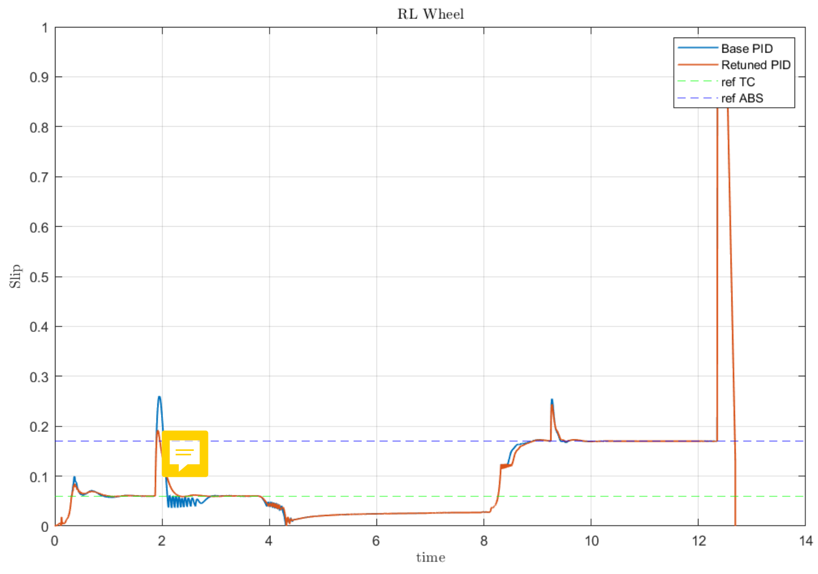


Figure 13: Retuning for road friction changes

So, for instance, the PID for Traction control has bigger values of Kp and Kd, to make the reaction to the spike faster, so that it readjust to the reference in an acceptable time and reduces the overshoot due to the change in road friction. We can see that this change does improve the slip response as seen in figure 13.

It's important to notice that the spikes seen in figure 13 cannot be removed completely. Therefore, an aggressive retuning of the PID to counteract the road friction changes can only mitigate the disturbances up to a certain point. Moreover, the trade-off between optimal performance and stress on the actuator must be considered. Maybe it is possible to reduce further more the spikes, but this would be at the expenses of the actuators.