Dynamic bicycle model MPC design

1st Jaume Albardaner i Torras
Universitat Politècnica de Catalunya (UPC)
jaume.albardaner@estudiantat.upc.edu

***Abstract*—This document describes the literature on dynamic bicycle models. Afterwards, a comparison is conducted on different types of Model Predictive Controllers (MPC) to model this system.**

***Index Terms*—MPC, bicycle model, nonlinear MPC, linearization, dynamic model, kinematic model, noise resistance.**

I. Introduction

Usually, when the behavior of a vehicle has to be simulated, many variables must be accounted for: velocity at each wheel, friction coefficients, yaw angle, etc. Although in some isolated cases it can be a necessity to work with every possible variable, it is not always the case. Usually, working with all the information available requires a computational load that not everybody wants to work with. Thereafter, in most cases the behavior of the system is simplified to that of a bicycle (Figure 1).

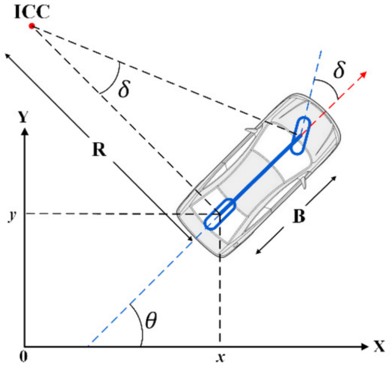


Fig. 1. Vehicle with complex behavior simplified into a bicycle. Font: [1].

It must be noted that despite the conversion, the bicycle behavior can still be split into two groups: assuming it slips (dynamic model) and assuming it does not (kinematic model). The main difference between the two is that the first is assumed to go at higher velocities than the second one. In the case where the assumption does not hold, the model behaves with an increased error [2] [3].

The vehicles that will want to do the simplification in order to ease computations will perform it in order to either obtain results in a higher frequency or because they have limited computational resources. In this project, the dynamic model has been selected because of its complexity (compared to the kinematic) and because it is the most commonly used in the industry [4].

In this document not only will the dynamic bicycle model be explored, but it will be used towards the implementation of a predictive controller. This MPC will employ the bicycle model to generate the optimal control signal in order to follow a reference trajectory.

II. Problem statement

The problem that this document wishes to solve is the design of a predictive controller based on the dynamic bicycle model (Figure 2).

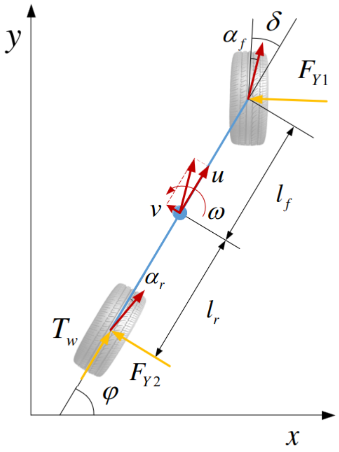


Fig. 2. Dynamic bicycle model. Font: [5].

To obtain the dynamic bicycle model, a similar procedure to [4] and [5] has been followed.

A. Newton's 2nd law

Newton's 2nd law of motion is applied to the external forces of the bicycle (Equation 1).

$$\sum F = ma \tag{1}$$

For the vehicle, there are three external forces that are applied, one in the front wheel (F_{Y1}) and two in the rear (F_{Y2} and $F_{X2} = \frac{Tw}{R_w}$) [1]. These forces can be separated depending on whether they affect the longitudinal axis (along u) or the transversal axis (along v) of the bicycle (Equation 2).

$$\begin{cases} \sum F_u = F_{X2} - F_{Y1} sin(\delta) \\ \sum F_v = F_{Y1} cos(\delta) \end{cases} \tag{2}$$

[1] T_w corresponds to the driving torque applied on the rear wheel, which has been converted to a force for simplicity.

The equations in vehicle-frame will look as depicted in Equation 3.

$$\begin{cases} F_{X2} - F_{Y1}sin(\delta) = m \cdot a_u \\ F_{Y1}cos(\delta) = m \cdot a_v \end{cases} \tag{3}$$

Since the vehicle is rotating with a rotational velocity ω, the rotational acceleration must be taken into account when the total acceleration of the vehicle is computed.

The velocity of the vehicle takes the vectorial form in Equation 4.

$$V_G = u\mathbf{i} + v\mathbf{j} \tag{4}$$

Thus, the bicycle's acceleration will take the form of the derivative of the previous function, yielding Equation 5.

$$(\frac{dV_G}{dt})_{total} = (\frac{dV_G}{dt})_{linear} + \Omega \times V_G \tag{5}$$

The result of Equation 5 can be separated into Equations 6 and 8.

$$(\frac{dV_G}{dt})_{linear} = \dot{u}\mathbf{i} + \dot{v}\mathbf{j} \tag{6}$$

$$\Omega \times V_G = \omega\mathbf{k} \times (u\mathbf{i} + v\mathbf{j}) \tag{7}$$

$$\Omega \times V_G = u\omega\mathbf{j} - v\omega\mathbf{i} \tag{8}$$

Joining the results into Equation 5, the total acceleration of the system if obtained (Equation 9).

$$(\frac{dV_G}{dt})_{total} = \dot{u}\mathbf{i} + \dot{v}\mathbf{j} + u\omega\mathbf{j} - v\omega\mathbf{i} \tag{9}$$

$$(\frac{dV_G}{dt})_{total} = a_u + a_v \tag{10}$$

These accelerations can be separated depending on what axis of the vehicle they influence (Equation 10).

The accelerations affecting each axis are the following:

- Tangential acceleration : $a_v = \dot{v} + u\omega$
- Longitudinal acceleration : $a_u = \dot{u} - v\omega$

Substituting them onto Equation 3, part of the state space equations are obtained (Equation 12).

$$\begin{cases} F_{X2} - F_{Y1}sin(\delta) = m(\dot{u} - v\omega) \\ F_{Y1}cos(\delta) = m(\dot{v} + u\omega) \end{cases} \tag{11}$$

$$\begin{cases} \dot{u} = \frac{1}{m}F_{X2} - \frac{1}{m}F_{Y1}sin(\delta) - v\omega \\ \dot{v} = \frac{1}{m}F_{Y1}cos(\delta) - u\omega \end{cases} \tag{12}$$

B. *Sum of momentums*

When it comes to the momentums on the vehicle, its sum will equal the angular acceleration of the vehicle with respect to its center of mass (Equation 13).

$$\sum M = I\alpha \tag{13}$$

In Equation 14, the sum of the momentums with respect to the center of mass of the bicycle has been portrayed. Isolating $\dot{\omega}$ another state function can be obtained (Equation 15).

$$I_z\dot{\omega} = F_{Y1}cos(\delta)l_f - F_{Y2}l_r \tag{14}$$

$$\dot{\omega} = \frac{1}{I_z}(F_{Y1}cos(\delta)l_f - F_{Y2}l_r) \tag{15}$$

So far, three equations that describe the evolution of the state have been found (Equations 12, 15). These equations depend on external forces, these forces being F_{X2}, F_{Y1} and F_{Y2}. Force F_{X2} is intrinsically related to the control variables (how much thrust is applied to the pedals), whilst F_{Y1} and F_{Y2} are related to the motion of the bicycle.

These forces correspond to the friction that the wheels experience on the ground. In the case where there is slippage, which corresponds to this case, the friction coefficient is not constant through all slippage angles. The Pacejka formulation is the best approximation to the real model (Equation 16). However, its parameters must be trained with real data of the system that is to be simulated in order to obtain the best results (Such as in [6]).

$$F_y = Dsin(Carctan(Bk - E(Bk - arctan(Bk)))) \tag{16}$$

Since in this project there is no real platform with which to compare the simulated system and train parameters, a linear tyre model has been selected for the realization of this project. This model (Equation 17) only requires a constant to be set (C_f), and increases the friction of the wheel as the slip angle increases.

$$F_f = C_f \cdot \alpha_f \tag{17}$$

To apply the linear model, each tyre must be observed individually. Beginning with the front tyre (Figure 3(b)), it is easy to visualize that the sum of the steering angle with the slip angle equal the angle with which the wheel moves (Equation 18).

$$\delta + \alpha_f = \zeta_f \tag{18}$$

The angle that corresponds to the velocity can be obtained by applying and accounting for the angular velocity of the bicycle with respect to its center of mass (Equation 19).

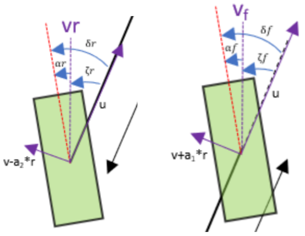
$$\zeta_f = arctan(\frac{v + \omega \cdot l_f}{u}) \tag{19}$$

These equations can be joined together to find the slipping angle (Equation 20).

$$\alpha_f = arctan(\frac{v + l_f \cdot \omega}{u}) - \delta \tag{20}$$

Afterwards, the same computation is performed on the rear wheel. The subtle difference is that in this case, $\delta = 0$. The value of the slip angle is the one depicted in Equation 21.

$$\alpha_r = \zeta_r = arctan(\frac{v - l_r \cdot \omega}{u}) \tag{21}$$



(a) Rear tyre. (b) Front tyre.

Fig. 3. Slip angle in each wheel of the bicycle. Font: [7]

The equation for each force becomes:

$$\begin{cases} F_{Y1} = C_f \cdot (arctan(\frac{v+l_f \cdot \omega}{u}) - \delta) \\ F_{Y2} = C_r \cdot arctan(\frac{v-l_r \cdot \omega}{u}) \end{cases} \tag{22}$$

These equations can be introduced to Equations 12 and 15. If the result is combined with the equations to update the position and orientation of the bicycle, the final model is obtained (Equation 23). The forces have been left in the system as variables in order to simplify the lecture.

$$\dot{X} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{u} \\ \dot{v} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} u\cos(\theta) - v\sin(\theta) \\ v\cos(\theta) + u\sin(\theta) \\ \omega \\ \frac{1}{m}F_{X2} + v\omega - \frac{1}{m}F_{Y1}\sin\delta \\ -u\omega + \frac{1}{m}(F_{Y1}\cos\delta + F_{Y2}) \\ \frac{1}{I_z}(l_f F_{Y1}\cos\delta - l_r F_{Y2}) \end{bmatrix} \tag{23}$$

The inputs are the following:

- F_{X2} [N]: The force that is applied to the rear wheel of the bicycle to propel it forward.
- δ [rad]: The angle that the handlebar is turned with respect to the longitudinal axis of the bicycle.

The parameters for the state space equations have been set as observed in Table I.

Name	Value	Unit
m	10	kg
Iz	3.3356	kg/m^2
C_f	-0.16	deg
C_f	-0.16	N/deg
l_f	1	m
l_r	1	m
u (linear model)	1	m/s

TABLE I
SYSTEM PARAMETERS

III. Proposed control design

IV. Linear model

The most usual linearization technique that is applied to these models consists in assuming that the control angles are small:

- $\sin\delta = \delta$
- $\cos\delta = 1$
- $\tan\delta = \delta$

Although with that technique all trigonometric functions are simplified, some equations remain nonlinear. In the equations for the forces (Equation 22), it can be seen that the velocity of the bicycle divides the lateral speed. If the system is simplified into assuming a constant longitudinal velocity, equations for v and ω become linear. However, equations related to the global positioning of the bicycle remain nonlinear.

Thus, the only equations that can be worked on are those related to the lateral and rotational velocities of the bicycle, which can be organized into the state space matrices in Equation 24.

$$\begin{bmatrix} v(\dot{t}) \\ \omega(\dot{t}) \end{bmatrix} = \begin{bmatrix} \frac{c_r + c_f}{mu} & \frac{c_f l_f - c_r l_r}{mu} - u \\ \frac{c_f l_f - c_r l_r}{u I_z} & \frac{c_f l_f^2 - c_r l_r^2}{u I_z} \end{bmatrix} \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} - \begin{bmatrix} \frac{c_f}{m} \\ \frac{c_f l_f}{I_z} \end{bmatrix} \delta \tag{24}$$

An unsuccessful approach was followed in order to track a reference with the linear model. Applying the Euler method, the nonlinear equations were discretized into Equation 25.

$$X(k+1) = A(k)x(k) + B\delta(k) + K(k) \tag{25}$$

$$A(k) = \begin{bmatrix} 1 & 0 & 0 & -Dt\sin(\theta^(k)) & 0 \\ 0 & 1 & 0 & Dt\cos(\theta(k)) & 0 \\ 0 & 0 & 1 & 0 & Dt \\ 0 & 0 & 0 & 1 + Dt\frac{c_r + c_f}{m \cdot u} & Dt(\frac{c_f \cdot l_f - c_r \cdot l_r}{m \cdot u} - u) \\ 0 & 0 & 0 & Dt\frac{c_f \cdot l_f - c_r \cdot l_r}{u \cdot I_z} & 1 + Dt\frac{c_f \cdot l_f^2 - c_r \cdot l_r^2}{u \cdot I_z} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{c_f}{m} \\ \frac{c_f l_f}{I_z} \end{bmatrix}$$

$$K(k) = \begin{bmatrix} u\cos\theta(k) \\ u\sin\theta(k) \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

In this system of equations, $\theta(k)$ is the yaw angle of the bicycle in time iteration k. Thus, when this system was ran in simulations, these state space matrices A, B and K had to be recomputed in each time iteration.

The system is introduced into MATLAB alongside the positioning equations in order to observe the evolution of the system as it tries to maintain a certain velocity configuration (minimizing cost function from Equation 26). The YALMIP

[11] library has been employed for this purpose and the results have been included in the Results section.

$$min \sum_{i=0}^{Hp-1} (x(i+1)-r)'Q(x(i+1)-r)+u(i)'Ru(i) \tag{26}$$

$$st,$$
$$-\frac{\pi}{4} \leq u\{i\} \leq \frac{\pi}{4}$$

A. Nonlinear model

For the nonlinear model, all the state space equations have been included in the resolution (Equations 12 and 15). They can be directly introduced to Matlab using the CasADi [12] library.

The evolution of the system has been computed for two different scenarios.

- Optimizing for v and ω: In order to compare the nonlinear and linear models, the cost function to be used is 26. The performance has been evaluated with and without noise, and the results included in the Results section.
- Optimizing for x and y: In order to track a certain trajectory, the cost function must minimize the distance from it. This model has also been evaluated with and without disturbances.

For the model that attempts to follow a trajectory, a desired velocity has been set. If the velocity is increased gradually until the vehicle cannot reach it, it will ensure that the trajectory is cleared as fast as possible. Thus, in order not to penalize the driving force, the cost function that this model uses will be the one in Equation 27.

$$min \sum_{i=0}^{Hp-1} ((x(i)-x_{ref})'Q(x(i)-x_{ref})+u(i)'Ru(i)) \tag{27}$$

$$st,$$
$$-10 \leq F_{X2} \leq 10$$
$$-\frac{\pi}{4} \leq \delta \leq \frac{\pi}{4}$$
$$0.1 \leq u$$

In the previous equation, x_{ref} represents the point of the trajectory at which the bicycle should be at each time instant. The path has been discretized into a series of points in order to introduce it into CasADi, as attempts at using a costmap with it were unsuccessful.

Lastly, another loss function was introduced in order to reduce the sudden changes on the input variables (Equation 28).

$$min \sum_{i=0}^{Hp-1} (x(i)-x_{ref})'Q(x(i)-x_{ref})+ \tag{28}$$
$$+(u(i+1)-u(i))'R(u(i+1)-u(i))$$
$$st,$$
$$-10 \leq F_{X2} \leq 10$$
$$-\frac{\pi}{4} \leq \delta \leq \frac{\pi}{4}$$
$$0.1 \leq u$$

V. Results

A. Key Performance indices (KPI)

In order to properly evaluate the performance of each model, different KPI were set.

Linear model

For the linear model the KPI are:

- Time to convergence: Time (in seconds) it takes the system to converge to the equilibrium value.
- Accumulated error: Value of the loss function at the end of the simulation, divided by the number of computed iterations.

The variables for which the linear system was evaluated were:

- Prediction horizon (Hp): 2, 5, 10, 20 seconds.
- Time increments (Dt): 0.1, 0.5, 1.0, 2.0 seconds.
- Weight on the state variables (Q): 0.8, 10, 100
- Noise: The difference in behavior of the system with and without noise.

Nonlinear model

Secondly, for the nonlinear model, the KPI to be evaluated were:

- Time per iteration: How much time it takes to compute the entire prediction horizon.
- Accumulated error: Value of the loss function at the end of the simulation, divided by the number of computed iterations.

The variables for which the nonlinear system will be evaluated were:

- Noise: The system without noise, with a Gaussian noise of standard deviation of 0.01 and a standard deviation of 0.1.
- Different loss functions: Minimizing state loss, and also smoothing the input variables (Loss functions 27 and 28).

B. Results of the Linear model

Since the linear model can only track $\dot{v}$ and $\dot{\omega}$, the system was solved for a desired input of $\delta = \pi/10$. This yields the desired state variables in Equation 29.

$$\begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix}_{\delta=\frac{\pi}{10}} = \begin{bmatrix} -4.5717 \\ 0.1571 \end{bmatrix} \tag{29}$$

Evaluation of Hp

For the evaluation of the system in different prediction horizons, Q was set to 0.8 and Dt to 1 second.

In its time response (Figure 4), it can be observed that the system is not stable if the prediction horizon is not greater than 2 seconds. Furthermore, as Hp increases, the response becomes quicker until $H_p \approx 10s$, where it can be seen that the system reaches the desired value at 35 seconds into the simulation.

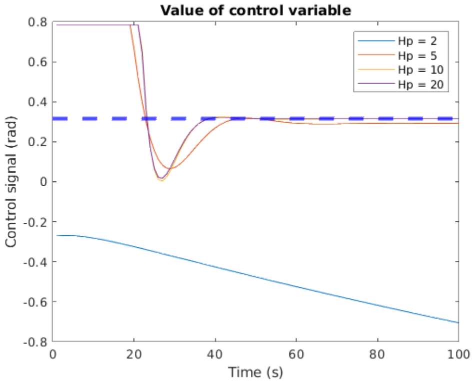


Fig. 4. Response of the linear model for different Hp.

This behavior is also observed on the values of the loss function (Table II), where the loss per iteration stops dropping once $H_p \approx 10$.

Hp (s)	2	5	10	20
Loss/it	61.8742	2.3399	2.2857	2.2857

TABLE II
LOSS PER ITERATION OF THE LINEAR MODEL FOR DIFFERENT HP

Thus, the best value for the H_p variable is $10s$, which will be kept for the next evaluations.

Evaluation of Dt

Apart from setting $H_p = 10s$, the previous value of $Q = 0.8$ has been maintained in order to find the best value for Dt.

In Figure 5, it can be faintly observed that as Dt is increased, the system tends to an equilibrium faster. Nonetheless, the difference is insignificant.

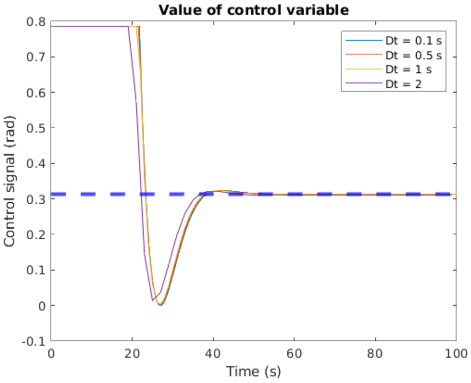


Fig. 5. Response of the linear model for different Dt.

When the values of the losses are observed (Table III), it can also be seen that if the sampling time is increased, the loss value also increases. Nonetheless, the sampling time cannot have too small of a value, as that also increases its loss.

Dt (s)	0.1	0.5	1	2
Loss	4.3464	2.2226	2.2857	2.4177

TABLE III
LOSS OF THE LINEAR MODEL FOR DIFFERENT DT

Since the best loss value was obtained for $Dt = 0.5s$, this is the value that will be used from here onwards.

Evaluation of Q

Using the previously found values for the other variables, the last variable to be iterated is Q. Increasing Q increases the importance of the error in the state variables, which in turn forces the solver to find a quicker convergence to the equilibrium point (Figure 6). This quicker convergence, however, results in the model performing stronger variations on the values of its states.

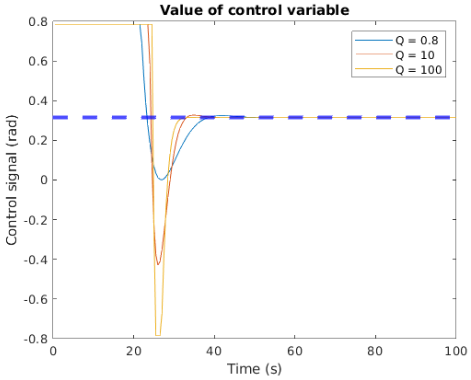


Fig. 6. Response of the linear model for different Q.

One could think that this stronger variations in the state variables would yield to an overall smaller loss value. However, since Q is included in the calculation of the loss function, increasing Q increases the value of the loss function.

Dt (s)	0.8	10	100
Loss/it	2.2226	27.3033	272.6285

TABLE IV
LOSS OF THE LINEAR MODEL FOR DIFFERENT Q

Seeing that the loss increases for higher Q, the selected Q for the rest of this project was Q = 0.8.

Linear model with noise

When the linear model comes across a disturbance, it takes much longer for the state space to converge to a value close to the desired one. In Figure 7 a noise with mean 0 and standard deviation 0.01 has been introduced to the longitudinal velocity, as if the friction of the terrain was not homogeneous. The control signal converges to a value that does not correspond to the equilibrium one, which explains why the system takes much longer to converge.

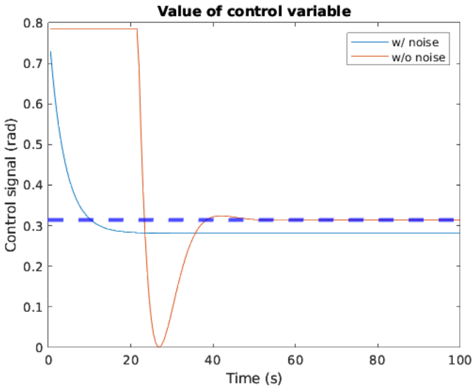


Fig. 7. Responses of the linear system without and with noise.

The difference in behavior is evident in Figure 8, where with a simulation time of 100 seconds the noiseless system converges after 30 seconds, while the model with noise does not converge by the end of simulation time.

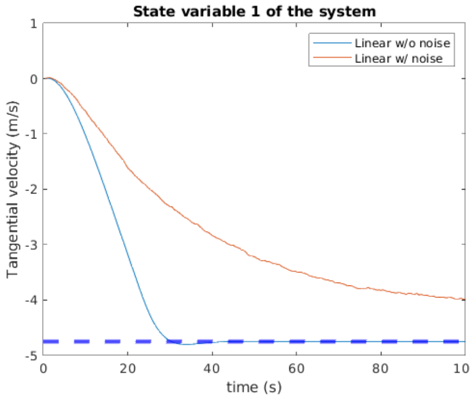


Fig. 8. Behavior of the state variables without and with noise for the linear system.

Linear model vs nonlinear model

Even though the same reference is applied to the linear and nonlinear model, responses are completely different. That is due to the nonlinear model having 4 more state variables and another input to compute with. In Figure 9 it can be seen that despite using the loss function that penalizes big changes in value for the control variables (Equation 28), its values oscillate at a high frequency.

To sum up the last two evaluations, the loss of the linear and nonlinear with or without noise has been provided in Table V. It can be seen that the nonlinear model presents a higher robustness to noise than the linear model. Whilst the linear model duplicates its loss per iteration, in the nonlinear model it decreases.

Model	NL w/o noise	NL w/ noise	L w/o noise	L w/ noise
Loss/it	3.24195	3.16692	2.2226	4.4121

TABLE V
LOSS OF THE LINEAR MODEL FOR DIFFERENT DT

Evaluation of the linear Euler-discretized model

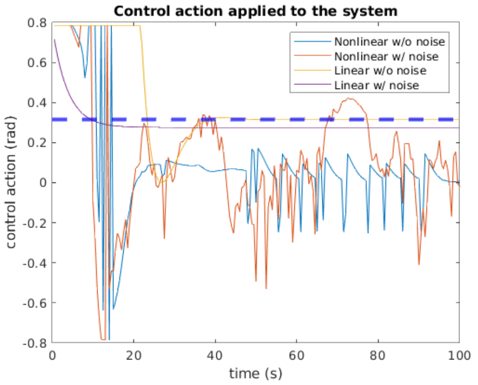


Fig. 9. Input signal δ of the linear and nonlinear systems.

Lastly, applying the Euler method (Equation 25) did not obtain good results for the linearized model, as it can be seen in Figure 10. In said Figure, the bicycle starts moving away from the desired trajectory, probably in order to increase its angular momentum. However, since it cannot accelerate to compensate for the increase in error, it strays from the desired path altogether.

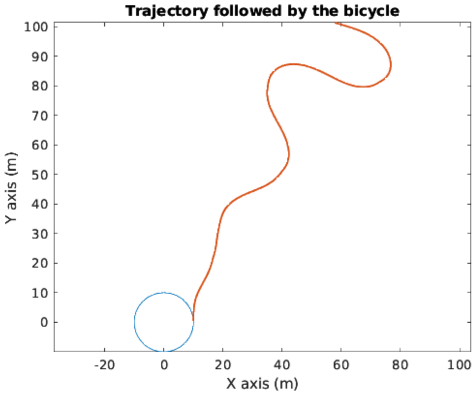


Fig. 10. Response of the discretized Euler system to discrete trajectory following.

To finalize this section, the model with the best performance was the linear model with configuration (Q = 0.8, Dt = 0.5s, Hp = 10s). However, if disturbancies are to be accounted for, the nonlinear model is a much better alternative.

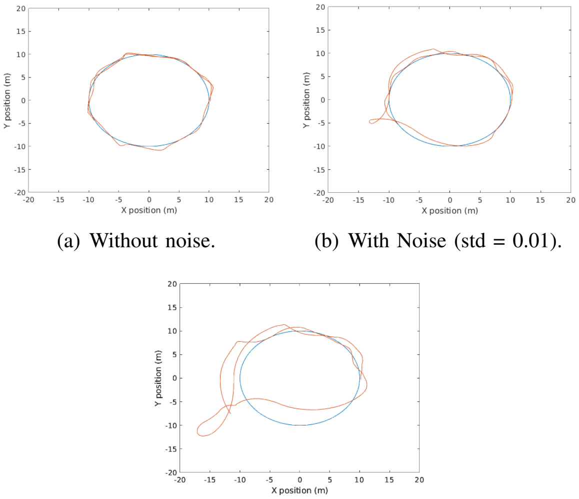
C. Results of the Nonlinear model

In this section, the capabilities of the nonlinear model to follow a trajectory were tested.

Evaluation of noise robustness

For the first test, different noises were applied to the tangential velocity and the response of the system was stored in Figure 11. It can be observed that as the noise increases the bicycle has a harder time trying to follow the given trajectory. Nonetheless, in all three scenarios the vehicle successfully stays in the desired trajectory.

The expected values are found when the losses of all three simulations are obtained (Table VI), the system has its lowest



(a) Without noise.

(b) With Noise (std = 0.01).

(c) With Noise (std = 0.1).

Fig. 11. Trajectory followed by the bicycle with different noise values applied to its tangential velocity.

loss value when there is no noise and it increases as the noise is increased.

Model	NL w/o noise	NL w/ 0.01 noise	NL w/ 0.1 noise
Loss/it	0.737692	5.57263	24.689

TABLE VI
LOSS OF THE NONLINEAR MODEL FOR DIFFERENT NOISE STD.

Furthermore, if the time it takes the model to perform all these computations is stored (Table VII), it can be observed that the model that is resistant to noise takes the double amount of time to perform the computations than the noiseless model.

Model	NL w/o noise	NL w/ 0.01 noise	NL w/ 0.1 noise
Time/it (s)	0.6404	0.1388	0.1341

TABLE VII
TIME PER ITERATION OF THE NONLINEAR MODEL FOR DIFFERENT NOISE STD.

Evaluation of Loss function

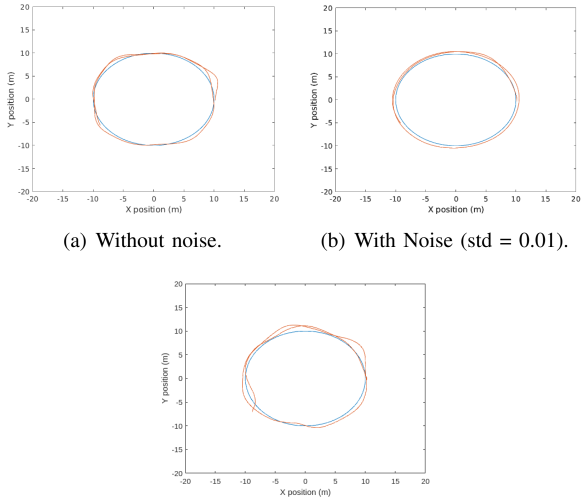
The previous section was evaluated with cost function 27, which does not take input into account when searching for the best control value. For the sake of stability, the previous experiment was run with loss function 28.

In Figure 12, the results of the simulations are portrayed. It can be seen that these results are much smoother than the ones obtained in Figure 11.

The comparison with the losses obtained with each noise (Table VIII) speak of the same results: the loss function that smoothens the controls provides much better results than the one that does not.

Model	NL w/o noise	NL w/ 0.01 noise	NL w/ 0.1 noise
Loss/it	0.81235	0.585837	1.49113

TABLE VIII
LOSS OF THE NONLINEAR MODEL FOR DIFFERENT NOISE STD FOR LOSS FUNCTION 28.



(a) Without noise.

(b) With Noise (std = 0.01).

(c) With Noise (std = 0.1).

Fig. 12. Trajectory followed by the bicycle with different noise values applied to its tangential velocity for Loss function 28.

Although one could think that adding the restriction complicates the problem, leading to longer solving time, that is not the case. By suggesting the solver to take similar values for the input in each iteration, it solves the problem much faster, as it can be observed in Table IX.

Model	NL w/o noise	NL w/ 0.01 noise	NL w/ 0.1 noise
Time/it (s)	0.4772	0.0756	0.0831

TABLE IX
TIME PER ITERATION OF THE NONLINEAR MODEL FOR DIFFERENT NOISE STD.

To finalize this section, the model with the best performance was the nonlinear model that included the increment of the control signals in its loss function.

VI. CONCLUDING REMARKS

In this document several MPC configurations were developed and tested. A first approach with a linear MPC resulted in a controller that cannot perform trajectory following. However, given a target configuration it can not only bring the system to the given configuration but also account for disturbances. Several parameters were tested in order to obtain the best configuration for this controller.

Lastly, a nonlinear MPC was developed which can perform trajectory following by performing a discretization of the path to follow. The capability of the model to follow a given path was tested for with different loss functions and noises

The final model is robust to noise and can perform trajectory following. For future work, the given trajectory to follow should be converted into a costmap. Even if this means not relying on Yalmip [11] or CasADi [12]. An example of this could be MPPI [8] (consulted during the making of this project), a nonlinear MPC implementation [9], or other studies cited in [10].

REFERENCES

[1] Cao, S.; Jin, Y.; Trautmann, T.; Liu, K. Design and Experiments of Autonomous Path Tracking Based on Dead Reckoning. Appl. Sci. 2023, 13, 317. https://doi.org/10.3390/app13010317

[2] Polack, Philip, et al. ”The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?.” 2017 IEEE intelligent vehicles symposium (IV). IEEE, 2017.

[3] Meijaard, Jaap P., and Arend L. Schwab. ”Linearized equations for an extended bicycle model.” III European Conference on Computational Mechanics: Solids, Structures and Coupled Problems in Engineering: Book of Abstracts. Springer Netherlands, 2006.

[4] Pepy, Romain, Alain Lambert, and Hugues Mounier. ”Path planning using a dynamic vehicle model.” 2006 2nd International Conference on Information & Communication Technologies. Vol. 1. IEEE, 2006.

[5] Ge, Qiang, et al. ”Numerically stable dynamic bicycle model for discrete-time control.” 2021 IEEE Intelligent Vehicles Symposium Workshops (IV Workshops). IEEE, 2021.

[6] Morera-Torres, Eduard, Carlos Ocampo-Martinez, and Fernando D. Bianchi. ”Experimental modelling and optimal torque vectoring control for 4WD vehicles.” IEEE Transactions on Vehicular Technology 71.5 (2022): 4922-4932.

[7] Michael, N. (2021, June 5). Vehicle dynamics: The dynamic bicycle model. The F1 Clan — An F1 Community created by the Fans for the Fans. https://thef1clan.com/2020/12/23/vehicle-dynamics-the-dynamic-bicycle-model/

[8] W. Wu, Z. Chen and H. Zhao, ”Model Predictive Path Integral Control based on Model Sampling,” 2019 2nd International Conference of Intelligent Robotic and Control Engineering (IRCE), Singapore, 2019, pp. 46-50, doi: 10.1109/IRCE.2019.00017.

[9] D. Kalaria et al., “Local NMPC on global optimised path for autonomous racing,” in Proc. Int. Conf. Robot. Autom. (ICRA) Workshop Opportunities Challenges Auton. Racing, 2021, pp. 1–6. [Online]. Available: https://linklab-uva.github.io/icra-autonomous-racing/contributed_papers/paper8.pdf

[10] J. Betz et al., ”Autonomous Vehicles on the Edge: A Survey on Autonomous Vehicle Racing,” in IEEE Open Journal of Intelligent Transportation Systems, vol. 3, pp. 458-488, 2022, doi: 10.1109/OJITS.2022.3181510.

[11] J. Lofberg, ”YALMIP : a toolbox for modeling and optimization in MATLAB,” 2004 IEEE International Conference on Robotics and Automation (IEEE Cat. No.04CH37508), Taipei, Taiwan, 2004, pp. 284-289, doi: 10.1109/CACSD.2004.1393890.

[12] J. A. E. Andersson, J. Gillis, G. Horn, J. B Rawlings, M. Diehl. ”CasADi, a software framework for nonlinear optimization and optimal control”. Mathematical Programming Computation (2019 Springer), vol 11, num 1, pp 1-36. doi: 10.1007/s12532-018-0139-4.