Local Path Planning with Moving Obstacle Avoidance based on adaptive MPC in ATLASCAR2

Alberto Franco
Department of Information Engineering,
Università degli Studi di Padova, Italy
alberto.franco.3@studenti.unipd.it
alberto.franco@ua.pt

***Abstract*—This report presents two different strategies for a self-driving car short-term path planning among multiple moving obstacles. The main task is to study and implement a motion planning and execution framework in order to make ATLASCAR2 coexist with other moving obstacle vehicles by avoiding collision and overtake them when necessary and possible. The proposed techniques, based on the Model Predictive Control paradigm, solve optimization problems formulated in terms of cost minimization under constraints. Simulation results demonstrate and verify the feasibility and the usefulness of methods considering different scenarios, opening space for real scenario implementation.**

I. Introduction

In robotic research, the problem of navigation is among the most important. Basically, all autonomous mobile robots need some kind of navigation abilities to perform, localization, motion planning and guidance [6]. In the present context, we focus on navigation as a process of planning a path of a mobile robot from its current position to a desired goal location, following the planned path, and avoiding any discovered obstacles along the way [7] [8]. The desired paths have to fulfill several conditions to ensure safety and feasibility of the navigation [9]. Moreover, the paths can be also defined in terms of specifications; for example, short or smooth paths are usually more desirable than long and curved ones in very dynamic environments [10]. Beyond the path planning, the navigation problem also involves reacting to changes of the environment model. Robots are required to move towards the target in a short time and avoid either static or dynamic obstacles observed by their sensors, which involves efficient path planning and valid obstacle avoidance [11] [12]. Though these two topics have been well researched, currently, there is no definitive solution to manage the navigation problem within cluttered dynamic environments. In the literature many collision avoidance methods have been developed for wheeled mobile robots; they can be grouped in four categories: graph-search based methods [1], artificial potential field based methods [2], meta-heuristic based methods [3] and mathematical optimization based methods [4]. The latter are very interesting because they offer a systematic and precise way to take vehicle dynamics and safety constraints into account and can produce optimal control inputs. This approach can be used in an closed-loop with a feedback controller for a stronger solution regarding to the Model Predictive Control (MPC). A prior research about obstacle avoidance using MPC in UGVs is described in [13] while the applications of nonlinear MPC to ground vehicles is described and evaluated in [5].

II. Context and Proposed Solution

The proposed algorithms were studied for the ATLASCAR project in which the group for Robotics and Automation at the University of Aveiro has setup and adapted a common commercial vehicle to provide a versatile framework to develop studies and research [15] [16]. Figure 1 shows the chosen platform for the second generation of ATLASCAR, which is an electric car, a Mitsubishi iMiEV with a range of a $100\,\mathrm{km}$.

Fig. 1: ATLASCAR2, based on an electric car, a Mitsubishi iMiEV, (adapted from [17])

The fact that the vehicle is electric allows to use the energy stored in the batteries, making it easier to power

the sensors installed. In fact, the ATLACAR2 is equipped with sensors, such as lidar, that measure the distance to obstacles in front and around the vehicle. The obstacles can be static, such as a large pothole, or moving, such as a moving vehicle on the same or a nearby lane. The most common maneuver from the driver is to temporarily move to another lane, drive past the obstacle, and move back to the original lane afterward. In this case, we want to design an obstacle avoidance system that moves the ATLASCAR2 around a moving obstacle in the lane using throttle and steering angle. This system uses an adaptive Model Predictive Controller that updates both the predictive model and the mixed input/output constraints at each control interval. Moreover we want to develop a lane-keeping assist system for the vehicle: it has a sensor, such as camera or laser, that measures the lateral deviation and relative yaw angle between the centerline of a lane and the ATLASCAR2; it also measures the current lane curvature and its derivative. Depending on the curve length that the sensor can view, the curvature in front of the vehicle can be calculated from the current curvature and its derivative. This system keeps the autonomous car travelling along the centerline of the lanes on the road by adjusting the front steering angle. The goal for lane keeping control is to drive both lateral deviation and relative yaw angle close to zero.

III. Theoretical background
Adaptive Model Predictive Control

Model Predictive Control is an advanced method that predicts future behavior using a linear-time-invariant (LTI) dynamic model. These predictions are not exact and a good strategy is to make MPC insensitive to prediction errors. If the plant is strongly nonlinear or its characteristics vary dramatically with time, MPC performance might become unacceptable because LTI prediction accuracy degrade [22]. A method that can address this degradation by adapting the prediction model for changing operating conditions is called Adaptive MPC: this control strategy uses a fixed model structure, but allows the model parameters to evolve with time. Ideally, whenever the controller requires a prediction, it uses a model appropriate for the current conditions. At each control interval, the adaptive MPC controller updates the plant model and nominal conditions. Once updated, the model and conditions remain constant over the prediction horizon. The plant model used as the basis for the adaptive MPC must be an LTI discrete-time, state-space model with a structure as follows:

$$\begin{aligned}\boldsymbol{x}(k+1) &= \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}_u\boldsymbol{u}(k) + \boldsymbol{B}_v\boldsymbol{v}(k) + \boldsymbol{B}_d\boldsymbol{d}(k)\\ \boldsymbol{y}(k) &= \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{D}_v\boldsymbol{v}(k) + \boldsymbol{D}_d\boldsymbol{d}(k)\end{aligned}$$

where the matrices $\boldsymbol{A}$, $\boldsymbol{B}_u$, $\boldsymbol{B}_v$, $\boldsymbol{B}_d$, $\boldsymbol{C}$, $\boldsymbol{D}_v$ and $\boldsymbol{D}_d$ can vary with time. The other parameters in the previous expression are:

- k is the time index/current control interval;
- $\boldsymbol{x}$ are the plant model states;
- $\boldsymbol{u}$ are the manipulated inputs that can be adjusted by the MPC controller;
- $\boldsymbol{v}$ are the measured disturbance inputs;
- $\boldsymbol{d}$ are the unmeasured disturbance inputs;
- $\boldsymbol{y}$ are the plant outputs, including both measured (necessary at least one) and unmeasured.

In the adaptive MPC control, there are additional requirements for the plant model, like the sample time T_s that has to be constant and identical to the MPC control interval. This control strategy prohibits direct feed-through from any manipulated variable to any plant output. Thus, $\boldsymbol{D_v} = \boldsymbol{0}$ in the above model. A traditional MPC controller includes a nominal operating point at which the plant model applies, such as the condition at which you linearize a nonlinear model to obtain the LTI approximation (equilibrium, reference trajectory and the most updated value) [22]. In adaptive MPC, as time evolves it should update the nominal operating point to be consistent with the updated plant model. It is possible to rewrite the plant model in terms of deviations from the nominal conditions as follows:

$$\begin{aligned}\boldsymbol{x}(k+1) &= \overline{\boldsymbol{x}} + \boldsymbol{A}(\boldsymbol{x}(k) - \overline{\boldsymbol{x}}) + \boldsymbol{B}(\boldsymbol{u}_t(k) - \overline{\boldsymbol{u}}_t) + \overline{\Delta\boldsymbol{x}}\\ \boldsymbol{y}(k) &= \overline{\boldsymbol{y}} + \boldsymbol{C}(\boldsymbol{x}(k) - \overline{\boldsymbol{x}}) + \boldsymbol{D}(\boldsymbol{u}_t(k) - \overline{\boldsymbol{u}}_t)\end{aligned}$$

where the matrices $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{D}$ are updated with respect to time. The other parameters in the previous structure are:

- $\boldsymbol{u}_t$ is the combined plant input variable, comprising $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{d}$ variables defined earlier;
- $\overline{\boldsymbol{x}}$ are the nominal states;
- $\overline{\Delta\boldsymbol{x}}$ are the nominal state increments;
- $\overline{\boldsymbol{u}_t}$ and $\overline{\boldsymbol{y}}$ are the nominal inputs and outputs.

The adaptive MPC uses a Kalman filter to update its controller states which include the plant, the disturbance and measurement noise model states. In particular this filter is linear-time-varying (LTV) because adjusts the gains at each control interval to maintain consistency with the updated plant model.

IV. Moving Obstacle Avoidance

A. Problem Formulation

The collision avoidance problem is very dependent on the vehicle modeling since it is a requirement for adaptive MPC law design. Figure 2 illustrates a typical scenario of overtaking of a moving obstacle.

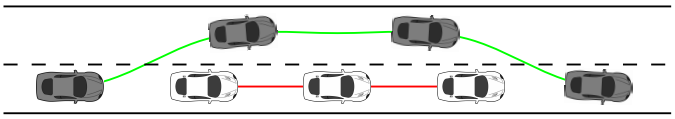


Fig. 2: Problem description of collision avoidance on a road with only two lanes.

The model used in this report should take into account the kinematic and dynamic aspects of the vehicle. Here, we present a non linear mathematical model of a vehicle used for the development of a collision avoidance system. The model has four states and two inputs:

$$\boldsymbol{x} = \begin{bmatrix} x & y & \theta & v \end{bmatrix}^\intercal, \qquad \boldsymbol{u} = \begin{bmatrix} T & \delta \end{bmatrix}^\intercal$$

where (x, y) are the global coordinates of the contact point between the rear wheel and the ground, θ is the heading angle of the car body with respect to the x-axis and v is the linear speed of the car (positive). The manipulated variables are T the throttle (positive when accelerating/negative when braking) and δ the steering angle of the front wheel with respect to the vehicle (0 when aligned with car, counterclockwise positive). Figure 3 illustrates the applied nonlinear bicycle model and the related parameters.

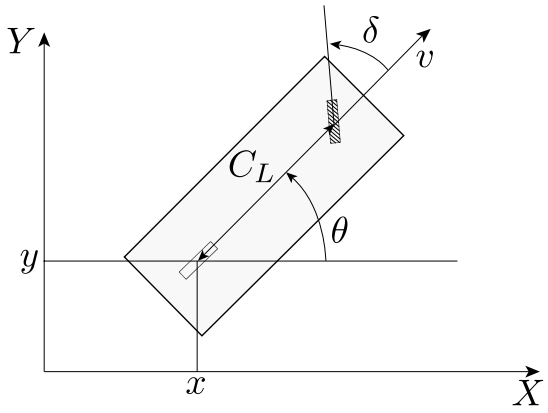


Fig. 3: Bicycle model of a car, (adapted from [18]).

The ATLASCAR2 can be modeled using the non-linear kinematic bicycle model described by the following equations of motion [9] [20]:

$$\begin{cases} \dot{x} = v\cos(\theta) \\ \dot{y} = v\sin(\theta) \\ \dot{\theta} = \dfrac{v}{C_L}\tan(\delta) \\ \dot{v} = 0.5 \cdot T \end{cases} \implies \begin{matrix} \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \\ \boldsymbol{y} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) \end{matrix}$$

where C_L is the car length. In order to simplify the model, it is assumed that only the front wheel can be steered. Moreover, in this paper it is assumed that the ATLASCAR2 does not slip, so any slippage is thus considered as an external disturbance. Under this assumption, the slip angle is zero, meaning that the velocity is directed along the heading of the vehicle. In order to use MPC, the state space model needs to be linearized with a first order approximation and also re-written in a more compact form:

$$\begin{matrix} \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \\ \boldsymbol{y} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) \end{matrix} \implies \begin{matrix} \dot{\boldsymbol{x}} = \boldsymbol{A}_c\boldsymbol{x} + \boldsymbol{B}_c\boldsymbol{u} \\ \boldsymbol{y} = \boldsymbol{C}_c\boldsymbol{x} + \boldsymbol{D}_c\boldsymbol{u} \end{matrix} \tag{1}$$

where the matrices $\boldsymbol{A}_c$, $\boldsymbol{B}_c$, $\boldsymbol{C}_c$ and $\boldsymbol{D}_c$ are obtained as follows:

$$\boldsymbol{A}_c = \frac{\partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}} = \begin{bmatrix} 0 & 0 & -v\sin(\theta) & \cos(\theta) \\ 0 & 0 & v\cos(\theta) & \sin(\theta) \\ 0 & 0 & 0 & \tan(\delta)/C_L \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\boldsymbol{B}_c = \frac{\partial \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{u}} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & \dfrac{v}{C_L}\big(\tan(\delta)^2 + 1\big) \\ 0.5 & 0 \end{bmatrix},$$

$$\boldsymbol{C}_c = \frac{\partial \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{u}} = \mathbf{I}_4, \qquad \boldsymbol{D}_c = \frac{\partial \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{u}} = \mathbf{0}_{4\times 2}.$$

The simple linearized approximation of the system to describe the dynamics of the ATLASCAR2 will be evaluated at the operating conditions. Note also that the system we are considering is a linear state-space model whose dynamics vary as a function of certain time-varying parameters. The system to be controlled is usually modeled by a discrete state-space model in the MPC literature. Therefore, (1) is transformed into a discrete state-space model to be used by the Model Predictive Controller:

$$\begin{matrix} \dot{\boldsymbol{x}} = \boldsymbol{A}_c\boldsymbol{x} + \boldsymbol{B}_c\boldsymbol{u} \\ \boldsymbol{y} = \boldsymbol{C}_c\boldsymbol{x} + \boldsymbol{D}_c\boldsymbol{u} \end{matrix} \Rightarrow \begin{matrix} \boldsymbol{x}(k+1) = \boldsymbol{A}_d\boldsymbol{x}(k) + \boldsymbol{B}_d\boldsymbol{u}(k) \\ \boldsymbol{y}(k) = \boldsymbol{C}_d\boldsymbol{x}(k) + \boldsymbol{D}_d\boldsymbol{u}(k) \end{matrix}$$

where $\boldsymbol{A}_d$ and $\boldsymbol{B}_d$ are the state and control matrices for the discrete state-space equation, respectively, which can be calculated with the Euler method as

$$\boldsymbol{A}_d = e^{\boldsymbol{A}_c T_s}, \qquad \boldsymbol{B}_d = \int_{kT_s}^{(k+1)T_s} e^{\boldsymbol{A}_c[(k+1)T_s - \eta]} \boldsymbol{B}_c d\eta$$

where T_s is the sampling interval for the discrete state-space model. The matrices $\boldsymbol{C}_d$ and $\boldsymbol{D}_d$ are equivalent to those in the continuous case. For simplicity, we assume that all the states are measurable and the ATLASCAR2 drives east with a constant speed at the nominal operating point. In the scenario that we are going to consider, the road is straight and our vehicle stays in the middle of the center lane when not passing. Without losing generality, the ATLASCAR2 passes an obstacle both to the right and to the left lane depending on where it is placed on the road. We create also a safe zone around the obstacles so that the vehicle does not get too close to the obstacle when passing it.

B. Design of Adaptive Model Predictive Control

We designed a Model Predictive Controller that can make the ATLASCAR2 maintain a desired velocity and stay in the middle of center lane. We used an Adaptive MPC controller because it handles the nonlinear vehicle dynamics more effectively than a traditional MPC controller; in fact, the latter uses a constant plant model but the former allows us to provide a new plant model at each control interval. Because the new model describes the plant dynamics more accurately at the new operating condition, an adaptive MPC controller performs better than a traditional MPC controller. In practice, at each control interval, the adaptive MPC controller updates the plant model and the nominal conditions. Once updated, the model and the conditions remain constant over the prediction horizon. In motion planning that uses adaptive MPC, it is common to formulate the constrained control problem as a real-time optimization problem subject to hard constraints on plant variables and soft constraints on outputs; at the beginning, we specified the constraints for the manipulated variables: to prevent the ATLASCAR2 from accelerating or decelerating too quickly, we added an hard constraint on the throttle rate of change and another one on the steering angle rate of change. We used an approach that takes advantage of the ability of MPC to handle constraint explicitly. Figure 4 shows a conditional state machine designed for higher-level behavior planning.

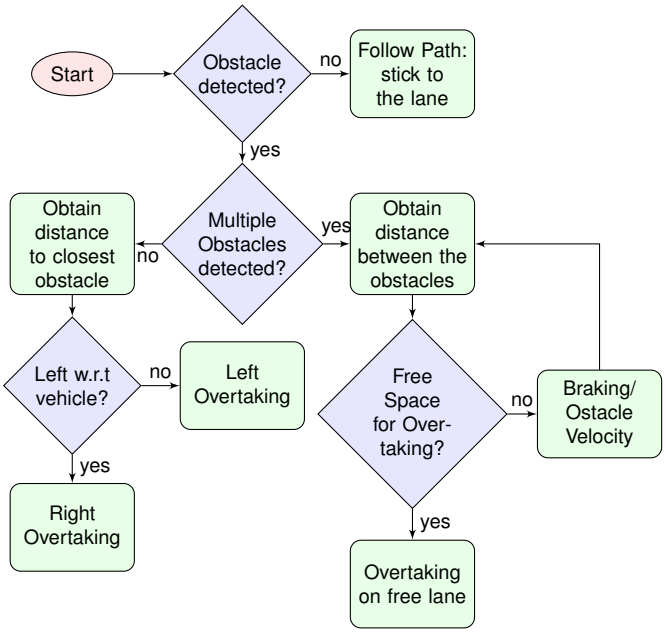


Fig. 4: Behaviour planning conditional flowchart.

When an obstacle is detected, it defines an area on the road (in terms of constraints) that the ATLASCAR2 must not enter during the prediction horizon. At the next control interval the area is redefined based on the new positions of the vehicle and the obstacle until passing is completed. To define the area to avoid, we used the following mixed Input/Output constraints:

$$\boldsymbol{E}\boldsymbol{u} + \boldsymbol{F}\boldsymbol{y} \leq \boldsymbol{G}$$

where $\boldsymbol{u}$ and $\boldsymbol{y}$ are respectively the manipulated variable vector and the output variable vector, while $\boldsymbol{E}, \boldsymbol{F}, \boldsymbol{G}$ are the constraint matrices that can be updated when the controller is running. Five constraints were defined:

1) upper bound on the y-coordinate (left boundary of the road);
2) lower bound on the y-coordinate (right boundary of the road);
3) constraint for obstacle avoidance; although no obstacle is considered in the nominal condition, we must add this virtual constraint here because we cannot change the dimensions of the constraint matrices at run time;
4) upper bound on the x-coordinate (position of the closest obstacle);
5) lower bound on the x-coordinate (position of the ATLASCAR2).

The matrices for the above inequality are the following:

$$\boldsymbol{E} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \boldsymbol{F} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ cS & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix}, \boldsymbol{G} = \begin{bmatrix} W/2 \\ W/2 \\ -cI \\ x_{\max} \\ x_{\min} \end{bmatrix}$$

where W is the width of the road, cI and cS are the required parameters such that the ATLASCAR2 must be above the line formed from the vehicle to safe zone corner for left/right passing and finally $x_{\max}$ represents the position of the closest obstacle in the case there is not free space for the overtaking (otherwise $x_{\max} = +\infty$) while $x_{\min}$ depicts the location of our vehicle. Figure 5 illustrates the constraints that are computed at each T_s in the case of a left overtaking.

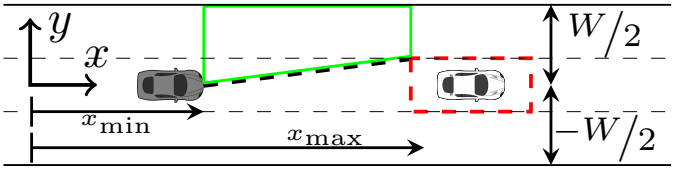


Fig. 5: Constraints in the case of left overtaking.

C. Simulation Results

The performances of the proposed adaptive MPC based vehicle control method are demonstrated in three simulation examples. We tried to choose parameters that were as close as possible to a real situation: the sampling time used in the discretization of the system is $T_s = 0.02\,\mathrm{s}$ while the values of the prediction and the control horizon are respectively $p_H = 25$ and $c_H = 5$. In all simulations, the distance between the front and rear axles is $C_L = 5\,\mathrm{m}$ and the width of the vehicle is $C_W = 2\,\mathrm{m}$. The saturation ranges of the control inputs are: the steering angle lies in $\left[-\frac{\pi}{30}, +\frac{\pi}{30}\right]$ rad/s while in order to prevent the ATLASCAR2 from accelerating or decelerating too quickly, we impose an hard constraint of $2.5\,\mathrm{m/s^2}$ on the throttle rate of change. Moreover we are using a constant reference signal for the velocity of $v = 20\,\mathrm{m/s}$ ($\approx 72\,\mathrm{km/h}$). Blue paths in Figures 6, 7, 8 are known only at the end of the simulations.

1) One Moving Obstacle - Right Overtaking: In this first simulation (Fig. 6) the ATLASCAR2 drives in the middle of the center lane while the road is completely free and when there is an obstacle, the vehicle passes it only using the right lane (the same simulation can be launched so that the car goes over to the left fast lane). In other words if the ATLASCAR2 is already in the

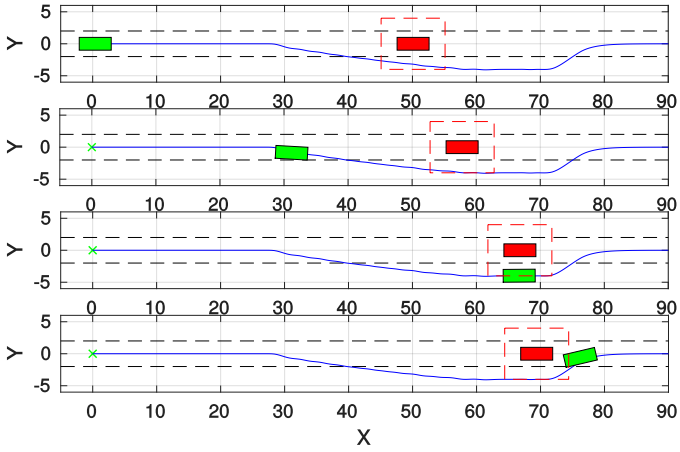


Fig. 6: Simulation of right overtaking with one moving obstacle that moves in the same direction as the vehicle.

Algorithm 1 Right Overtaking if an obstacle is detected

function RIGHTOVERTAKING(car, obstacle, road)
  $x_{\min} \leftarrow \mathrm{car}X$, $x_{\max} \leftarrow +\infty$;
  obsYrr = obstacle.RearRightSafeY;
  obsXrr = obstacle.RearRightSafeX;
  if ATLASCAR2 is behind the obstacle **then**
    if ATLASCAR2 is in the adjacent lane **then**
      $cS \leftarrow 0$; $cI \leftarrow$ obsYrr;
    else
      $cS \leftarrow \tan(\mathrm{atan2}(\frac{\mathrm{obsYrr} - \mathrm{carY}}{\mathrm{obsXrr} - \mathrm{carX}}, 1))$;
      $cI \leftarrow \mathrm{obsYrr} - cS * \mathrm{obsXrr}$;
    end if
  else
    if ATLASCAR2 is parallel to the obstacle **then**
      $cS \leftarrow 0$; $cI \leftarrow$ obsYrr;
    else
      $cS \leftarrow 0$; $cI \leftarrow W/2$;
    end if
  end if
  return $x_{\min}$, $x_{\max}$, cI, cS
end function

adjacent lane, it uses the safety zone as the constraint, otherwise the vehicle must be above the line formed from the ATLASCAR2 to safe zone corner for right passing. If the vehicle is parallel to the obstacle, it uses the safety zone as the constraint and finally if it has passed the obstacle, it uses the inactive constraint to go back to the center lane. Algorithm 1 summarizes the main steps to compute custom constraints for the obstacle; when the vehicle detects the obstacle, the constraints are computed. In this simulation only the first three constraints are necessary because there is space for the overtaking without braking (the fourth and fifth constraints don't change).

2) Multiple Moving Obstacles: For a second test, additional obstacles were added to make the scenario more complex (Fig. 7).

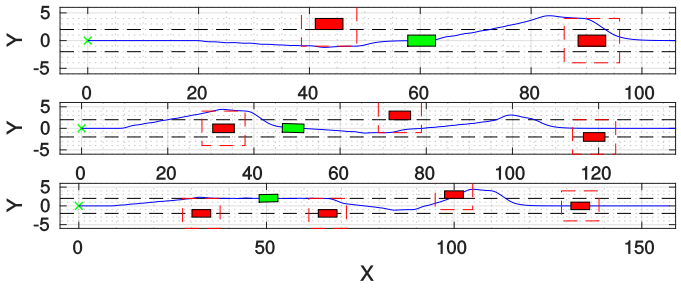


Fig. 7: Simulations of overtaking with $N = 2, 3, 4$ moving obstacles that drive in the opposite direction with respect to the ATLASCAR2.

The vehicle is capable of overtaking the obstacles on the right or left depending on their positions with respect to the road. If the y-coordinate of the closest obstacle is greater than 0, then the vehicle overtakes to the right, otherwise the overtaking takes place on the left lane. We also hypothesized that the obstacles move at different speeds but that they are initially at a common distance; they can have a uniform motion or a uniformely accelerated motion. In case two obstacles are too close during the simulation and their distance is less than the detection range, which is $30\,\mathrm{m}$, the ATLASCAR2 perceives the objects as a single entity and adapts to the situation. The same test can be done with the obstacles that drive in the same direction as the vehicle but to better assess and demonstrate results we decided to show a very unusual scenario.

3) Vehicle Braking and Obstacles Overtaking: Finally we have improved the code related to the mixed Input/Output constraints so that in the case there are 3 obstacles that block the road and drive at a lower speed than the ATLASCAR2, the velocity of the vehicle decreases in order to prevent the collision. Figure 8 depicts a simulation in which there are 3 obstacles at the same x-coordinate. Two obstacles have a constant speed of $8\,\mathrm{m/s}$ while the one on the left lane of $15\,\mathrm{m/s}$. In this particular case it is essential to consider the fourth and the fifth restriction in order to allow the ATLASCAR2 to slow down without colliding with the cars in front. The fifth constraint is simply the position of the ATLASCAR2 that updates at each interval, while the fourth constraint, until there is a free lane, is the position of the closest obstacle (both the positions are with respect to the global reference frame). In the calculation of the fourth constraint is necessary to consider the speed at which the vehicle and the obstacle are moving in order to keep a safe distance. In particular three parameters have been identified as the key factors computing the safe distance between cars:

- detection range of the ATLASCAR2;
- velocities of the vehicle and the obstacle;
- distance between the cars.

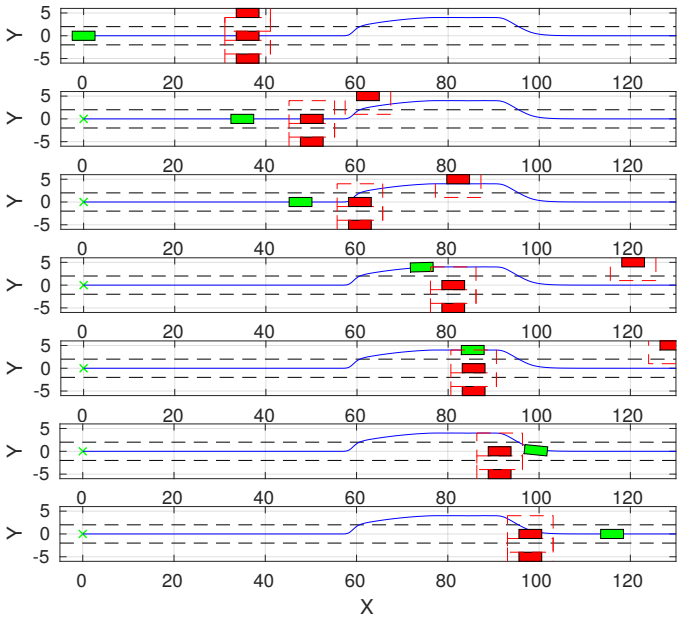


Fig. 8: Simulation of braking and overtaking obstacles.

At the beginning, the vehicle moves with the reference velocity of $v = 20\,\mathrm{m/s}$. When the ATLASCAR2 detects all the other cars on the road, checks if there is a free lane. In the first part of the simulation, the ATLASCAR2 brakes because there is not enough space for overtaking the cars as shown in Fig. 9a. A collision would happen if the vehicle continues to follow the initially planned path with the reference velocity. It is possible to notice that the speed decreases because the applied throttle is negative, so a consistent deceleration is set after $\approx 1.5\,\mathrm{s}$ as depicted in Fig. 9b. The velocity of the ATLASCAR2 for $\approx 2\,\mathrm{s}$ adapts to that of the closest obstacle. After a few seconds the fastest car moves and makes available the left lane for overtaking. Dramatic changes of steering angle in early stage are observed in Fig. 9c and consequently also on the heading angle in Fig. 9d. Then, the ATLASCAR2 returns to the reference velocity during the overtaking of the two obstacles (the applied throttle after $\approx 5\,\mathrm{s}$ is positive). It is seen that the ATLASCAR2 avoids the obstacles and returns to the road center line with a low overshoot.

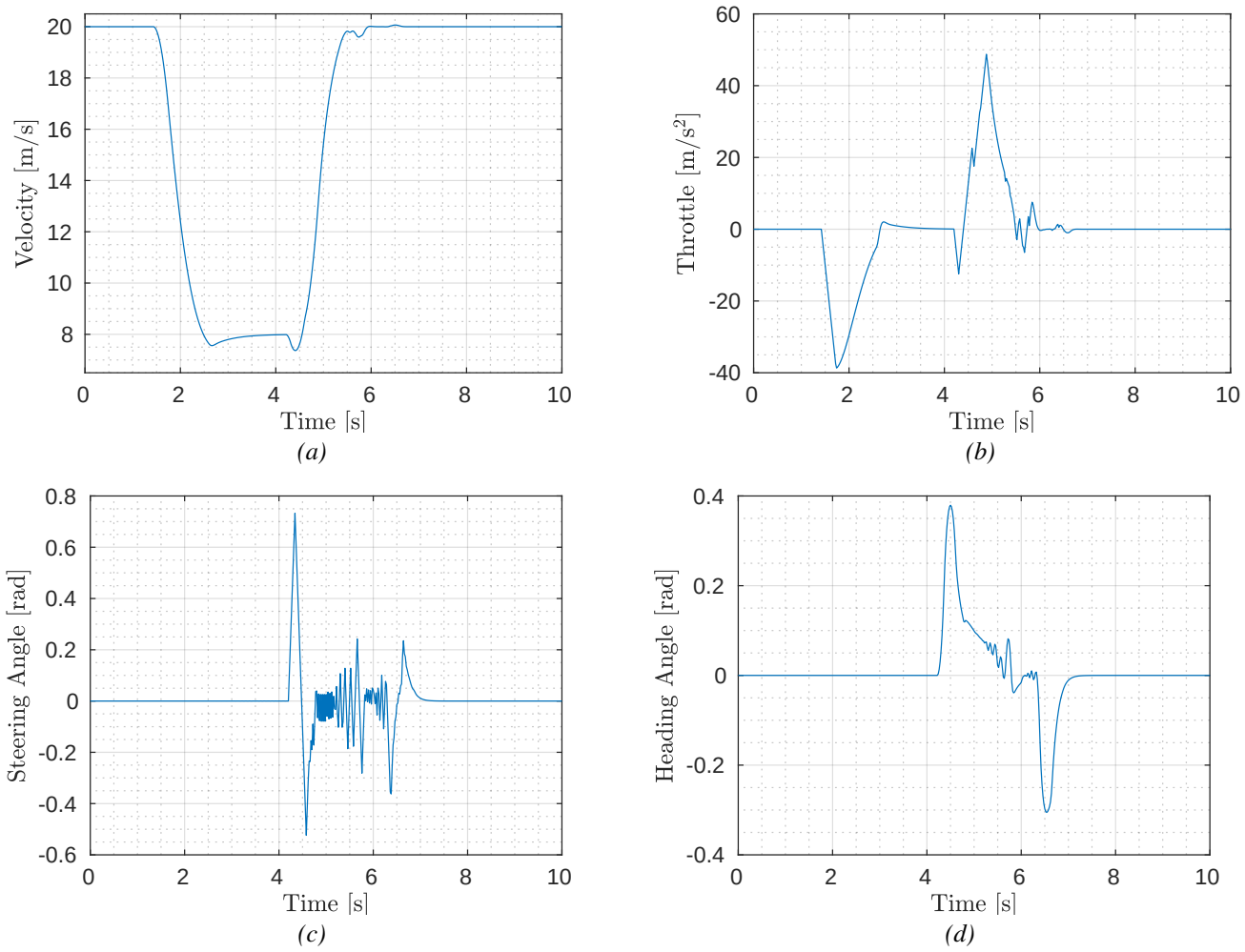


Fig. 9: Time signals of the ATLASCAR2 in the simulation of braking and overtaking in the situation illustrated in Fig. 8.

V. Lane Following System

A. Problem Formulation

A lane-following system is a control system that keeps the vehicle traveling along the centerline of a highway lane, while maintaining a user-set velocity. Figure 10 illustrates a typical lane following scenario. In a classic lane keeping assist, it is assumed that the longitudinal velocity is constant [14]. This restriction is relaxed in this model because the longitudinal acceleration varies in this MIMO control system. This lane-following system manipulates both the longitudinal acceleration and the front steering angle of the vehicle to keep the lateral deviation and the relative yaw angle small and the longitudinal velocity close to a driver set velocity. If these two goals cannot be met at the same moment, the system tries to balance them. The model that we are considering contains many parameters. The first fundamental block describes the vehicle dynamics: we have applied the bicycle model of lateral vehicle dynamics and approximate the longitudinal dynamics using a time constant obtaining a linear model.

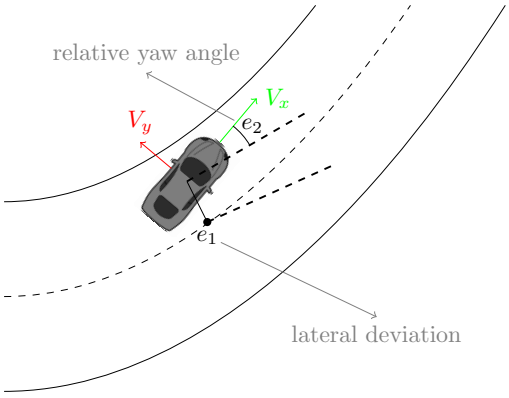


Fig. 10: Problem description of a lane following system.

1) Longitudinal dynamics: We can use the following state space to describe the longitudinal model:

$$\begin{aligned} \dot{\boldsymbol{x}}_{\text{lon}} &= \boldsymbol{A_m}\boldsymbol{x}_{\text{lon}} + \boldsymbol{B_m}\boldsymbol{u}_{\text{lon}} \\ \boldsymbol{y}_{\text{lon}} &= \boldsymbol{C_m}\boldsymbol{x}_{\text{lon}} + \boldsymbol{D_m}\boldsymbol{u}_{\text{lon}} \end{aligned} \tag{2}$$

where the input is the acceleration and the states are the longitudinal velocity and the actual acceleration which is also the only output of this system.

$$\boldsymbol{x}_{\text{lon}} = \begin{bmatrix} \dot{V}_x & V_x \end{bmatrix}^T, \qquad \boldsymbol{u}_{\text{lon}} = a$$

and

$$\boldsymbol{A_m} = \begin{bmatrix} -\frac{1}{\tau} & 0 \\ 1 & 0 \end{bmatrix}, \quad \boldsymbol{B_m} = \begin{bmatrix} \frac{1}{\tau} \\ 0 \end{bmatrix},$$

$$\boldsymbol{C_m} = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad \boldsymbol{D_m} = 0.$$

where τ is a time constant [21]; in practice we are considering a second order transfer function like in [24].

2) Lateral dynamics: Local function: we have a continuous vehicle lateral model from parameters obtained by simplifying the one in [23]:

$$\begin{aligned} \dot{\boldsymbol{x}}_{\text{lat}} &= \boldsymbol{A}_g\boldsymbol{x}_{\text{lat}} + \boldsymbol{B}_g\boldsymbol{u}_{\text{lat}} \\ \boldsymbol{y}_{\text{lat}} &= \boldsymbol{C}_g\boldsymbol{x}_{\text{lat}} + \boldsymbol{D}_g\boldsymbol{u}_{\text{lat}} \end{aligned} \tag{3}$$

where the input is the steering angle in radians, and the outputs are the lateral velocity in meters per second and yaw angle rate in radians per second:

$$\boldsymbol{x}_{\text{lat}} = \begin{bmatrix} V_y & \dot{\psi} \end{bmatrix}^T \qquad \boldsymbol{u}_{\text{lat}} = \delta$$

and

$$\boldsymbol{A}_g = \begin{bmatrix} -\frac{2C_F+2C_R}{mV_x} & -\frac{2C_Fl_F-2C_Rl_R}{mV_x} - V_x \\ -\frac{2C_Fl_F-2C_Rl_R}{I_ZV_x} & -\frac{2C_Fl_F^2+2C_Rl_R^2}{I_ZV_x} \end{bmatrix},$$

$$\boldsymbol{B}_g = \begin{bmatrix} 2C_F/m \\ 2C_Fl_F/I_Z \end{bmatrix}, \quad \boldsymbol{C}_g = \boldsymbol{I}_2, \quad \boldsymbol{D}_g = \boldsymbol{0}_{2\times 1}.$$

The parameters in the previous matrices are:

- V_x is the longitudinal velocity of the car;
- m is the total mass parameter;
- I_Z is the yaw moment of inertia parameter;
- l_F and l_R are the longitudinal distances from center of gravity to front and rear tires parameters;
- C_F and C_R are the cornering stiffnesses of front and rear tires parameters.

The goal for the driver steering model is to keep the vehicle in its lane and follow the curved road by controlling the front steering angle . This goal is achieved by driving the yaw angle error $e_2 = \psi - \psi_{\text{des}}$ and lateral displacement error e_1 to zero ($\dot{e}_1 = V_x e_2 + V_y$). We can incorporate these two paramenters in the augmented model:

$$\begin{aligned} \dot{\boldsymbol{x}}_{\text{aug}} &= \boldsymbol{A}_a\boldsymbol{x}_{\text{aug}} + \boldsymbol{B}_a\boldsymbol{u}_{\text{aug}} \\ \boldsymbol{y}_{\text{aug}} &= \boldsymbol{C}_a\boldsymbol{x}_{\text{aug}} + \boldsymbol{D}_a\boldsymbol{u}_{\text{aug}} \end{aligned} \tag{4}$$

where

$$\boldsymbol{x}_{\text{aug}} = \begin{bmatrix} V_y & \dot{\psi} & e_1 & e_2 \end{bmatrix}^T, \qquad \boldsymbol{u}_{\text{aug}} = \begin{bmatrix} \delta & \dot{\psi}_{\text{des}} \end{bmatrix}^T$$

and

$$\boldsymbol{A}_a = \begin{bmatrix} \boldsymbol{A}_g & \boldsymbol{0}_{2\times 2} \\ \boldsymbol{I}_2 & \begin{matrix} 0 & V_x \\ 0 & 0 \end{matrix} \end{bmatrix}, \quad \boldsymbol{B}_a = \begin{bmatrix} \boldsymbol{B}_g & \boldsymbol{0}_{2\times 1} \\ \begin{matrix} 0 \\ 0 \end{matrix} & \begin{matrix} 0 \\ -1 \end{matrix} \end{bmatrix},$$

$$C_a = \begin{bmatrix} \boldsymbol{0}_{2\times 2} & \boldsymbol{I}_2 \end{bmatrix}, \quad D_a = \boldsymbol{0}_{2\times 2}.$$

Combining (2) with (4) yields the state-space model that characterizes the Model Predictive Controller:

$$\begin{aligned} \dot{\boldsymbol{x}}_{\text{tot}} &= \boldsymbol{A}_f\boldsymbol{x}_{\text{tot}} + \boldsymbol{B}_f\boldsymbol{u}_{\text{tot}} \\ \boldsymbol{y}_{\text{tot}} &= \boldsymbol{C}_f\boldsymbol{x}_{\text{tot}} + \boldsymbol{D}_f\boldsymbol{u}_{\text{tot}} \end{aligned} \tag{5}$$

where

$$\boldsymbol{x}_{\text{tot}} = \begin{bmatrix} \dot{V}_x \\ V_x \\ V_y \\ \dot{\psi} \\ e_1 \\ e_2 \end{bmatrix}, \qquad \boldsymbol{u}_{\text{tot}} = \begin{bmatrix} a \\ \delta \\ \dot{\psi}_{\text{des}} \end{bmatrix}$$

and

$$\boldsymbol{A}_f = \begin{bmatrix} \boldsymbol{A}_m & \boldsymbol{0}_{2\times 4} \\ \boldsymbol{0}_{4\times 2} & \boldsymbol{A}_a \end{bmatrix}, \quad \boldsymbol{B}_f = \begin{bmatrix} \boldsymbol{B}_m & \boldsymbol{0}_{2\times 2} \\ \boldsymbol{0}_{4\times 1} & \boldsymbol{B}_a \end{bmatrix},$$

$$\boldsymbol{C}_f = \begin{bmatrix} \boldsymbol{C}_m & \boldsymbol{0}_{1\times 4} \\ \boldsymbol{0}_{2\times 2} & \boldsymbol{C}_a \end{bmatrix}, \quad \boldsymbol{D}_f = \boldsymbol{0}_{3\times 3}.$$

However the system to be controlled is usually modeled by a linear discrete state-space model:

$$\begin{aligned} \boldsymbol{x}_{\text{tot}}(k+1) &= \boldsymbol{A}\boldsymbol{x}_{\text{tot}}(k) + \boldsymbol{B}\boldsymbol{u}_{\text{tot}}(k) \\ \boldsymbol{y}_{\text{tot}}(k) &= \boldsymbol{C}\boldsymbol{x}_{\text{tot}}(k) + \boldsymbol{D}\boldsymbol{u}_{\text{tot}}(k) \end{aligned}$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ are the state and control matrices for the discrete state-space equation, respectively, which can be calculated, also in this case, with the Euler method as:

$$\boldsymbol{A} = e^{\boldsymbol{A}_f T_s}, \qquad \boldsymbol{B} = \int_{kT_s}^{(k+1)T_s} e^{\boldsymbol{A}_f[(k+1)T_s - \eta]}\boldsymbol{B}_f d\eta$$

where T_s is the sampling interval for the discrete state-space model. The matrices $\boldsymbol{C}$ and $\boldsymbol{D}$ are equivalent to those in the continuous case.

B. Design of Adaptive Model Predictive Control

We created an Adaptive MPC controller with a prediction model that has six states, three outputs (longitudinal velocity, lateral deviation, relative yaw angle), and two manipulated signals (acceleration and steering). The objective of the trajectory planning along specified path can be described as follows: given a path which the vehicle is expected to follow design a trajectory of a car-vehicle configuration. In order to do this, according with [25] it is possible to derive the road curvature and its derivative. The product of the road curvature and the longitudinal velocity is modeled as a measured disturbance. We have set the constraints for manipulated variables and the scale factors. Moreover we have specified the weights in the standard MPC cost function. The third output, yaw angle, is allowed to float because there are only two manipulated variables to make it a square system. In this controller, there is no steady-state error in the yaw angle as long as the second output, lateral deviation, reaches 0 at steady state. Finally we have also penalized acceleration change more for smooth driving experience. This controller uses a linear model for the vehicle dynamics and updates the model online as the longitudinal velocity varies.

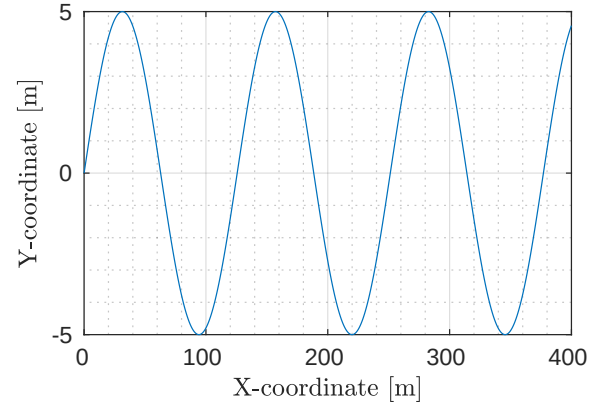
C. Simulation Results

The proposed adaptive MPC algorithm is designed in the MATLAB/Simulink and validated through simulation. The objective of this test is to evaluate the behavior of the proposed control strategy in critical situations. Table 3 shows the parameters used in the lane following simulation.

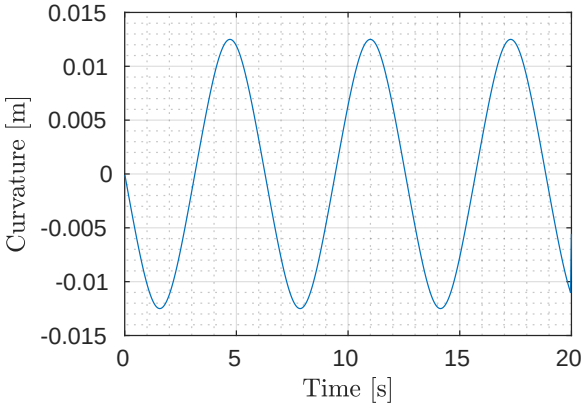
Parameters	Values
m	$1575\,\mathrm{kg}$
I_z	$2875\,\mathrm{kgm}^2$
l_F	$1.2\,\mathrm{m}$
l_R	$1.6\,\mathrm{m}$
C_F	$19\,000\,\mathrm{N/rad}$
C_R	$33\,000\,\mathrm{N/rad}$
τ	0.2
V_0	$15\,\mathrm{m/s}$
V_{set}	$20\,\mathrm{m/s}$
T_s	$0.02\,\mathrm{s}$

Figures 11a and 11b show the desired path that the car must follow and its curvature, where the former is described in terms of the lateral position Y_{ref} as function of the longitudinal position X_{ref} and the latter is derived according with [25]. The ATLASCAR2 is controlled to follow a sinusoidal trajectory which is given as follows:

$$X_{\text{ref}} = V_x \cdot t, \quad Y_{\text{ref}} = 5\sin(X_{\text{ref}}/20) \quad \text{with} \quad t \in [0, 20]\text{s}$$



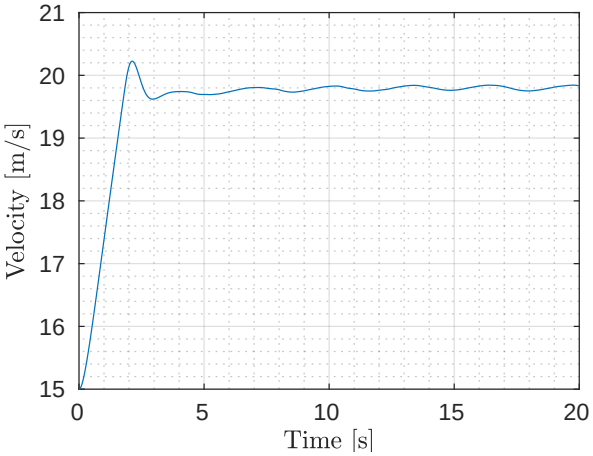
(a)



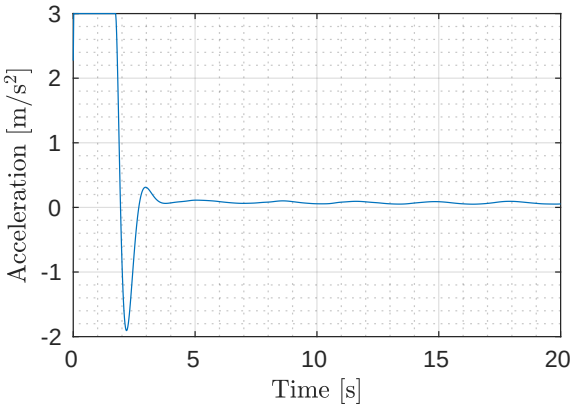
(b)

Fig. 11: Desired path and curvature of the ATLASCAR2 in a simulation of $20\,\mathrm{s}$

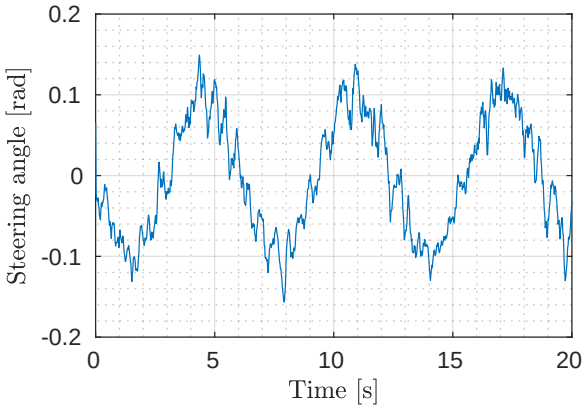
Moreover the following figures show the trend of the main parameters confirming that the control strategy used allows the vehicle to follow the path. In particular we simulated also a small error in the sensor dynamics in order to make the simulation more realistic: we added a 3 percent error to the longitudinal velocity and this is evident from the small noise in the graphs of the steering angle (Fig.12c) and the lateral deviation (Fig.12d). Figure 12a shows the evolution of the vehicle longitudinal velocity. At the start of the simulation, this velocity is equal to the initial condition for longitudinal velocity parameter V_0. At run time, we can note that V_x reaches the predefined value of $20\,\mathrm{m/s}$ meters for second and then it stabilizes near the cruising speed because it continues to vary the steering angle to adapt to the path to be followed. Finally we presents the overall scheme for the lane following developted in Simulink depicted in Figure 13.



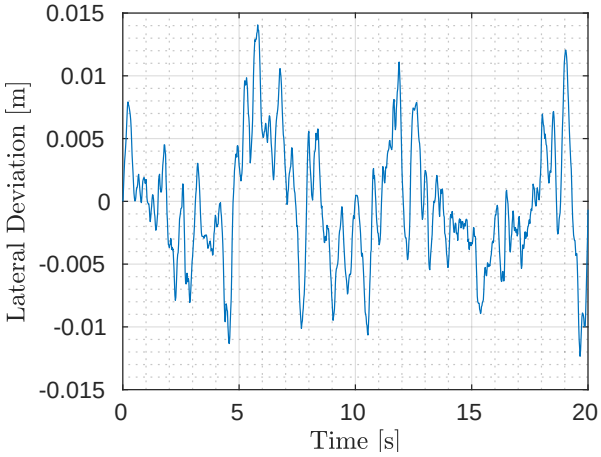
(a) Longitudinal velocity V_x with respect to time.



(b) Acceleration u_1 with respect to time.



(c) Steering angle u_2 with respect to time.



(d) Lateral deviation e_1 with respect to time.

Fig. 12: Time signals of the ATLASCAR2 in the simulation with a sinusoidal path

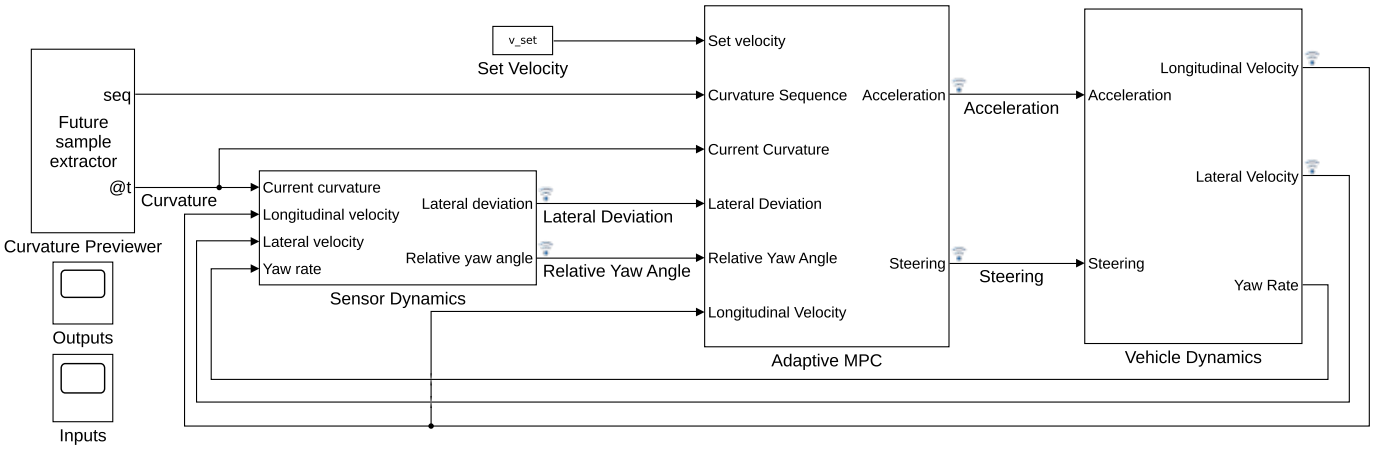


Fig. 13: Overall procedure scheme lane following.

VI. Conclusions and Future Work

This report proposes two advanced methods for short term motion planning of an autonomous car based on adaptive Model Predictive Control. The first component is an obstacle avoidance system that moves the vehicle around different moving obstacles in the lane using throttle and steering angle. This system updates both the predictive model and the mixed input/output constraints at each control interval. The vehicle is also able to brake in order to prevent collisions against closest obstacles. Instead, in the second scheme we have developed a lane following system that keeps the ATLASCAR2 traveling along the centerline of the lanes on the road by adjusting the front steering angle of the car. The flexibility of the concepts used in the algorithms allows a multitude of refinements and extensions to this work. The future work includes the combination of these two control strategies in a way that they can operate simultaneously. Next expected steps include the migration to ROS-Gazebo simulation environment and, later on, the usage of real data collected on board the ATLASCAR2 and, ultimately, test it in a real autonomous driving scenario.

References

[1] Kuwata, Y., Fiore, G. A., Teo, J., Frazzoli, E., and How, J. P., "Motion Planning for Urban Driving using RRT," Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Nice, France: 2008, pp. 1681–1686.

[2] Lei Tang, Songyi Dian, Gangxu Gu, Kunli Zhou, Suihe Wang and Xinghuan Feng, "A novel potential field method for obstacle avoidance and path planning of mobile robot," 2010 3rd International Conference on Computer Science and Information Technology, Chengdu, 2010, pp. 633-637.

[3] Hussein, H. Mostafa, M. Badrel-Din, O. Sultan, and A. Khamis, "Metaheuristic optimization approach to mobile robot path planning," in 1st International Conference on Engineering and Technology (ICET), New Cairo, Egypt, 2012.

[4] Y. Gao, T. Lin, F. Borrelli, E. Tseng, and D. Hrovat, "Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads," in Proceedings of Dynamic Systems and Control Conference (DSCC), 2010.

[5] L. Grune and J. Pannek, in Nonlinear Model Predictive Control: Theory and Algorithms. Springer London, 2011, pp. 43-66

[6] J. Škoda, '3D Navigation for Mobile Robots', Dissertation, 2017.

[7] Y. Nishio, K. Nonaka and K. Sekiguchi, "Moving obstacle avoidance control by fuzzy potential method and model predictive control," 2017 11th Asian Control Conference (ASCC), Gold Coast, QLD, 2017, pp. 1298-1303.

[8] Yu, S., Li, X., Chen, H. and Allgöwer, F. (2015), "Nonlinear model predictive control for path following problems". Int. J. Robust Nonlinear Control, 25: 1168–1182.

[9] T. Xu and H. Yuan, "Autonomous vehicle active safety system based on path planning and predictive control," 2016 35th Chinese Control Conference (CCC), Chengdu, 2016, pp. 8889-8895.

[10] M. Nolte, M. Rose, T. Stolte and M. Maurer, "Model predictive control based trajectory generation for autonomous vehicles — An architectural approach," 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, 2017, pp. 798-805.

[11] J. Ji, A. Khajepour, W. W. Melek and Y. Huang, "Path Planning and Tracking for Vehicle Collision Avoidance Based on Model Predictive Control With Multiconstraints," in IEEE Transactions on Vehicular Technology, vol. 66, no. 2, pp. 952-964, Feb. 2017.

[12] J. V. Frasch et al., "An auto-generated nonlinear MPC algorithm for real-time obstacle avoidance of ground vehicles," 2013 European Control Conference (ECC), Zurich, 2013, pp. 4136-4141.

[13] V. Turri, A. Carvalho, H. E. Tseng, K. H. Johansson and F. Borrelli, "Linear model predictive control for lane keeping and obstacle avoidance on low curvature roads," 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), The Hague, 2013, pp. 378-383.

[14] Monimoy Bujarbaruah, Xiaojing Zhang, H. Eric Tseng, Francesco Borrelli; Adaptive MPC for Autonomous Lane Keeping, 14th International Symposium on Advanced Vehicle Control (AVEC), Beijing, China, July 2018

[15] V. Santos et al., "ATLASCAR - technologies for a computer assisted driving system on board a common automobile", 13th International IEEE Conference on Intelligent Transportation Systems, Funchal, 2010, pp. 1421-1427.

[16] V. Santos, "ATLASCAR: A Sample of the Quests and Concerns for Autonomous Cars", Informatics in Control, Automation and Robotics: 14th International Conference, ICINCO 2017 Madrid, Spain, Revised Selected Papers (in press March 2019).

[17] R. Silva, "Navegação Local do ATLASCAR2 para Condução Autónoma e Assistência ao Condutor", Dissertação de mestrado. Departamento de Engenharia Mecanica, Universidade de Aveiro, 2018.

[18] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. "Robotics: Modelling, Planning and Control", Springer Publishing Company, Incorporated, 2008.

[19] M. Werling and D. Liccardo, "Automatic collision avoidance using model-predictive online optimization," 2012 IEEE 51st IEEE Conference on Decision and Control (CDC), Maui, HI, 2012, pp. 6309-6314.

[20] Wei Xi and J. S. Baras, "MPC based motion control of car-like vehicle swarms," 2007 Mediterranean Conference on Control & Automation, Athens, 2007, pp. 1-6.

[21] Y. Wang, Y. Bin and K. Li, "Longitudinal acceleration tracking control of low speed heavy-duty vehicles," in Tsinghua Science and Technology, vol. 13, no. 5, pp. 636-643, Oct. 2008.

[22] A. Bemporad, M. Morafi, and N. Ricker, Model Predictive Control Toolbox User's Guide, The MathWorks, Inc., https://www.mathworks.com/help/pdf_doc/mpc/mpc_ug.pdf

[23] Murali Madhavan Rathai, Karthik & Amirthalingam, Jegan & Jayaraman, Balaji. (2017). Robust tube-MPC based lane keeping system for autonomous driving vehicles. 1-6.

[24] Filip, Jan. (2018). Trajectory Tracking for Autonomous Vehicles.

[25] Danwei Wang, Feng Qi, "Trajectory planning for a four-wheel-steering vehicle," 2001 International Conference on Robotics and Automation, Seoul, Korea, 2001